UNIVERSITA' DEGLI STUDI DI PAVIA

FACOLTA' DI INGEGNERIA
DIPARTIMENTO DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

DOTTORATO DI RICERCA IN BIOINGEGNERIA E BIOINFORMATICA
XXIX CICLO - 2016

MATRIX FACTORIZATION TECHNIQUES FOR BIOMEDICAL DATA FUSION

PhD Thesis by
Andrea Demartini

Advisor:
Prof. Riccardo Bellazzi

PhD Program Chair:
Prof. Riccardo Bellazzi

Abstract (Italiano)

Negli ultimi anni, in molti settori si è assistito ad un'incredibile crescita della quantità di dati raccolti e sfruttati per diversi tipi di analisi. Questo è particolarmente vero per l'ambito biomedico, dove informazioni caratterizzate da una natura eterogenea sono oggigiorno disponibili per diversi scopi. Tuttavia, estrarre nuova conoscenza semplicemente combinando molteplici dati grezzi può essere un compito impegnativo, che richiede la definizione di nuove strategie e lo sviluppo di specifici strumenti. Infatti, questi dati sono spesso rumorosi, eterogenei a difficili da integrare, richiedendo così intense operazioni di pre-processing. Le tecniche tradizionali di machine learning sono in genere non adeguate a eseguire questo tipo di compito, poiché richiedono che tutti i dati di input siano strutturati in una forma specifica. In ogni caso, la disponibilità di una così grande quantità di dati pubblici ha stimolato lo sviluppo di opportune strategie di apprendimento finalizzate all'integrazione di diversi tipi di informazione. Questa operazione è comunemente nota come data fusion.

Questo lavoro è incentrato su una particolare classe di tecniche di data fusion, basate su metodi di fattorizzazione matriciale. Questi sono stati sviluppati e applicati con successo nell'ambito dei recommender system, con l'obbiettivo di predire in modo accurato i gusti di specifici utenti nei confronti di specifici prodotti. Grazie alla loro abilità di eseguire una riduzione della dimensioni del problema, sono in grado di evidenziare strutture latenti nascoste nei dati. Questa proprietà è cruciale in caso di dataset grandi e sparsi, che rappresentano una situazione comune in ambito biomedico.

Queste tecniche di fattorizzazione matriciale possono essere integrate in un contesto di machine learning tradizionale. Questo è il caso dei modelli di Factorization Machine, estensioni di metodi comuni di classificazione e regressione ma in grado di incorporare efficacemente le interazioni tra le variabili di input grazie all'utilizzo di una decomposizione matriciale. In questo modo, questi metodi possono sfruttare e rivelare relazioni sinergiche tra le proprietà misurate.

Altri metodi, sempre basati su fattorizzazione matriciale, possono essere direttamente impiegati per eseguire una data fusion. In questa dissertazione due di essi sono presentati: il metodo di Tri-fattorizzazione recentemente pubblicato e un metodo di nuova concezione basato su una fattorizzazione probabilistica Bayesiana. Entrambe queste tecniche richiedono che i dati di input siano espressi in forma di matrici relazionali, una per ciascun tipo di interazione modellizzata. Relazioni multiple devono coinvolgere gli stessi oggetti, al fine di propagare l'informazione tra le diverse sorgenti di dati. I

metodi operano una decomposizione congiunta di tutte le matrici di input, riassumendo le relative informazioni in vettori di dimensione ridotta. Una volta calcolati, questi vettori possono essere manipolati al fine di ricercare nuove interessanti associazioni tra coppie di diversi tipi di oggetti.

In questa tesi, viene presentata l'applicazione di due di queste tecniche, il metodo di Tri-fattorizzazione e le Factorization Machine. Entrambe i casi di studio riguardano particolari tumori del sangue, le neoplasie mieloidi.

Per quanto riguarda il metodo di Tri-fattorizzazione, è stato applicato a un insieme di diversi tipi di dati nell'ambito delle sindromi mielodisplastiche. Cinque tipi di oggetti e le loro relative associazioni sono state inclusi nel modello: pazienti, mutazioni, geni, malattie e processi biologici. Lo scopo del lavoro era di rivelare nuove interessanti interazioni gene-gene associate con la patologia in esame.

Il secondo caso di studio, invece, è incentrato sull'applicazione di un modello di Factorization Machine a un insieme di dati relativi a pazienti affetti da leucemia mieloide acuta. L'algoritmo di classificazione è stato addestrato per predire la gravità della malattia sulla base di alcuni dati personali (età, genere, razza) e includendo nel modello un insieme di mutazioni identificate per ciascun paziente. Anche in questo caso, particolare attenzione è stata data all'analisi delle interazioni tra geni mutati.

Per entrambe i casi di studio sono stati ottenuti risultati promettenti, suggerendo la capacità di questi metodi di sfruttare efficacemente tutta l'informazione a disposizione al fine di individuare associazioni non banali.

Nel dettaglio, la tesi è organizzata come segue:

Nel Capitolo 1 saranno descritti alcuni dei diversi tipi di dati biomedici con le loro caratteristiche peculiari. In particolare l'attenzione sarà posta sulla eterogeneità dei dati e sulle difficoltà legate alla data fusion.

Nel Capitolo 2 sarà presentata una panoramica delle tecniche più comuni di fattorizzazione matriciale e tensoriale, sottolineandone le loro proprietà matematiche.

Nel Capitolo 3 sarà discussa l'applicazione di metodi di fattorizzazione matriciale all'ambito del data mining. Saranno descritti i recommender system e i relativi algoritmi. In seguito, sarà introdotto il modello di Factorization Machines con le relative caratteristiche.

Nel Capitolo 4 saranno presentate due diverse tecniche di data fusion: il metodo di Tri-fattorizzazione e la fattorizzazione matriciale Bayesiana per la data fusion. Per entrambe i metodi, il modello sottostante, con i relativi parametri, sarà caratterizzato in dettaglio.

Nel Capitolo 5, in due diverse sezioni, saranno descritti i due casi di studio. Ciascuna sezione contiene una descrizione del problema iniziale

con i dati disponibili e lo scopo dell'analisi. In seguito, sono presentati e discussi la configurazione del metodo e i risultati dell'analisi.

Nel Capitolo 6 sono discussi le conclusioni finali e gli sviluppi futuri.

Abstract (English)

In the last few years many fields have experienced an incredible growth of the amount of data collected and exploited for different types of analysis. This is particularly true for the biomedical area, where information characterized by heterogeneous nature is nowadays available for many purposes. However, extracting new knowledge by simply combining multiple raw measures can be a challenging task, which requires the definition of novel strategies and the development of specific tools. In fact, these data are often noisy, heterogeneous, and difficult to integrate, requiring heavy pre-processing operations. The traditional machine learning techniques are generally inadequate to perform this kind of task, since they require all the input data to be structured in a specific form. However, the availability of such a big amount of public data has stimulated the development of proper learning strategies aimed at integrating different kinds of information. This operation is commonly known as data fusion.

This work is focused on a particular class of data fusion techniques, based on matrix factorization methods. These ones have been developed and successfully applied in the field of recommender systems, with the objective of predicting in an accurate way the tastes of specific users towards specific products. Thanks to their ability of performing a dimensionality reduction, they are able to highlight latent structures hidden in the data. This property is crucial in case of large and sparse datasets, which represents a common situation in the biomedical field.

These matrix factorization techniques can be integrated in a traditional machine learning framework. This is the case of Factorization Machines models, extensions of common classification and regression methods but able to effectively incorporate interactions between the input variables, thanks to the usage of matrix decomposition. In this way, these methods can exploit and reveal synergic relations between the measured features.

Other methods, still based on matrix factorization, can be directly employed to perform data fusion. In this dissertation two of them are presented: the recently published Tri-factorization method and a newly developed method based on a Bayesian probabilistic factorization. Both these techniques require the input data to be expressed in form of relation matrices, one for each type of modeled interaction. Multiple relations must involve the same objects, in order to propagate the information across the different data sources. The methods operate a joint decomposition of all the input matrices, summarizing the related information in low dimensional vectors. Once computed, these vectors can be manipulated in order to

investigate new interesting pairwise associations between different types of objects.

In this thesis, the application of two of those techniques, the Tri-factorization method and the Factorization Machines, is presented. Both the case studies focused on particular blood cancers, the myeloid neoplasms.

Regarding the Tri-factorization method, it has been applied to a set of different types of data within the context of the myelodysplastic syndromes. Five types of objects, and their related associations, were included in the model: patients, mutations, genes, diseases and pathways. The aim of the work was to point out novel interesting gene-gene interactions associated with the studied pathology.

The second case study, instead, is focused on the application of a Factorization Machines model to a set of data referring to patients affected by acute myeloid leukemia. The classification algorithm has been trained to predict the severity of the disease on the basis of some personal data (age, gender, race) and including in the model the set of mutations identified for each patient. Also in this case, particular attention was given to the analysis of the interactions between mutated genes.

For both the case studies, promising results were obtained, suggesting the capability of these methods to effectively exploit all the available information in order to detect non-trivial associations.

In details, this thesis is organized as follows:

In Chapter 1, some of the different types of biomedical data, with their peculiar characteristics, will be described. In particular the focus will be on data heterogeneity and the challenges of data fusion.

In Chapter 2, an overview of all the most common matrix and tensor factorization techniques will be presented, highlighting their underlying mathematical properties.

In Chapter 3, the application of matrix factorization methods to data mining will be discussed. The recommender systems and their related algorithms will be described. Afterwards, the Factorization Machines model with all its characteristics will be introduced.

In Chapter 4, two data fusion techniques will be presented: the Tri-factorization method and the Bayesian matrix factorization for data fusion. For both of them, the underlying model, with the related parameters, will be characterized in detail.

In Chapter 5, the two case studies will be described in two appropriate sections. Each of them contains a proper description of the starting problem with the available data and the purpose of the analysis. Afterwards, the description of the method's configuration and the results of the analysis are presented and discussed.

In Chapter 6, the overall conclusions and the future works will be discussed.

Contents

Introduction **1**
1.1. The heterogeneity of biomedical data *1*
1.2. Biomedical data sources *2*
1.3. Issues and challenges of biomedical data fusion *4*

Matrix factorization methods **7**
2.1. Basic concepts *7*
2.2. Single matrix factorization techniques *8*
2.2.1. Singular value decomposition 8
2.2.2. Principal component analysis 12
2.2.3. CUR matrix decomposition 13
2.2.4. Non-negative matrix factorization 15
2.2.5. Probabilistic factorization 17
2.3. Tensor factorization *21*
2.3.1. CP decomposition 23
2.3.2. Tucker decomposition 25

Matrix factorization for data mining **27**
3.1. Matrix factorization for recommender systems *27*
3.1.1. Collaborative filtering algorithms 29
3.1.1.1. Neighborhood methods 30
3.1.1.2. Latent factor models 31
3.2. Application to machine learning *34*
3.2.1. Factorization machines 34
3.2.1.1. FM Properties 36
3.2.1.2. Comparison with other methods 39
3.2.1.3. FM Parameters Learning 40
3.2.1.4. Stochastic Gradient Descent (SGD) 42
3.2.1.5. Alternating Least Squares (ALS) 44
3.2.1.6. Markov Chain Monte Carlo (MCMC) 46

Data fusion techniques **50**
4.1. Types of data integration *50*
4.2. Approaches based on matrix factorization *53*
4.2.1. Matrix tri-factorization algorithm 53
4.2.1.1. Input data 53
4.2.1.2. Method description 55
4.2.1.3. Parameters choice 57
4.2.1.4. Inference of new associations 58
4.2.1.5. Implementation details 60
4.2.2. Bayesian matrix factorization for data fusion 61
4.2.2.1. Input data 61
4.2.2.2. Model description 61
4.2.2.3. Parameters estimation though MCMC 63
4.2.2.4. Inference from predictive distribution 65
4.2.3. Comparison of the two methods 65

Data fusion in myeloid neoplasms **67**
5.1. Myelodysplastic syndromes *67*
5.1.1. Problem description 68

5.1.2. Available data 69
5.1.3. Preprocessing and matrices construction 70
5.1.4. Tri-factorization setting 72
5.1.5. Results 73
5.1.5.1. Genetic interaction networks 74
5.1.5.2. Enrichment analysis with Reactome 75
5.1.5.3. KEGG pathways analysis 77
5.1.5.4. Protein-protein interactions 79
5.1.5.5. Co-expression analysis 80
5.1.5.6. Conclusions 82
5.2. Acute myeloid leukemia *82*
5.2.1. Problem description 83
5.2.2. Available data 83
5.2.3. Data preprocessing 83
5.2.4. Factorization machines setting 85
5.2.5. Results 87
5.2.5.1. Literature analysis 88
Conclusions and future works 90
Example of application of the Bayesian factorization for data fusion 92
Additional results for MDS case study 95
Additional results for AML case study 102
References 107

Chapter 1

Introduction

This first introductory chapter is dedicated to the description of the data types commonly available in the biomedical field. Their peculiar characteristics, in fact, require the development and the application of specific techniques, aimed at exploiting in a proficient way all the available information. This dissertation is specifically focused on one of those characteristics: the data heterogeneity. Nowadays, different data sources are easily accessible, and this represents a great opportunity to increase our knowledge about many biological mechanisms and to improve many aspects of the medical process. Therefore in this chapter, the main characteristics of biomedical data will be discussed, with particular attention to the benefits of data integration.

1.1. The heterogeneity of biomedical data

Due to its intrinsic variability, the biomedical field is naturally characterized by inevitable data heterogeneity. Dealing with complex systems like the human organism, a lot of features are clearly necessary to properly describe the overall set of characteristics. As a matter of fact, healthcare is one of the most important generator of the so-called Big Data [1–4]. This expression has become very popular in the past few years across many disciplines. It denotes a class of data characterized by specific characteristics, typically referred to as the 4 Vs:

- Volume: the term "big" is often related explicitly to the dimensions of these data. The growth of the amount of data produced and collected led to the need of finding proper way to storage this information, ranging from terabytes to exabytes. In the biomedical field, this characteristic is typical of some molecular exams and of bioimages, which produce massive amounts of data for each patient.
- Velocity: this characteristic refers to the rate at which data are generated and transmitted. Hugh amounts of information are

nowadays collected very rapidly, making impossible the usage of traditional analytics methods. In addition, raw data are more and more frequently recorded by remote sensors and applications and transferred in real time through the network, generating great streams of data that need appropriate technologies to be managed. The social media, generating large amounts of data for each user on a daily basis, represents a clear example of this phenomenon. In addition, in a closer biomedical framework, telemedicine systems are increasingly used to remote monitoring the patient's conditions.

- Variety: this is a crucial aspect in many contexts, especially the biomedical one. Very often in fact, heterogeneous data are gathered to describe in a more holistic way a target system. Complementary information is collected, describing the system at different levels of granularity or from different perspectives. These data are often characterized by very different domains (e.g. counts, images, unstructured texts…). In addition, also the dimensional scales may greatly vary, from the molecular level to the population point of view. Moreover, measure may span over different time scales, from milliseconds to years. Therefore, for each particular category, specific data structures and analysis techniques have to be developed, making it challenging a joint integration of all the information sources.
- Veracity: this term refers to the uncertainty present in the data. It may be related to data inconsistency, incompleteness, intrinsic noise, and approximations. Also this characteristic plays a key role in the biomedical field. In fact, in this context data are very often noisy, measured in different settings and acquired with different accuracy levels, requiring heavy pre-processing operations to prepare them for the subsequent analyses.

In the next section, some of the most important sources of biomedical data will be introduced, with the related characteristic. In any case, particular emphasis will be posed on the heterogeneity of all these type of data.

1.2. Biomedical data sources

Depending on the particular context, many different types of data may be available for the analysis. The last few years have been characterized by an explosion of the amount of data collected and made publically accessible by different sources. Other data may be provided by health institutions, typically consisting of a set of electronic health records about the treated patients. Some of these patients may be part of cohorts specifically selected on the basis of some discerning criteria and for specific purposes (e.g. clinical trials).

From the single patient perspective, many kinds of information may be collected. Electronic health records typically contain some personal data, such as age, gender, race and more specific features based on the particular condition of the patient. For example, on the basis of the diagnosis, specific vital parameters may be measured, and specific information about pre-existing conditions, exposition to risk factors and family medical history may be acquired. On the basis of patient's clinical path, different diagnostic exams may be performed, and different laboratory tests may be carried out. Therefore it is clear that this type of data source itself may provide very heterogeneous information: for example molecular data and bioimages belong to completely different domains and required *ad hoc* techniques to be analyzed. In particular, data structures can be completely different, including big tabular files, for example for personal data and results of laboratory exams, images and signals graphs, unstructured text for the notes by the physician.

The previously cited types of data anyway, are part of the traditional clinic features: the only different of recent times is the attempt of digitalization of all the information. But the greatest impact of the evolution of modern medicine is represented by another source of information: the molecular data. Thanks to new cheaper sequencing technologies, it is possible to easily get information about single patients' mutations and gene expression values [5]. Due to their peculiar characteristic, also these data require appropriate techniques to be stored and manipulated.

Switching to public data sources, a large number of databases are easily accessible, especially in the field of Bioinformatics, thanks to -omics data (such as genomics, proteomics, metabolomics) repositories. Typically each of them focuses on a specific topic, but there are many overlap and cross-references between all of them. In 2016 the journal Nucleic Acids Research published a special issues on biological databases containing a list of about 180 different databanks [6].

Some of them focus on nucleic acids (DNA, RNAs) under different points of view (sequence, structure, regulation, expressions, interactions, relations with phenotype). For example, databases such as GenBank [7] and the EMBL database [8] contain annotated collection of publicly available DNA sequences. This information is obtained through submissions from individual laboratories and from large-scale sequencing projects.

A dual information, but from a protein point of view is collected by other data banks such as UniProt [9] and Swiss-Prot [10].

Other public databases, instead, focus on diseases, try to model in a comprehensive ontology all the relations among them (e.g Disease Ontology [11]). Some of them, in particular, focus on diseases with a genetic component (e.g. OMIM [12]).

Furthermore, the most important biological processes occurring inside a cell have been deeply studied and modeled, and this information is

available thanks to public databases such as KEGG [13] and Reactome [14].

Another interesting data source is represented by the molecular interactions. For example, the public data repository STRING [15] contains protein-protein interactions coming from different knowledge bases, and combine them in order to obtain confidence scores about the associations. Same thing can be done directly for genes, as contained in BioGRID [16].

Many others can be listed, depending on the particular problem to address. The important thing to underline is that these sources are not independent from each other and many noteworthy results may be obtained from a joint integration of all the knowledge they contain.

Of course, another primary source of information is directly represented by the scientific literature. However, it is often hard to mine useful information from such a huge mass of documents. For this reason, some controlled vocabularies have been proposed in order to index journal articles and books, thus facilitating the search. An example in life science is represented by the Medical Subject Headings (MeSH) [17]. These terms can be put in direct association with other objects, such as diseases or genes, thus linking information coming from different sources.

Another type of data that is gaining more and more interest in the last years, also in the biomedical field, comes from the social media. This huge stream of data may provide useful feedbacks from a patient, even if it requires heavy text processing operation, due to the unstructured text often characterizing this type of data [18].

At last, a new promising source of information is represented by exposomics [19]. This new field of research aims at correlating the health condition with some environmental factors to which the people are exposed. For example, interesting associations can be found between pollution levels and specific pathological conditions.

1.3. Issues and challenges of biomedical data fusion

The availability of huge amounts of data represents, of course, a great opportunity both for the clinical practice and the biomedical research. In particular, all of these huge public repositories, even if dealing with different topics, are characterized by many mutual interactions, thus suggesting the possibility of a propagation of the information across them. Traditional data mining methods are often inadequate to treat so heterogeneous types of data, because they were developed to address specific problems. A way to overcome this problem would be to perform heavy pre-processing operation in order to represent the input data in a

more traditional fashion (e.g. tabular datasets containing different features), but in this way the inner structure of data coming from different data sources would be lost. Therefore, new strategies and tools are required to effectively manage a so variegated class of data [20].

Considering clinical practice, an integrated system would be a crucial factor for the improvement of system. For example, the integration of imaging, modeling, and real-time sensing can be very useful for the management of disease progression and the planning of intervention procedures [21].

In addition, on the basis of the complete set of data about a patient, an analogical reasoning could be performed in order to identify similar patients, thus allowing the translation of the related treatments [22]. This is particularly interesting under the perspective of the precision medicine. This expression refers to an emerging approach for disease treatment and prevention, more focused on individual patients or small groups of patients. The aim is to classify people into subpopulations that differ in their susceptibility to a particular disease, in the biology and/or prognosis of the diseases they may develop, or in their response to a specific treatment [23]. For this reason, an effective integration of different data sources may represent a valid evidence-based approach to the decision process for diagnosis and therapy. This is of course particularly crucial for oncologic patients, often characterized by very specific genetic alterations, which make difficult the development of treatments with general effectiveness.

From the research point of view, the availability of these huge amounts of data is an incredible stimulus to the development of data mining techniques able to discover latent characteristics hidden in this massive stack of information. Using a data driven approach and exploiting the connections between different data sources, it would be possible to highlight complex interactions among different objects, hard to detect using standard approaches. This may lead to methods able to suggest new research hypothesis, perform prediction and data interpolation.

Tools may be designed to help biologist in the integration of multiple heterogeneous public sources with their own experimental data [24].

Another possible application is the drug repurposing [25]: already developed drugs may show a potential effectiveness for other types of diseases due to the fact that they operate on the same biological mechanisms.

Regarding the genomics field, the research in this area is very intensive, and large, highly heterogeneous data are produced continuously. The need of finding a suitable representation of this knowledge and linking these heterogeneous data sets is a clear issue that has to be addressed in the very near future [26].

For all these reasons, the development of new strategies and techniques to operate a fusion of data coming from different sources represents currently a hot topic in the biomedical research field.

Chapter 2

Matrix factorization methods

This chapter is focused on the description of the main factorization techniques developed for bidimensional and multidimensional arrays (i.e. matrices and tensors). In the first section, some basic concepts will be introduced. Afterwards, a set of the most common techniques for both matrices and tensors will be described.

2.1. Basic concepts

In Mathematics, the term factorization indicates, generically, a procedure aimed at performing the decomposition of a particular object. The name comes from the fact that, at the end of this procedure, the initial object is expressed as the product of a certain number of elements, the so-called factors. This work is focused only on matrix and tensors factorization techniques, meaning that the starting object is a 2-dimensional or N-dimensional vector. There are several methods designed to perform this type of decomposition, each of them characterized by different mathematical properties, and therefore more or less suitable to address specific problems. In any case, these techniques typically express the values of the starting object through the product of several elements. Depending on the method, the number and the characteristics of the factors may vary a lot, as well as the output of the procedure, which can lead to an exact or an approximate representation of the initial data. Most of the techniques that will be discussed are primarily used for dimensionality reduction, meaning that they try to compress the starting information using a lower number of features (i.e. vector components). This operation can be useful from many points of view. First of all, it allows compressing the space needed to store the data. This can be very critical in case of large sparse datasets, like for examples those commonly used in the field of recommender systems (more on that will be discussed in the next chapter).

In addition, from the computational point of view such reduction may lead to much faster operations, particularly useful in case of big data applications and online processes. From the conceptual point of view instead, the decomposition methods play a central role in the analysis of the latent structures hidden in the data. The mathematical basis of these operations may give interpretability properties to the computed factors, revealing unknown interactions of the initial data. For example, the reduction of the dimensionality obliges the method to identify common characteristics of the data, summarizing them with a small number of elements. For all these reasons, matrix and tensor factorization techniques are gaining more and more interest in many different applications. Next sections will focus on very popular methods, highlighting their principal properties and their weaknesses.

2.2. Single matrix factorization techniques

This section is dedicated to four different types of decomposition specifically designed to target a single bidimensional array, i.e. a matrix.

2.2.1. Singular value decomposition

The singular value decomposition (SVD) is one of the most popular decomposition techniques [27–32]. From the mathematical point of view, the SVD decomposes a matrix into the product of three terms:

$$M = U\Sigma V^* \tag{1}$$

where:

M is the starting matrix, which can be a real or complex, with dimension $m \times n$ and rank r.

U is a $m \times m$ real or complex matrix. If U is real, then it is also orthogonal, i.e. its transpose U^t is also its inverse U^{-1}. If U is complex, then it is unitary, i.e. its conjugate transpose U^* is also its inverse.

Σ is a $m \times n$ rectangular diagonal matrix, meaning that it contains non-zeros values only on the main diagonal. In particular, in SVD these entries are non-negative real numbers, ordered by decreasing value of magnitude. Since M has rank r, there will be exactly r strictly positive values on the diagonal, while all the others will be zeros.

V is a $n \times n$ real or complex matrix. Like U, it is unitary (and therefore orthogonal if real).

This type of decomposition is strongly related to the concept of singular value. By definition, given the matrix M, a singular value σ is a non-negative real number such that:

$$Mv = \sigma u \; and \; M^{*}u = \sigma v \tag{2}$$

where u and v are unit-length vectors. They are respectively called left-singular and right-singular vectors for σ. The relation with the SVD is pretty straightforward: the positive entries on the diagonal of the Σ matrix are the singular values of M, while the first p=min(m,n) columns of U and the first p columns of V are, respectively, the associated left-singular vectors and the right-singular vectors.

In the following, an example of SVD decomposition is shown. Given a matrix M:

$$M = \begin{bmatrix} 3 & 1 & -1 & 0 \\ 1 & 6 & -1 & 2 \\ 4 & 0 & 1 & 4 \end{bmatrix} \tag{1}$$

it can be decomposed in:

$$M = U\Sigma V^{*}$$

$$U = \begin{bmatrix} -0.34 & 0.1 & 0.93 \\ -0.77 & -0.6 & -0.21 \\ -0.54 & 0.79 & -0.29 \end{bmatrix}$$

$$\Sigma = \begin{bmatrix} 7.4 & 0 & 0 & 0 \\ 0 & 5.08 & 0 & 0 \\ 0 & 0 & 2.91 & 0 \end{bmatrix} \tag{2}$$

$$V = \begin{bmatrix} -0.54 & 0.56 & 0.5 & -0.38 \\ -0.67 & -0.7 & -0.12 & -0.24 \\ 0.12 & 0.23 & -0.67 & -0.7 \\ -0.5 & 0.38 & -0.54 & 0.56 \end{bmatrix}$$

It can be easily noticed that the Σ matrix is diagonal, with the singular values (7.4, 5.08, 2.91) sorted in decreasing order of magnitude. The three columns of U represent the left singular vectors, while the first three columns of V represent the right singular vectors.

The transformation f(x)=Mx maps a vector in $\mathbb{R}^n$ into a vector in $\mathbb{R}^m$. It can be shown that the columns of V and U provide orthogonal bases for the domain and the range of f(x) [29].

For real squared matrices, SVD has a very intuitive interpretation [29]. Each of the three matrices can be associated to a transformation so that the entire process can be interpreted as the composition of three geometrical operations: U represents a rotation, Σ represents a scaling and V represents another rotation. In practice, the transformation f(x)=Mx dilates or contracts some components of x (after the first rotation), on the basis of the magnitude of the associated singular values. Because of the different

dimensions between domain and range, some components may be discarded or some zeros may be appended.

Given this interpretation, there is clearly a strong correlation between SVD and the well-known eigenvalue decomposition. Given a square matrix $A \epsilon C^{mxm}$, its eigenvalue decomposition is:

$$A = X\Lambda X^{-1} \tag{3}$$

where:

Λ is a $m \times m$ diagonal matrix and its entries are called eigenvalues.

X is a $m \times m$ matrix, whose columns are linearly independent vectors called eigenvectors.

Even if both these decompositions try to express the starting matrix in a diagonal form, they differ for many reasons. First of all, SVD can be computed for every matrix, even rectangular ones. Eigenvalues decomposition, on the contrary, can be applied only on particular classes of squared matrices. In addition, SVD utilizes two different (orthonormal) bases, composed respectively by the set of left and the set of right singular vectors, while the eigenvalue decomposition uses just one (in general not orthonormal) basis, determined by the eigenvectors. The two decompositions coincide only for positive semi-definite normal matrices.

But there is also another important association between the two operations, which can be expressed and demonstrated through the following formulas:

$$\begin{aligned} MM^* &= U\Sigma V^* V\Sigma^* U^* = U(\Sigma\Sigma^*)U^* \\ M^*M &= V\Sigma^* U^* U\Sigma V^* = V(\Sigma^*\Sigma)V^* \end{aligned} \tag{4}$$

As can be easily noticed, the left singular vectors represent the eigenvectors of MM^*, while the right singular vectors represent the eigenvectors of M^*M. In the same way, the non-zeros singular values of M can be seen as the square roots of the eigenvalues of MM^*or M^*M.

Another way to describe the SVD is using the outer product form [27]. By definition, the outer product of two vectors $u \in \mathbb{R}^n$ and $v \in \mathbb{R}^m$ (in symbols $u \otimes v$) is a matrix $X \in \mathbb{R}^{nxm}$, whose generic entry is given by $X_{ij} = u_i v_j$. This matrix has rank equals to 1.

Within this framework, the SVD of the matrix M, can be also expressed as:

$$M = \sum_{i=1}^{r} X_i = \sum_{i=1}^{r} \sigma_i u_i \otimes v_i \tag{5}$$

Where σ_i is the i-th singular value, with u_i and v_i the associated left and right singular vectors. The limit of the summation, r, represents the rank of the M matrix, and corresponds also to the number of non-zeros singular values. From this perspective, SVD can be interpreted as a weighted sum of rank-one matrices. (See Section 2.3.1 CP decomposition for the tensor equivalent).

This representation is also useful to show another important property of the decomposition. Keeping in mind that the singular values are sorted by decreasing value of magnitude, the summation defined above can be truncated after $\upsilon < r$ terms:

$$M_\upsilon = \sum_{i=1}^{\upsilon} X_i = \sum_{i=1}^{\upsilon} \sigma_i u_i \otimes v_i \tag{6}$$

From the mathematical point of view, it can be demonstrated that M_υ represents the best υ-rank approximation of the original matrix M, in terms of both 2-norm and Frobenius norm [29]. In addiction, this relation holds:

$$\|M_\upsilon - M\|_2 = \sigma_{\upsilon+1} \tag{7}$$

This is a remarkable characteristic from a dimensionality reduction perspective, since it allows keeping just the most important singular values, discarding all the others.

This property is crucial for all the applications that require a low rank approximation of the starting matrix. The idea behind this approach is that the knowledge represented in the original matrix can be expressed as a combination of several (latent) components, some of them very informative (those characterized by a high singular value), others negligible (those characterized by a low singular value). For this reason, even if the summation is truncated, the most explanatory characteristics are in any case maintained by the decomposition.

Due to its characteristics, SVD is exploited in many different applications. First of all, it is possible to compute the pseudo-inverse of the original matrix [33]:

$$M^+ = V\Sigma^+ U^* \tag{8}$$

Where Σ^+ is the pseudo-inverse of the matrix Σ, and it's obtained by substituting all the non-zero entries on the diagonal with the associated reciprocal, and then transposing the resulting matrix. This operation can be very useful for example when solving a linear least squares problem.

SVD and its truncated version have also been used extensively in signal processing [34,35] and for image compression [36,37].

However, the most interesting application of SVD is in the field of Recommender systems, which will be extensively described in the following chapter.

2.2.2. Principal component analysis

The SVD is also related to another popular technique, used typically for dimensionality reduction in data mining: the Principal Component Analysis (PCA) [38]. As the name suggests, the objective of this method is to identify the most informative components characterizing in the input dataset. It is applied when the dataset is structured as a matrix, whose rows represent a group of samples, and the columns represent the associated features. The PCA operates an orthogonal transformation on the original data, projecting them into a new coordinate system. The transformed features are called principal component and they are linearly uncorrelated. The first component is characterized by the largest possible variance; the remaining ones are sorted by decreasing order of captured variability. To describe how the methods works, is thus necessary to start from the covariance matrix of the data.

Starting from a matrix D, with columns mean subtracted and shifted to zero, the covariance matrix C is given by $C = D^t D/(N-1)$, where N is the number of rows of D. By construction, this matrix is symmetric and so it can be diagonalized with eigenvalue decomposition:

$$C = X\Lambda X^{-1} = X\Lambda X^{\mathrm{t}} \tag{9}$$

The eigenvectors, represented by the columns of the X matrix, are called principal axes or principal directions. Since C is symmetrical, the spectral theorem states that X is orthogonal, so $X^{-1} = X^{\mathrm{t}}$. The principal components mentioned in the name of the method are the projections of the data on the principal axes, and they are computed as the columns of a matrix, DX, obtained by multiplying the data matrix D by the eigenvectors matrix X.

To evaluate the relation between PCA and SVD, let's perform SVD on the data matrix D:

$$D = U\Sigma V^{\mathrm{t}} \tag{10}$$

The covariance matrix thus becomes:

$$C = \frac{D^t D}{N-1} = \frac{V\Sigma^t U^t \mathrm{U}\Sigma V^{\mathrm{t}}}{N-1} = V\frac{\sigma^2}{N-1}V^t \tag{11}$$

This means that the right singular vectors in V are principal directions (X=V) and that singular values are related to the eigenvalues of covariance matrix by the equivalence $\lambda_i = \sigma_i^2/(N-1)$. In the same way, the principal components can be expressed in terms of SVD as:

$$DX = U\Sigma V^t V = U\Sigma \tag{12}$$

As for the truncated version of SVD, it is possible to keep just the most important components and discarding those characterized by a low associated eigenvalue. For PCA, the singular values have also a statistical meaning, since they are related to the amount of variance captured by the related singular component. For these reasons, PCA is used in many applications for dimensionality reduction.

In the following, an example of PCA is described. Starting from a data matrix D:

$$D = \begin{bmatrix} 0.508 & 0.929 & 0.459 \\ 0.086 & 0.730 & 0.963 \\ 0.2625 & 0.489 & 0.547 \\ 0.801 & 0.579 & 0.521 \\ 0.029 & 0.237 & 0.232 \end{bmatrix} \tag{3}$$

The eigenvalues and the eigenvectors associated to the covariance matrix are computed:

$$\Lambda = \begin{bmatrix} 0.125 & 0 & 0 \\ 0 & 0.093 & 0 \\ 0 & 0 & 0.022 \end{bmatrix}$$
$$X = \begin{bmatrix} 0.76 & -0.538 & -0.364 \\ 0.596 & 0.352 & 0.722 \\ 0.26 & 0.766 & -0.588 \end{bmatrix} \tag{1}$$

By multiplying D and X, the original data are projected into the new space where the features are linearly uncorrelated. The first component, associated to the eigenvalue 0.125, explains the 52.17% of the entire variance (0.125 divided by the sum of all the eigenvalues), while considering the first two components, the 90.84% of the variance is explained. Therefore, in some application it could be possible to exclude the third component from the subsequent analyses, thus performing a data size reduction.

2.2.3. CUR matrix decomposition

As highlighted in the previous sections, one of the most important applications of matrix factorization techniques is related to dimensionality reduction. SVD and PCA can be valuable tools to provide low rank approximations of a data matrix, trying to capture as much information as possible. However, since they operate a transformation of the original data, it is difficult to assign an explicit meaning to the elements resulting from the decomposition. For example, with PCA the principal components are linear combination of the original features, so they lack of a proper actual

interpretation [39]. A common temptation is to try to associate a meaning to these transformed variables. This operation is called reification, and it may lead to misleading conclusions.

In some applications, the interpretability of the results is a crucial factor, especially when the final aim is to extract insights from the analysis. For this reason, another factorization method, called CUR decomposition, has been proposed. This technique tries to approximate a starting matrix M with the product of three terms:

$$M \approx CUR \tag{13}$$

where:

C is a matrix consisting of a small number of columns taken from M

R is a matrix consisting of a small number of rows taken from M

U is a matrix defined by minimizing the reconstruction error produced by the product of the three matrices, typically measured in term of $\|M - CUR\|_2$ or $\|M - CUR\|_{Frobenius}$.

In an equivalent way to SVD, it is possible to obtain a low rank approximation of the original matrix. For example, a rank-k approximation is achieved by fixing to k the number of columns of C and the number of rows of R. In this case, U matrix will be a squared matrix $k \times k$.

Although CUR decomposition is less accurate of SVD, it provides a natural interpretation, since both C and R are composed of actual vectors from M. There exist different algorithms to compute the decomposition, mainly distinguished by the way the columns and rows are selected from the original matrix. The simplest version of CUR is based on a uniform random sampling of these vectors [40]. More sophisticated algorithms try to weight this sampling using a probability for each column (or row) based on the related Euclidean norms [41]. Others aim at reducing the relative error by directly taking into account the influence of the vectors on the approximation of the original matrix [39].

In addition to interpretability, CUR has another interesting property, related to the sparseness of the starting matrix. Very often in fact, CUR is applied to matrices characterized by huge dimensions, but with a very large number of zero entries. If SVD is applied, this characteristic is lost after the decomposition, because the two matrices of left and right singular vectors are still huge, but they are dense in general. The matrix containing the singular values will be instead diagonal and thus very sparse. On the contrary, since CUR uses columns and rows from the starting matrix, both C and R will be characterized by high sparseness, while the U matrix, although dense, will be small. This characteristic can be crucial from the computational point of view, guaranteeing a much lower computational time and memory cost to storage the results.

2.2.4. Non-negative matrix factorization

In certain applications, constraints on the results of the decomposition are required. In many fields, the collected data are typically real positive numbers, therefore a factorization with non-negativity condition can provide a more natural representation and interpretation of the results. In addition, non-negativity constraints may lead to much sparser representations, particularly important in case of high dimensional problems. For this reason, a class of decompositions has been defined, falling under the name of non-negative matrix factorizations (NMF) or non-negative matrix approximation [42,43]. Since an exact solution of the problem can be difficult to compute, often these methods aim at finding a good approximate representation of the original matrix. Given a non-negative matrix M, NMF determines a lower rank non-negative approximation given by the product of two matrices, W and H, both of them non-negative too:

$$M \approx WH, \quad with\, W \geq 0\, and\, H \geq 0 \tag{14}$$

This operation can be expressed also using a vector representation. In this case, the i-th column from M, $\boldsymbol{m_i}$, is expressed as a linear combination of the columns of W, weighted by the values of the i-th column from H:

$$\boldsymbol{m_i} \approx W\boldsymbol{h_i} = \sum_j h_{ji}\boldsymbol{w_j} \tag{15}$$

From this point of view, W can be interpreted as a suitable basis for the linear approximation of the data in M. For this reason W is also called component matrix and H is called mixing matrix [44].

There exists also a strong correlation between this decomposition and some well-known clustering techniques. In particular, imposing the orthogonality constraint on the H matrix, it is possible to demonstrate that this factorization is equivalent to a K-means clustering applied to the columns of the starting matrix M [45]. It is also possible to exploit this method in order to perform a simultaneous clustering of rows and columns of a matrix [45].

The solution of this factorization problem is not unique, and numerical approximations are typically employed. First of all, the dimensions of W and H have to be chosen. If M is a $m \times n$ matrix, the W has dimension $m \times p$ and H has dimensions $p \times n$. As easily understandable, the value of p is critical in practice, and it often depends on the specific problem [46]. A common choice is to select a value of p that is much lower than both m and

n. The result of the operation will be therefore a compressed version of the original data matrix, easier to store and manipulate. In addition, as for SVD and PCA, factorizing a data matrix into smaller matrices allows highlighting the presence of latent structures in the data.

As mentioned above, there are multiple algorithms to compute non-negative matrix factorizations. First of all, a cost function must be defined to evaluate the quality of the approximation. A common choice is to use the Frobenius norm of the approximation error. In this case, a solution to the NMF problem can be obtained by solving an optimization problem:

$$\min_{W,H} f(W,H) = \frac{1}{2}\|M - WH\|^2_{Frobenius} \tag{16}$$

An example of NMF using this strategy is shown in the following.
Given a matrix M:

$$M = \begin{bmatrix} 0.404 & 0.942 & 0.06 & 0.821 \\ 0.1 & 0.96 & 0.235 & 0.015 \\ 0.132 & 0.575 & 0.353 & 0.04 \end{bmatrix} \tag{1}$$

it can be decomposed using rank-2 matrices:

$$\begin{aligned} M \approx WH &= \begin{bmatrix} 1.242 & 0.104 \\ 0.054 & 0.946 \\ 0.022 & 0.658 \end{bmatrix} \begin{bmatrix} 0.315 & 0.681 & 0.018 & 0.661 \\ 0.118 & 0.932 & 0.342 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0.403 & 0.943 & 0.058 & 0.821 \\ 0.129 & 0.919 & 0.325 & 0.036 \\ 0.085 & 0.628 & 0.225 & 0.015 \end{bmatrix} \end{aligned} \tag{2}$$

It can be noticed that all the entries of the two decomposition matrices W and H are positive. This is a very simple example, using a small dense matrix, but more significant and useful results can be obtained on very large and sparse matrices.

Additional constraints, depending on the problem-specific prior knowledge, can be included to the model. They often are expressed as regularization terms, like L1-norm penalty (Lasso), favoring characteristic like sparseness, smoothness, or specific relationships between components [44,47].

Many algorithms have been developed to perform this type minimization. The problem of finding the global minimum is particularly complex, even without constraints. However, there are many numerical techniques that can be applied to find local minima. The simplest method is probably the well-known gradient descent, but it is characterized by a slow convergence [43]. More complex but faster techniques have been proposed. In particular, some iterative methods are very used, one based on multiplicative rules and another based on additive rules. The main idea is to

iteratively update one matrix at a time by multiplying or adding a term dependent on the approximation error [43].

Anyway, in general the standard NMF algorithms utilize the complete data matrix during the entire estimation problem. This can be critical for applications where the amount of data is huge and hard to fit into memory or if there is a continuous stream of data. For example, for collaborative filtering in recommendation systems, the data matrix is composed of a lot of users and a lot of items, and it would be very resource consuming to recalculate the factorization every time new information is added to the system. For this reason, more sophisticated algorithm has to be used [48].

NMF has been employed in many fields.

As mentioned above, a very common application is collaborative filtering for recommender systems [49]. In this case, the two factors of the decomposition can be interpreted as a user-specific matrix and an item-specific matrix. Then the method can be exploited to propose new items for each specific user.

Another common application of NMF is text mining. In this case, the starting matrix contains information about the associations between documents and terms. Each entry usually represents the importance inside a certain document of a certain word, in general measured on the basis of its frequency in the text. In this case, the two matrices produced by the decomposition can be used to identify homogeneous groups of documents based on the terms they contain. For example, this strategy has been applied to documents from PubMed to extract publications related to a specific topic [50].

NMF has been also applied to the Bioinformatics field. For example, this technique allows to individuate clusters of genes based on their expression levels measured in many samples [44,51,52]. It has also been used to identify molecular signatures for human cancers based on somatic mutation patterns [53].

2.2.5. Probabilistic factorization

The problem of matrix factorization can be addressed also in a probabilistic fashion [54]. Following this approach, random variables with appropriate distributions come to play, and they are used to model the data and the latent factors behind them. The basic idea is to consider the generic entry of the data matrix as a stochastic variable, whose distribution depends on two factors, specifically two low dimensional vectors, one related to the element representing the row and the other related to the element representing the column. The figure 2.1 shows the structure of the model as reported in [54].

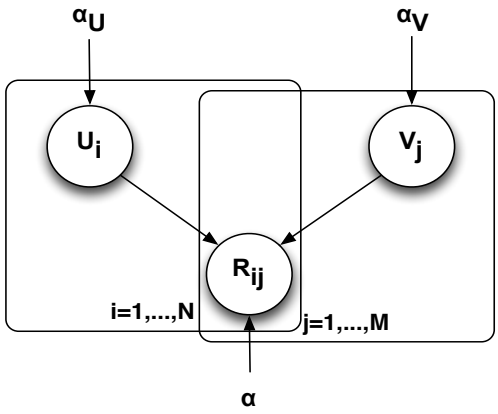

Figure 2.1: Probabilistic Matrix Factorization Model

This method has been firstly applied to the context of the recommender systems: each entry of the starting matrix, R_{ij} in the figure, represents the rating of i-th user with respect to the j-th movie. In this case, the starting matrix R is commonly large and very sparse, since in general each user rates just few movies. In addition, it is often very imbalanced, with some rows and columns characterized by just few values. The objective of the method is therefore to fill the gaps, using the information coming from the other users. In this context, the two factors have a clear interpretation: one, U_i in the figure, is a user-specific vector and the other, V_j, a movie-specific vector. In practice, to each row and column of the starting matrix a low dimensional vector is associated, summarizing the information about the related object. The same dimension, D, which is the critical parameter of the model, characterizes all the vectors: low values are not enough to represent the information, too high values may reconstruct well the original matrix but without discovering the latent structure inside the data, and therefore they are not useful for prediction. The assumption of an underlying probability distribution can be also fundamental for imbalanced datasets, since it is always possible, even with little information, to suggest a value for each entry of the matrix.

In [54], Salakhutdinov et al. proposed to use a probabilistic linear model with Gaussian noise, characterized by the following conditional distribution over the entries of the starting matrix R:

$$p(R|U,V,\alpha) = \prod_{i=1}^{N}\prod_{j=1}^{M}\left[N\left(R_{ij}|U_i^t V_j, \alpha^{-1}\right)\right]^{I_{ij}} \tag{17}$$

where N represents the Gaussian distribution with mean $U_i^t V_j$ and precision α (i.e. the observation noise variance), and I_{ij} is the indicator variable that is equal to 1 if the entry R_{ij} is observed and equal to 0

otherwise. The prior distributions over the low dimensional vectors are Gaussian too, with zero mean and an appropriate precision. For example, for U:

$$p(U|\alpha_U) = \prod_{i=1}^{N} N(U_i|0, \alpha_U^{-1} I) \tag{18}$$

The estimation problem can thus be expressed as the maximization of the posterior distribution $p(U, V|R, \alpha, \alpha_U, \alpha_V)$. As shown in [54], this is equivalent to minimize the following objective function:

$$E = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{M} I_{ij} (R_{ij} - U_i^t V_j)^2 + \frac{\alpha_U}{2\alpha} \sum_{i=1}^{N} \|U_i\|_F^2 + \frac{\alpha_V}{2\alpha} \sum_{j=1}^{M} \|V_j\|_F^2 \tag{19}$$

As easily observable, this measures the reconstruction error through a sum of squares, with the addition of two quadratic regularization terms, one for each dimension. There is not a close solution to find a global solution for this problem, so other methods like gradient descent are required to find a local minimum. Anyway, the problem scales linearly with the number of observed entries.

The two ratios, α_U/α and α_V/α, can be interpreted as regularization parameters. Their choice is critical to make the model generalize well, especially when very sparse and imbalances datasets are considered. A possible way to find values is to try different reasonable combinations on a training set, and test them on a validation set to identify the best one. Anyway this solution is very expensive from the computational point of view and it is not suitable for many applications.

An alternative solution is to introduce priors for the hyperparameters α_U and α_V, and then maximize the posterior distribution over both parameters and hyperparameters. In this way, the model complexity is controlled automatically using just the training data.

Another alternative is to exploit a fully Bayesian approach, as explained by Salakhutdinov et al. in [55]. The model of this Bayesian matrix factorization is depicted in figure 2.2.

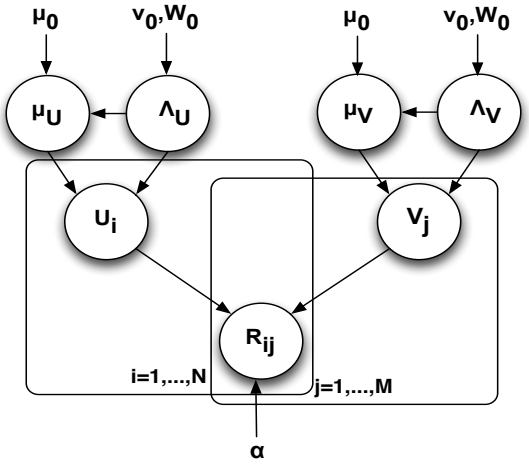

Figure 2.2: Bayesian factorization method

First of all, they placed prior distributions on top of the hyperparameters U_i and V_j. Following these assumptions, the distribution of the vector U can be expressed as:

$$p(U|\mu_U, \Lambda_U) = \prod_{i=1}^{N} N(U_i|\mu_U, \Lambda_U^{-1}) \tag{20}$$

where μ_U and Λ_U denote the hyperparameters of U, represented by the mean and the precision matrix of the associated Gaussian distribution. A convenient choice in a Bayesian context is to assume that the hyperparameters have Gaussian-Wishart priors [56]. In this case, their probability distribution is:

$$p(\mu_U, \Lambda_U|\mu_0, \beta_0, W_0, \nu_0) = N(\mu_U|\mu_0, (\beta_0\Lambda_U)^{-1})W(\Lambda_U|W_0, \nu_0) \tag{21}$$

where $\mu_0$and β_0 represent the mean and the precision of the Gaussian distribution, while υ_0 and W_0 represent the degrees of freedom and the *DxD* scale matrix characterizing the Wishart distribution W.

As regards the predictive distribution, it is necessary to compute a complex posterior distribution by integrating over all the parameters and hyperparameters. Since the analytical derivation of this distribution is unfeasible, approximate inference has to be used.

One possibility is to use the so-called variational methods [57]. The main idea of these techniques is to factorize the posterior distribution over some partition of the latent variables. Each factor is assumed to have a specific parametric form such as a Gaussian distribution. These methods are widely used because they can be really fast, even if they may lead to inaccurate results due to the strong assumptions they introduce.

In [55], an MCMC solution, based on Gibbs sampling, is provided. The idea behind it is to exploit a Markov chain having the true posterior distribution as target distribution. In order to that, each new sample of the chain is extracted after a cyclic process. In practice, the latent factors are divided into different groups (in this case two groups, parameters and hyperparameters). For each group of variables a new value is drawn from the related distribution, conditioned on the current values of all the others, until all the groups are updated. In order to do that, the fully conditional distribution must be of course easy to sample from. For this reason, the choice of the priors is critical, and it's very convenient to use conjugate distributions, as decided in [55].

After a burn-in phase, the algorithm converges to the posterior distribution of the unknown variables. A sufficiently large number, B, of samples is kept to compute the empiric distribution:

$$p(\hat{R}_{ij}|R) \approx \frac{1}{B}\sum_{b=1}^{B} p(\hat{R}_{ij}|U_i^{(b)}, V_j^{(b)}) \tag{22}$$

The result is that, after the learning phase, for each pair a predictive distribution can be computed, exploiting the related latent factors. In this way it is possible to fill the gaps corresponding to the unknown entries of the original matrix.

2.3. Tensor factorization

The term tensor is used to indicate a multidimensional array [58]. Each tensor is characterized by an order, i.e. the number of its dimensions. For example, a vector is a first-order tensor, a matrix a second-order tensor and tensors of order three or higher are called higher-order tensors. The dimensions of a tensor are also known as modes or ways. From a mathematical point of view, a Nth-order tensor is the result of the tensor product of N vector spaces, each of them characterized by its own coordinate system. Figure 2.3 shows an example of the simplest higher-order tensors, the third-order ones, which can be represented with a cube.

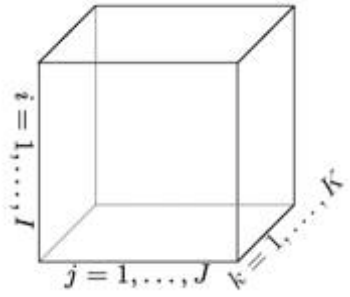

Figure 2.3: example of a third-order tensor. Figure from [58]

There exists also a term to express the high order analogue of matrix rows and columns. This term is fiber: a fiber is the subset of values obtained by fixing all the dimensions but one. Figure 2.4 shows the three types of fiber in a third-order tensor.

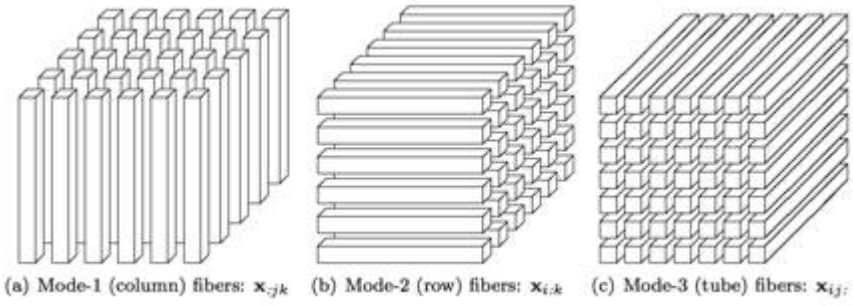

Figure 2.4: types of fiber in a third-order tensor. Figure from [58]

From a practical point of view, tensors are a useful tool to represent data characterized by multiple dimensions. For example, in the biomedical fields, one dimension can be related to the patients, one to the diseases and one to the treatments. In this case, each point in this three dimensional coordinate system may represent if a certain patient has a certain disease and if he/she is treated with a certain therapy. This concept can of course be extended to more complex situations, in which many other variables are involved. Decompositions for tensors have been studied and applied to many fields, from signal processing [59–61] to data mining [62,63], to graph analysis [64], to neuroscience [65,66]. Many of the most common decompositions can be considered as extension of the previously described techniques for matrix factorization like SVD and PCA. In the same way, their objectives often coincide: these decompositions allow pointing out potential latent structures hidden in the data, through procedures that perform a dimensionality reduction. In the following sections, the two most common types of tensor decomposition will be briefly described. Before that, however, some basic concepts must be introduced.

First of all, a Nth-order tensor is defined as a rank-one tensor if it can be written as the outer product of N vectors:

$$\mathcal{X} = a^{(1)} \otimes a^{(2)} \otimes \cdots \otimes a^{(N)} \tag{23}$$

Equivalently, it means that each entry of the tensor can be computed as:

$$x_{i_1, i_2 \cdots i_N} = a_{i_1}^{(1)} a_{i_2}^{(2)} \cdots a_{i_N}^{(N)} \tag{24}$$

Figure 2.5 shows a graphical representation of a rank-one third-order tensor.

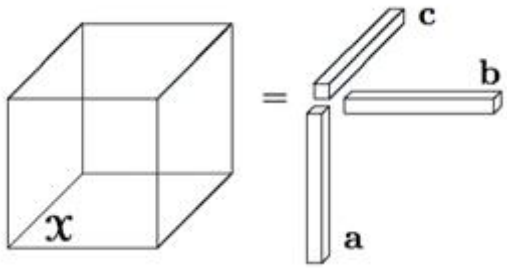

Figure 2.5: example of a rank-one third-order tensor, resulting from the outer product of vectors a, b and c. Figure from [58]

Many mathematical operations can be defined for tensors. One in particular must be introduced to understand the following sections. This operation is the tensor n-mode product. It is defined as the multiplication of a tensor by a matrix or a vector in mode n (i.e. along the n-th dimension). Given a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and a matrix $U \in \mathbb{R}^{J \times I_n}$ the result of the related n-mode product (in symbols $\mathcal{X} \times_n U$) is a tensor with size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$, whose generic entry is given by:

$$(\mathcal{X} \times_n U)_{i_1 \times \cdots \times i_{n-1} \times j \times i_{n+1} \times \cdots \times i_N} = \sum_{i_n=1}^{I_N} x_{i_1 i_2 \cdots i_N} u_{j i_n} \tag{25}$$

To express it in a more intuitive way, this operation describes the multiplication of each mode-n fiber by the matrix U.

After the introduction of these basic concepts, next sections will be focused on the two main factorization developed for the tensors.

2.3.1. CP decomposition

The CANDECOMP/PARAFAC decomposition (canonical decomposition [67]/parallel factors [68]) operates the factorization of a tensor into a sum of terms represented by rank-one tensors:

$$\mathcal{X} \approx \sum_{r=1}^{R} \lambda_r \, a_r^{(1)} \otimes a_r^{(2)} \otimes \cdots \otimes a_r^{(N)} \tag{26}$$

where R is a positive integer, indicating the number of factors used for the decomposition.

Each entry can be therefore expressed as:

$$x_{i_1,i_2,\ldots,i_N} \approx \sum_{r=1}^{R} \lambda_r \, a_{i_1 r}^{(1)} a_{i_2 r}^{(2)} \cdots a_{i_N r}^{(N)} \tag{27}$$

An example of CP decomposition of a third-order tensor is depicted in Figure 2.6.

Indicating with $A^{(i)} = [a_1^{(i)} \; a_2^{(i)} \; \ldots \; a_r^{(i)}]$, an alternative notation is:

$$\mathcal{X} \approx [\![\lambda; A^{(1)}, A^{(2)}, \ldots, A^{(N)}]\!] \tag{28}$$

These $A^{(1)}, A^{(2)}, \ldots, A^{(N)}$ matrices are commonly called factor matrices.

As can be easily noticed, the sum of the factors is weighted using the λ_r coefficients. They are often obtained by normalizing to length one the columns of the factor matrices.

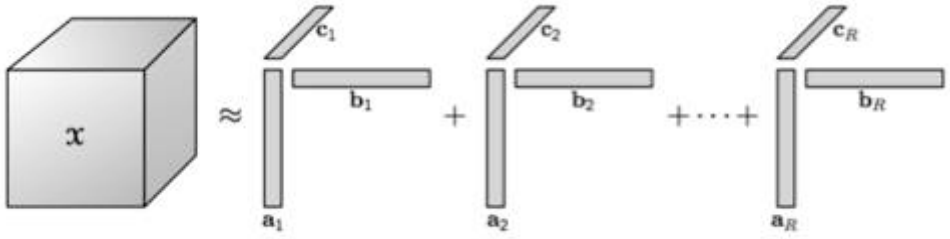

Figure 2.6: Representation of CP decomposition of a third-order tensor using R rank-one factors. In this example, λ=1 for each r. Figure from [58]

The CP decomposition is strongly related to the rank of a tensor. By definition, the rank of a tensor $\mathcal{X}$ is the minimum number of rank-one tensors required to generate $\mathcal{X}$ as their sum. Therefore, the rank is the smallest number of factors required to a CP decomposition to reconstruct exactly the original tensor. An exact CP decomposition, using R equals to the rank of $\mathcal{X}$, is called the rank decomposition.

While matrix decompositions, like SVD, are not unique, under very weak conditions rank decomposition leads to the uniqueness of the solution. In particular, there is a sufficient condition guaranteeing the uniqueness:

$$\sum_{n=1}^{N} k_{A^{(n)}} \geq 2R + (N-1) \tag{29}$$

where $k_{A^{(n)}}$ is the rank of $A^{(n)}$.

On the contrary, if for the truncated SVD v factors give the best v-rank decomposition, for CP decomposition the problem is much more complex and that property doesn't hold [69].

Concerning the computation of CP decomposition, there is not an exact solution to the problem. In fact, it's not even possible to compute the rank of a tensor with a straightforward algorithm (it's an NP-hard problem [70]). The solution can be expressed as the minimization the quantity $\|\mathcal{X} - \widehat{\mathcal{X}}\|$, where $\widehat{\mathcal{X}}$ is the approximation obtained by summing R rank-one tensors. A possible strategy to solve this problem is to use an iterative algorithm to find a local minimum. For example, alternating least squares are often used. In this case, each of the factor matrices $A^{(i)}$ is optimized keeping fixed all the others. The algorithm stops when some convergence criterion is met.

2.3.2. Tucker decomposition

The Tucker decomposition can be seen as a higher order principal component analysis. The main idea of this factorization is to express the original tensor as a core tensor, multiplied by a matrix along each mode [71]. In formulas:

$$\mathcal{X} \approx \mathcal{G} \times_1 A^{(1)} \times_2 A^{(2)} \ldots \times_N A^{(N)} = [\![\mathcal{G}; A^{(1)}, A^{(2)}, \ldots, A^{(N)}]\!] \tag{30}$$

$\mathcal{G}$ is the core tensor, and it expresses the interaction between the different components. $A^{(1)}, A^{(2)}, \ldots, A^{(N)}$ are the factor matrices, which are in general orthonormal.

The single elements is thus computed as:

$$x_{i_1 i_2 \cdots i_N} \approx \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \cdots \sum_{r_N=1}^{R_N} g_{r_1 r_2 \cdots r_N} a_{i_1 r_1}^{(1)} a_{i_2 r_2}^{(2)} \cdots a_{i_N r_N}^{(N)} \tag{31}$$

A graphical representation of Tucker decomposition for a third-order tensor is depicted in Figure 2.7.

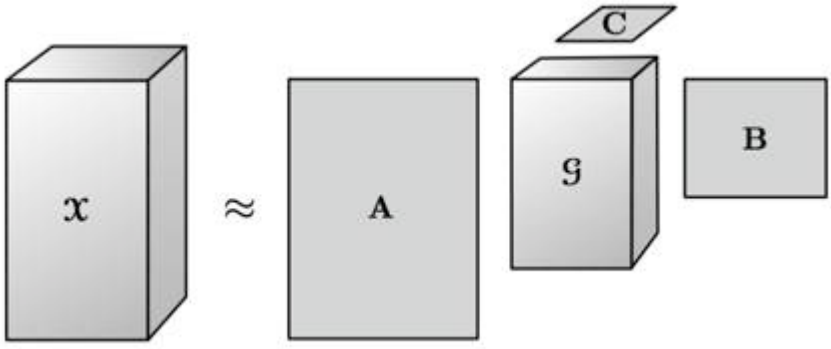

Figure 2.7: Representation of Tucker decomposition of a third-order tensor. Figure from [58]

There is a correlation between the Tucker decomposition and the CP decomposition. In particular, the CP decomposition can be seen as a particular form of Tucker decomposition characterized by a diagonal core tensor (i.e. a tensor for which an entry $x_{i_1 i_2 \cdots i_N} \neq 0$ only if $i_1 = i_2 = \cdots = i_N$). The elements of main diagonal in this case correspond to the λ_r coefficients. On the contrary of CP decomposition the Tucker decomposition is not unique.

The dimensions of the core tensor are typically chosen to be much smaller with respect to the dimensions of the original tensor. In this case, it can be interpreted as a dense and compressed representation of the original information. The factor matrices, instead, can be interpreted as the principal components in each mode. Using this strategy it is therefore possible to greatly reduce the amount of space needed to store the data [72].

Regarding the computation of the Tucker decomposition, some algorithms have been proposed [71,73]. In general they focus only on third-order tensors, which are a simple case but it's also the most used in practice. Anyway, the common idea of most of the algorithms is to exploit alternating least squares to find a local minimum of a cost function measuring the reconstruction error. After choosing appropriately the dimensions of the core tensor, the factors matrices are initialized. Different methods differentiate based on the initialization of the factor matrices, which represents a critical aspect. After that, an iterative algorithm is used: each of the factor matrices is updated keeping all the other fixed, and at the end of this step the core tensor is optimized.

Chapter 3

Matrix factorization for data mining

This chapter is focused on the application of the matrix factorization approach to the data mining field. As highlighted in the previous chapter, many mathematical properties characterizing this type of operations can be effectively applied for multiple purposes.

In the following, the topic of recommender systems will be introduced. It is currently a pretty hot area of interest, due to the incredible growth of online tools made available in the past few years by different type of business companies. It will be shown how in this field it is possible to exploit some of the characteristics of matrix decompositions in order to build efficient algorithms for recommendations prediction.

Afterwards, a more traditional machine learning problem will be discussed. In particular, a method employing a matrix factorization to perform both classification and regression will be described.

3.1. Matrix factorization for recommender systems

In these years, many companies have put great effort into trying to model people's individual tastes, with the final objective of suggesting new products they would probably like. This philosophy is on the basis of the so-called recommender systems [74–77]. The main idea is to exploit all the available information about a specific user in order to customize the offer towards him/her. Given a catalogue of available products, a good recommendation system is able to perform a ranking of these items, specifically calibrated for a certain user.

A great boost to the research in this area was given by the Netflix prize [78]: it was a competition, sponsored by Netflix from 2006 to 2009 with a great cash prize (one million dollars), aimed at promoting the development

of a new, much more accurate, recommender system for its huge database, consisting of millions of movie ratings. That's the reason why many of the algorithms published in recent years use the Netflix dataset as a benchmark to evaluate their performance.

The idea behind these systems is really general, therefore similar strategies have been proposed in many fields. For example, they are used for e-commerce (e.g. by Amazon), to suggest new movies and TV series to watch (e.g. by Netflix) or music to listen to (e.g. by Last.fm and Pandora Radio), but they are also employed by the social media, to identify similar users or even for papers research [79]. For example, some social networks like Facebook, LinkedIn and MySpace use recommender systems to suggest new friends and groups, often relying on the concept of trust between users [77,80]. As far as it concerns the biomedical area, more sophisticated and ambitious applications have been proposed, including the disease risk diagnosis prediction [81] and connecting patients with similar conditions [82].

Different strategies have been explored with the aim of solving this problem, depending on the available information and the way it is represented [83]. Figure 3.1 synthetizes the main categories of recommender systems. In particular, two great classes of algorithms are commonly employed in this context [84]:

- Content-based methods. The strategy adopted by these algorithms is to try to model each object using a certain number of discrete characteristics. In practice, for each available object, (item or user), a profile is created, trying to capture as much information as possible about its nature.
  Typically, the first step of these techniques is to find a proper description for each item. This is achieved by identifying a suitable set of features characterizing the related object in details, which will be of course dependent on the particular domain.
  After that, the user profiles are created, based on both explicit feedbacks, like the rating of past choices, and implicit, like the amount of time spent examining a particular item.
  The different classes of filtering algorithms are based on machine learning techniques that try to use this collected information in order to compute a weight for each feature, expressing the importance of that aspect for the specific user. Other data, like demographic and personal information, may be collected with appropriate techniques (like questionnaires) and integrated in the model. Therefore, the trait distinguishing the content-based filters is that they the recommendation for the users are independent from each other, since they are based just on the characteristics (i.e. the content) of the items.

- Collaborative filtering. For this type of algorithms, the concept of object similarity is the key to predict future custom interests. The idea behind this approach is quite innovative in the field of data mining, and due to its generality it can be easily applied also in contexts different from the recommendation field. For this reason, next section is focused on the description of the way these algorithms work, with particular interest to the techniques based on matrix decomposition.
- Hybrid approach: it is also possible to combine the two strategies described above to exploit their strong points and mitigate their limits. There are different ways to combine the two methods. For example, both of them can be applied, and then their results are combined together. Otherwise, a unique integrate model, exploiting both the strategies can be implemented [85].

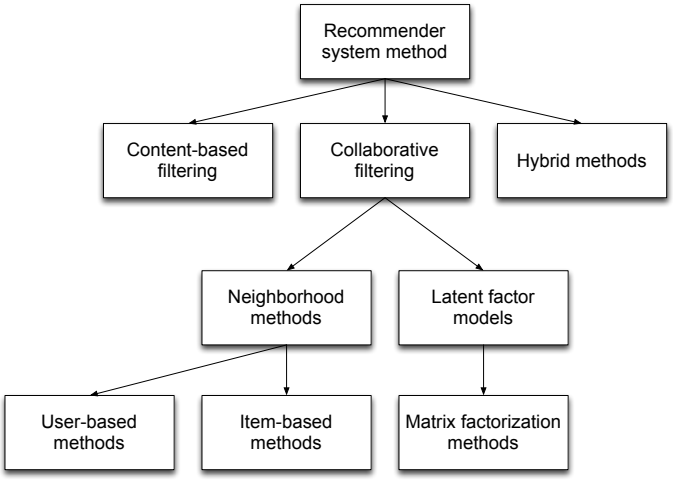

Figure 3.1: representation of the main classes of recommender systems.

3.1.1. Collaborative filtering algorithms

As the name suggests, the collaborative filtering algorithms for recommender systems aim at performing a selection, from a large list of items, exploiting a sort of collaboration between the users. The basic idea is that, if two users have similar tastes, then they would probably like the same items [86]. Therefore, the formulation of the problem revolves around the research of similarities, between both users and items, and how they are associated to each other. The key information in this case is represented by the past choices and the ratings of a user: for example, which products he/she bought and how he/she evaluated them. Logically, similar users will be characterized by similar ratings of the same objects. Thus, the algorithm will suggest to a specific customer all the items characterized by high ratings by similar users. From a dual point of view, if two products are typically bought together, it means that they are similar. Therefore, if a user buys one of them it is also probable he/she will buy also this other.

On the contrary of the content-based filtering, in this case a specific profile for each object is not required, because it is actually inferred directly from the data. This means that every item can be included in the catalogue, even if coming from a completely different context, because no specific domain knowledge is required. The algorithm itself has the capability of capturing the latent features hidden in the data, without any external information. The collaborative filtering approach is clearly much more general than content-based filtering, and it is easily extendible to different fields, since it does not require creating explicit objects' profiles using predefined features. This is particularly relevant when the definition of an item's characteristics is not straightforward. For example, while for texts there exist some techniques based on word frequencies, for other types of objects like music and movies, this procedure can be challenging [84].

Of course, there are also some drawbacks. The main critical point of the collaborative filtering algorithms is represented by the well known *cold start* problem [87]. It is the typical situation that occurs when a new user is added to model. In fact, in this case little information is available about his/her tastes, since number of rated items is negligible, and of course the algorithm is not able to identify similar users in an accurate way. Same thing happens when a new item is added to the catalogue. So, basically, a large amount of information is needed to make accurate recommendations.

Another issue related to collaborative filters is the dimension of the starting dataset. If thought as a traditional database, with a record for each user and a column for each item, it would be characterized by huge dimensions, even if, typically, most of the entries would be zeros, because each user in general rates just a very small part of the overall set of items. This represents a problem both from a storage point of view, since it is not an efficient way to structure the data, and from a computational point of view, since massive computational resources are needed to process the data.

Even if all based on the same basic idea, many different collaborative filters have been proposed in the past few years. In particular, two main classes can be distinguished: the neighborhood methods and the latent factor models.

3.1.1.1. Neighborhood methods

The core point of the neighborhood methods is the direct evaluation of the similarity between objects. Many similarity measures can be used, from a simple Pearson correlation coefficient to more advance procedures [88]. Two different approaches are typically implemented:

- User-user approach: in this case the term neighbors indicates the similar users, i.e. users with similar choices and similar ratings.
- Item-item approach: in this case the neighbors are similar items. If a user liked a certain item in the past, he/she will probability like a similar one in the future. A famous example this type of item-item similarity algorithm was implemented for example by Amazon [89].

3.1.1.2. Latent factor models

On the opposite side, the other main category of methods, the latent factor models, tries to express the relationships between objects by identifying hidden characteristics of the data. This operation is achieved by extrapolating, directly from the data without an external action, a set of latent factors. They may represent clear properties of an object, for example, given a movie, the latent factors may roughly represent its genre, while for a user they may be related to the age, the gender or the nature. Sometimes instead, these factors are not easily interpretable: this is of course a point in favor of the method, since it means that it is able to capture also non-trivial characteristics of the data. Under this perspective, the latent factors can be seen as the axes of a new feature space: they operate a projection of the data into a low dimensional space, condensing the information in a reduced number of meta-features. The operation is performed for both the users and the items. Consequently, to link the two features spaces, the algorithm has to learn the associations between the different latent factors. The intuitive interpretation of this strategy is the following: a user's ratings will move his/her position inside the users space, accordingly to the value of the latent factors, and therefore also the related position in the items space to investigate will change [90].

The nature of latent factors model make the problem eligible to be addressed using matrix decompositions. The system data, consisting of the ratings of all users, can be easily modeled as a very sparse big matrix $R \in \mathbb{R}^{nxm}$, with n number of users and m number of items. In their basic formulation, collaborative filters based on matrix factorization map both users space and items space in a low dimensional space. This dimension is a critical parameter of the method: a too small value is not able to capture all the information in the data, a too high value doesn't allow to generalize and thus discover the presence of latent variables. The behavior of each user u will be summarized in a vector $p_u \in R^f$, while to each item i a vector $q_i \in R^f$ will be associated. The elements of these vectors represent the relative importance of the associated latent factor for that object. For example, for the users, the latent factors can be interpreted as the distinct categories of people using the system, and each value of p_u indicates how much the user u can be considered as member of that category. The same reasoning is valid for items. A simple estimate of the interest of user u for

item i, $\hat{r}_{ui}$, can be obtained by the inner product the two related low dimensional vectors:

$$\hat{r}_{ui} = q_i^T p_u \tag{32}$$

Figure 3.2 depicts a representation of this basic recommender system.

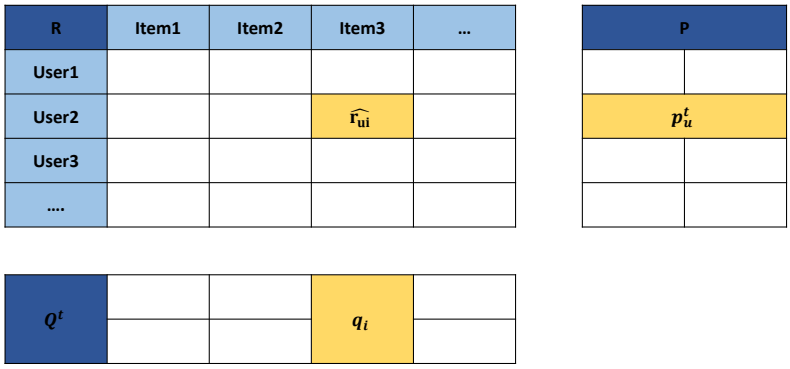

Figure 3.2: general schema of a recommender system based on matrix factorization.

As discussed in Chapter 2, there are many ways to compute these vectors. From a conceptual point of view, the learning phase should perform the minimization of the reconstruction error, measured on the known ratings. Since this operation by itself may lead to overfit the original data, regularization parameters may be added to the cost function to optimize. An example is given in [90]:

$$\min_{q_\cdot, p_\cdot} \sum_{(u,i)\epsilon K} (r_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2) \tag{33}$$

where K indicates the set of user-item pairs (u,i) for which the associated rating r_{ui} is known. The λ parameter is used for regularization, penalizing the squared norms of the two vectors.

The model described above is just a basic version of a collaborative filtering algorithm. In fact, this representation assumes homogeneity among the different users and items. However, very often there is an imbalance on the values of the ratings. For example, some items may have a higher global evaluation (e.g. movies with general acclaim) and in the same way, the users can be more or less hard to satisfy. For this reason, it is worthwhile to include these biases in the algorithm. The factorization model will be therefore used just to express the interaction aspects, while all the other effects will be condensed in a bias term. This, for example, may have this form:

$$b_{ui} = \mu + b_i + b_u \tag{34}$$

where μ indicates the global mean of all the ratings, while b_i and b_u denote the item bias term and the user bias term, respectively. In practice, b_i and b_u indicate how much a specific item's ratings and a specific user's rating commonly differ from the global mean. So, on the contrary of the latent factors parameters, these terms have a clear direct interpretation. The final recommendation for a certain pair (u,i) will be thus computed as [91]:

$$\hat{r}_{ui} = b_{ui} + q_i^T p_u \tag{35}$$

The new cost function will be modified consequently to take in account the bias terms:

$$\min_{q\cdot,p\cdot,b\cdot} \sum_{(u,i)\epsilon \mathrm{K}} (r_{ui} - b_{ui} - q_i^T p_u)^2 + \lambda(\|q_i\|^2 + \|p_u\|^2 + b_u^2 + b_i^2) \tag{36}$$

Despite these improvements, some weaknesses are still present in the model. In particular, to address the *cold start* problem additional modifications have to be introduced.

A viable direction is to include further information about the users. For example it is possible to exploit some personal data such as gender, age, geographic position… To each of these attributes another low-dimensional vector $y_a \epsilon R^f$, is associated. Through the dot product of the vector q_i^T by the vector y_a, the relationship between the item i and the property a will be measured. The procedure is repeated for each attribute, so that the overall effect is determined by the sum of these products.

Another possibility, implemented by algorithms such as the SVD++ [88], is to exploit the implicit feedbacks: as mentioned above, this expression is used to indicate all the actions that can't be considered actual ratings, for example the simple navigation history. A second item vector, $x_z \epsilon R^f$, can be introduced in the model, to express the implicit feedback of a certain user for the item z. Again, the overall effect of these feedbacks can be determined by summing up all the inner products between q_i^T and x_z.

A possible model that integrates both the solution is the following:

$$\hat{r}_{ui} = b_{ui} + q_i^T [p_u + \sum_{a\epsilon \mathrm{A(u)}} y_a + |N(u)|^{-0.5} \sum_{z\epsilon \mathrm{N(u)}} x_z] \tag{37}$$

where A(u) is the set of personal attributes available for user u, N(u) is the set of items for which for the same user provided an implicit feedback, and $|N(u)|^{-0.5}$ is a normalizing term to balance the fact that the number of implicit feedbacks varies from user to user.

A similar operation can be also extended to the items representation.

Further modifications can be introduced in the model in order to improve the performance.

In particular, another important aspect to take into account is the temporal dynamics of the ratings [92]. For example, the popularity of a movie may greatly vary over time and therefore sometimes it's convenient to model this evolution by introducing a temporal dependence on the associated low dimensional vector $b_i(t)$. In the same way, the user's characteristics, both related to the bias term and to the explicit feedbacks, may vary as well, thus making worthwhile to introduce a temporal dependence also for $b_u(t)$ and $p_u(t)$.

3.2. Application to machine learning

From a general point of view, the expression machine learning refers to a collection of techniques able to exploit a set on data in order to make predictions. In a traditional machine learning scenario, the starting dataset is composed of a set of examples, and for each of them a certain number of features are measured. This information is used to train an algorithm, which can vary a lot based on the purpose of the application. In case of supervised learning, the objective is to assign to a new, unseen, sample, a certain value. It can be a label, associating a class to the object (classification problem), or a numerical continuous value (regression problem). In order to perform this operation, the algorithm has to be trained on a proper training set. This must be composed of complete data, meaning that, for each example, also the value of the class or the quantity of interest must be known. After the learning phase, the algorithm is able to predict the quantity of interest for a new sample, using the values of the other measured features.

Within the traditional machine learning framework, matrix factorization techniques may be successfully employed, thanks to their ability of capturing the interaction effects among the variables. To highlight this aspect, in the next section, an integrated approach, called factorization machines, will be described.

3.2.1. Factorization machines

Factorization machines (FMs) [93,94] are a machine learning algorithm for classification and regression. Since it is a supervised method, the learning phase of the model requires a set $\mathrm{S} = \left\{\left(\mathrm{x}^{(1)}, \mathrm{y}^{(1)}\right), \left(\mathrm{x}^{(2)}, \mathrm{y}^{(2)}\right), \ldots\right\}$ of complete data, composed of a vector of measured features, x, and an associated target value, y, for each sample. The objective is to use the algorithm to predict the target value y for new unseen examples.

FMs were specifically developed for recommender systems applications. Figure 3.3 describes a possible input dataset for movies recommendations. In this case, each row represents a different rating, expressed by the value

of y. The set of features, instead, contains different types of information. The blue box highlights the mapping of the users: in each row, just one entry of those variables is set to 1, indicating the user who rated the movie, while all the other entries will be zeros. In the same way, the orange box indicates the variables used for the movies mapping. Other information may be added to improve predictions. For example, the yellow box highlights a set of variables used to represent the previous ratings of each user. Further examples are the time of the rating (green box) and the previously rated movie (brown box). From this simple example, it's immediately clear that the dataset in characterized by high sparseness, since most of the entries are zeros. FMs try to address this problem by relying on a matrix decomposition strategy.

Figure 3.3: example of input dataset for Recommender systems. Each row represents a complete data, including the feature vector x and the associated target value y. Figure from [93]

From the mathematical point of view, the model is described by the following equation:

$$\hat{y}(x) = w_0 + \sum_{i=1}^{p} w_i x_i + \sum_{i=1}^{p} \sum_{j=1}^{p} \langle v_i, v_j \rangle x_i x_j \tag{38}$$

where:

- x is the feature value of the considered example
- $\hat{y}(x)$ is the predicted target value
- p is the number of elements of x (i.e. the number of features)
- $\langle .,. \rangle$ indicates the inner product between two vectors with the same dimension.
- $w_0 \in R, w \in R^p, v_i \in R^k$ are the parameters of the model.

It's straightforward to notice a clear association with the well-known linear regression model, at least for what concerns the first two terms of the equation. For this reason, the w_0 and w parameters are associated the same meaning: w_0 represents a global bias term of the model, while w contains the linear weights of each feature, which takes into account the importance of the variable for the estimate of target value.

The last term, instead, represents the key characteristic of the method. For each pair of feature, a quantity is computed and added to the global summation. This value is equals to the product of the related variables, weighted by the dot product of two vectors: $\widehat{w}_{i,j} = \langle v_i, v_j \rangle$. As easily deducible, $\widehat{w}_{i,j}$ models the effect of the interaction between the i-th and the j-th feature. The v_i vectors are all characterized by the same dimension, k, and play the same role of latent factors in the collaborative filtering models described above. In practice, they try to summarize the information associated to a certain object by mapping it to the low dimensional space defined by the latent factors. The inner product between them measures the strength of the interaction between the two variables.

Under this perspective, there is an association with another type of regression: the polynomial regression. This one is defined by the following equation:

$$y(x) = w_0 + \sum_{i=1}^{p} w_i x_i + \sum_{i=1}^{p} \sum_{j \geq 1}^{p} W_{i,j} x_i x_j \tag{39}$$

As easily noticeable, the $W \in R^{p \times p}$ matrix is the only difference with respect to the FMs model. In the regression case, for each distinct pair a different parameter must be estimated. In the FM case, instead, the corresponding term is computed by the inner product of two vectors v_i and v_j. The low dimensional vectors can be condensed in a single matrix $V \in R^{pxk}$, where each row is related to a specific feature. Provided to choose a small value of k, in particular $k \ll p$, this strategy allows to greatly reduce the number of parameters to estimate to model the interaction effects, passing from $p \times p$ to $p \times k$. The corresponding W matrix can be anyway computed as the product $W = VV^T$.

It is clear that a key role is played by the value of k: a small value is not enough to model all the data, while a too large value reduces the advantages of using this approach.

3.2.1.1. FM Properties

There are several strong points characterizing this method:

- First of all, as stated above, reducing the number of parameters to estimate leads to more robust models, less prone to overfitting. In

particular, since all the interaction of a specific variable i share the same v_i , it means that the estimates are not independent. Equivalently, all the interactions of a specific feature depend on the same low dimensional vector, whose estimate relies also on all the other interactions of the same feature. For each variable then, each interaction helps the estimate of all the others, making the process more robust. This property is particularly critical in case of very sparse datasets. In fact, if just few entries are observed, an independent estimate of all the pairwise interactions can be challenging. It can even be impossible, if a specific combination is never observed in the training data.

- From the computational point of view, another property of the method is the linear complexity. In general, if the objective is to estimate interaction of n variables, the computational cost should be $O(n^2)$. For FM instead, thanks to the factorization operated by the algorithm, the computational complexity is $O(kn)$. To highlight this aspect, it's enough to focus in the term of FM equation characterized by a quadratic complexity, $\sum_{i=1}^{p}\sum_{i=1}^{p}\langle v_i, v_j\rangle x_i x_j$. By reformulating this term:

$$\begin{gathered}
\frac{1}{2}\sum_{i=1}^{p}\sum_{j=i+1}^{p}\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle x_i x_j - \frac{1}{2}\sum_{j=1}^{p}\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle x_i x_j = \\
\frac{1}{2}\left(\sum_{j=1}^{p}\sum_{j\prime=1}^{p}\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle x_i x_j - \sum_{j=1}^{p}\langle \boldsymbol{v}_i, \boldsymbol{v}_j\rangle x_i x_j\right) = \\
\frac{1}{2}\left(\sum_{i=1}^{p}\sum_{j=1}^{p}\sum_{f=1}^{k} v_{i,f} v_{j,f}\, x_i x_j - \sum_{j=1}^{p}\sum_{f=1}^{k} v_{i,f} v_{i,f}\, x_j x_j\right) = \\
\frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{p} v_{i,f} x_i\right)\left(\sum_{j=1}^{p} v_{j,f}\, x_j\right) - \sum_{i=1}^{p} v_{i,f}^2\, x_i^2\right) = \\
\frac{1}{2}\sum_{f=1}^{k}\left(\left(\sum_{i=1}^{p} v_{i,f} x_i\right)^2 - \sum_{i=1}^{p} v_{i,f}^2\, x_i^2\right)
\end{gathered} \tag{40}$$

- This shows how the complexity is actually proportional to $k \times p$, and so linearly proportional to the number of samples in the dataset. Of course this is fundamental when large datasets are analyzed.

- A key aspect is represented by the interpretability of the model. In fact, starting from the matrix V it is possible to obtain the complete interaction matrix with a simple operation: $W = VV^T$.

Another property characterizing the FMs is the multilinearity. Indicating with $\Theta = \{w_0, w_1, \dots, w_l, \dots, w_p, v_{1,1}, \dots, v_{l,f}, \dots, v_{p,k}\}$ all the parameters of the model, for each $\theta \in \Theta$ it is possible to express the basic equation as:

$$\hat{y}(\boldsymbol{x}) = g_\theta(\boldsymbol{x}) + \theta h_\theta(\boldsymbol{x}) \tag{41}$$

This means that the predicted value can be seen as a linear combination of two functions, g_θ and h_θ, which depend only on $\mathbf{x}$, while they are independent from the value of θ. In particular, h_θ represents the gradient of the FM model, computed as follows:

$$h_\theta(\boldsymbol{x}) = \frac{\partial \hat{y}(\boldsymbol{x})}{\partial \theta} = \begin{cases} 1, & if\ \theta\ is\ w_0 \\ x_i, & if\ \theta\ is\ w_i \\ \left(x_i \sum_{j=1}^{n} v_{j,f} x_j \right) - v_{i,f} x_i^2, & if\ \theta\ is\ v_{i,f} \end{cases} \tag{42}$$

Since the summation $\sum_{j=1}^{n} v_{j,f} x_j$ is independent from i, it can be computed separately. In general, each gradient can be computed in a $O(1)$.

The other function, g_θ, can be instead computed as $g_\theta(\boldsymbol{x}) = \hat{y}(\boldsymbol{x}) - \theta h_\theta(\boldsymbol{x})$. This subdivision will be useful to explain some of the learning algorithms developed for the optimization procedure.

The presented model is just the simple form of FM, since only pairwise interactions are considered. Anyway, it is easily extendible to a d-way model, considering higher order interactions. In this case the model equation becomes:

$$\hat{y}(x) = w_0 + \sum_{i=1}^{n} w_i x_i + + \sum_{l=2}^{d} \sum_{i_1=1}^{n} \dots \sum_{i_l=i_{l-1}+1}^{n} \left(\prod_{j=1}^{l} x\_ij \right) \left(\sum_{f=1}^{k_l} \prod_{j=1}^{l} v_{i_j,f}^{(l)} \right) \tag{43}$$

where d indicates the maximum order of interactions included in the model. The d-order interaction term is obtained by multiplying the values of the related d features, weighted by a quantity computed as a function of their low dimensional vectors $v_{i_j,f}^{(l)}$. A tensor factorization method such as the PARAFAC decomposition can be used to determine the interaction parameters in this context.

3.2.1.2. Comparison with other methods

In addition to the polynomial regression, FMs have similarities with other algorithms, both for machine learning and factorization-based recommender systems.

For example, Support Vector Machines (SVM) are a supervised classification algorithm very popular in the machine learning field [95]. Some similarities with FM will be briefly highlighted. The equation of SVM can be expressed as:

$$\hat{y}(x) = \langle \phi(x), w \rangle \tag{44}$$

In practice, the target value is computed as the dot product of the model parameters w with the function $\phi(x)$: this one maps the input data from the features space to a space where the classification problem is easier to address. Different kernel functions, defined as $K(x,z) = \langle \phi(x), \phi(z) \rangle$, discriminates among different classes of SVMs.

The simplest kernel function is the linear one:

$$K(x,z) = 1 + \langle x, z \rangle \tag{45}$$

The equation of the linear SVM is this case becomes:

$$\hat{y}(x) = w_0 + \sum_{i=1}^{p} w_i x_i \tag{46}$$

This is of course a linear regression, and no interactions are considered. If instead a polynomial kernel is used, it is possible for example to include in the model the interaction terms. Given the following polynomial kernel function:

$$K(x,z) = (\langle x, z \rangle + 1)^2 \tag{47}$$

the associated model equation is:

$$\hat{y}(x) = w_0 + \sqrt{2}\sum_{i=1}^{p} w_i x_i + \sum_{i=1}^{p} w_{i,i}^{(2)} {x_i}^2 + \sqrt{2}\sum_{i=1}^{p}\sum_{j=i+1}^{p} w_{i,j}^2 x_i x_j \tag{48}$$

This model is able to take into account all the pairwise interactions, but on the contrary of FM all the related parameters are independent, with the same overfitting problem of polynomial regression.

FMs have also analogies with other factorization models, for example with SVD++. This is immediately clear when considering a typical recommender system dataset. For example, each row may relate a certain

user $u \in U$, with the item $i \in I$, for which the algorithm wants to estimate the user's interest, and a set of objects $N(u) = \{l_1, l_2, \dots, l_m\} \in L$ for which user u showed interest about. Therefore, the values of vector x will be:

$$x_j = \begin{cases} 1, & if\ j = u \vee j = i \\ \frac{1}{\sqrt{m}}, & if\ j \in N(u) \\ 0, & else \end{cases} \tag{49}$$

The FM model in this case becomes:

$$\begin{aligned} \hat{y}(x) = w_0 + w_u + w_i + \langle v_u, v_i \rangle + \frac{1}{\sqrt{m}} \sum_{j=1}^{m} \langle v_i, v_{l_j} \rangle \\ + \frac{1}{\sqrt{m}} \sum_{j=1}^{m} w_{l_j} + \frac{1}{\sqrt{m}} \sum_{j=1}^{m} \langle v_u, v_{l_j} \rangle + \frac{1}{m} \sum_{j=1}^{m} \sum_{j'>j}^{m} \langle v_{l_j}, v_{l_{j'}} \rangle \end{aligned} \tag{50}$$

The first part is similar to the SVD++ model, containing the global bias effect, the user and the item effect and some interaction terms between the user u and the item i, and between the item i and all the objects in $N(u)$.

The second part, instead, differs from the SVD++ model because it contains additional interactions between user u and the objects in $N(u)$, along with the bias term and the interaction terms for the objects in $N(u)$.

3.2.1.3. FM Parameters Learning

As most of the matrix factorization techniques, the parameters estimation for FMs relies on the optimization of a cost function. Given a set S of complete data for the training, the goal is to find the best set of Θ that minimizes the prediction error, i.e. the difference between what the model predicts, $\hat{y}(\boldsymbol{x}|\Theta)$, and the real target value y. This error is measured using a loss function, $l(\hat{y}(\boldsymbol{x}|\Theta), y)$, which may vary depending on the algorithm. In general the learning phase can be therefore summarized in:

$$Opt(S) = \underset{\Theta}{\operatorname{argmin}} \sum_{(\boldsymbol{x},y) \in S} l(\hat{y}(\boldsymbol{x}|\Theta), y) \tag{51}$$

If a regression problem is addressed, a simple choice for the loss function is the least squares loss function:

$$l^{LS}(\hat{y}(\boldsymbol{x}|\Theta), y) := (\hat{y}(\boldsymbol{x}|\Theta) - y)^2 \tag{52}$$

It simply measures the squared error of the difference between the real and the predicted value.

For a binary classification problem instead, it is possible to use a logistic loss function:

$$l^C(\hat{y}(\boldsymbol{x}|\Theta), y) := -\ln \sigma(\hat{y}(\boldsymbol{x}|\Theta)y) \tag{53}$$

where $\sigma(\mathrm{x}) = \frac{1}{1+e^{-x}}$ is the logistic function.

As stated above, the choice of the parameter k is fundamental to obtain good performance. In particular, when k is large, the risk is to make the model overfit, i.e. it tries to adapt exactly to the training data, learning also the associated noise, and therefore it is not able to generalize well.

A common approach to avoid this condition is to introduce in the model some regularization parameters. For example, in case of L2 regularization, the cost function is modified in this sense:

$$OptReg(S) = \underset{\Theta}{\operatorname{argmin}} \left(\sum_{(\boldsymbol{x},y)\in S} l(\hat{y}(\boldsymbol{x}|\Theta), y) + \sum_{\theta\in\Theta} \lambda_\theta \theta^2 \right) \tag{54}$$

where $\lambda_\theta \in R^+$ is dependent on the specific parameter, meaning that each parameter can have a different regularization. In this way the cost function is penalized if the modules of the parameters are too high.

The FM model can be also represented in a probabilistic fashion [54], leaning on the idea of probabilistic matrix factorization introduced in the previous chapter. In this case, proper distribution can be used to model the error functions. A comprehensive representation of the probabilistic model is shown in Figure 3.4.

For what it concerns the regression problem, the usage of the least squares loss function corresponds to the assumption that y is Normally distributed, with mean $\hat{y}(\boldsymbol{x}|\Theta)$ and precision α:

$$y|\boldsymbol{x}, \Theta \sim N(\hat{y}(\boldsymbol{x}|\Theta), 1/\alpha) \tag{55}$$

In case of classification instead, y is assumed to follow a Bernoulli distribution:

$$y|\boldsymbol{x}, \Theta \sim Bernoulli\left(b\left(\hat{y}(\boldsymbol{x}|\Theta)\right)\right) \tag{56}$$

where b is a link function, defined from $\mathbb{R}$ to [0,1]. In general a logistic function or the cumulative density function of a standard Gaussian distribution is used.

A probabilistic explanation can also be associated to the regularization process. In particular, in case of L2 regularization, each parameter θ is

assumed to be Gaussian, with a certain mean μ_θ and precision λ_θ (corresponding to the regularization parameter):

$$\theta|\, \mu_\theta, \lambda_\theta \sim N(\mu_\theta, 1/\lambda_\theta) \tag{57}$$

Again, different values of μ_θ and λ_θ can be chosen for each attribute. It can be noticed as, assuming $\alpha = 1$ and $\mu_\theta = 0$, the Maximum A Posteriori (MAP) estimate of this model is equivalent to the optimization process defined by eq.(54).

Since a close solution to the optimization problem does not exist, a heuristic approach must be adopted. Some algorithms have been proposed. In particular, a tool, called libFM [94], contains the implementation in C++ language of three different learning methods: the Stochastic Gradient Descent (SGD), the Alternating Least Squares (ALS) and a Markov Chain Monte Carlo (MCMC) based method.

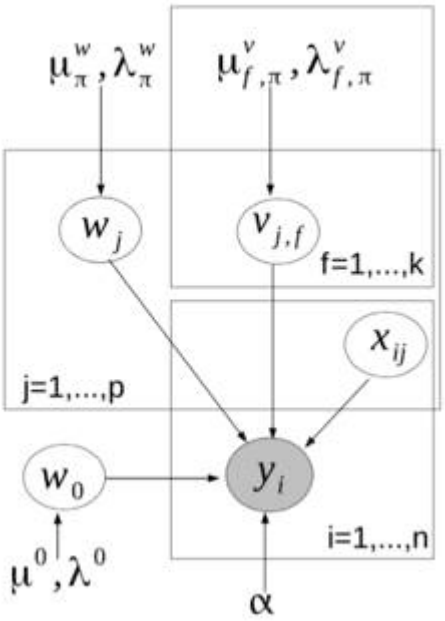

Figure 3.4: probabilistic representation of FM model. Figure from [94]

3.2.1.4. Stochastic Gradient Descent (SGD)

As the name suggests, the SGD methods perform a gradient descent to find the minimum of a function [96]. This is obtained by computing the gradient of the quantity to minimize, which always indicates the direction of maximum growth of the function. By moving in the opposite direction is thus possible to decrease the value of the target quantity. Repeating this operation iteratively, it's possible to reach a local minimum of the cost

function. The first step of the process is therefore to compute the gradients of the loss function for each parameter.

In case of least squares function, the partial derivative for a certain parameter is:

$$\frac{\partial l^{LS}(\hat{y}(\boldsymbol{x}|\Theta), y)}{\partial \theta} = \frac{\partial\,(\hat{y}(\boldsymbol{x}|\Theta) - y)^2}{\partial \theta} = 2(\hat{y}(\boldsymbol{x}|\Theta) - y)\,\frac{\partial}{\partial \theta}\hat{y}(\boldsymbol{x}|\Theta) \tag{58}$$

For binary classification instead, the partial derivative is given by:

$$\begin{aligned} \frac{\partial\, l^{C}(\hat{y}(\boldsymbol{x}|\Theta), y)}{\partial \theta} &= \frac{\partial \left(-\ln\left(\sigma(y\hat{y})\right)\right)^2}{\partial \theta} \\ &= \left(\sigma\left(y \cdot \hat{y}(\boldsymbol{x}|\Theta)\right) - 1\right)\frac{\partial}{\partial \theta}\hat{y}(\boldsymbol{x}|\Theta) \end{aligned} \tag{59}$$

The peculiarity of this version of gradient descent is that it iterates over the single examples $(\boldsymbol{x}, y) \in S$ of the training dataset. For each sample, all the parameters θ of the model are updated, using the derivative of the L2 penalized cost function:

$$\theta \leftarrow \theta - \eta\left(\frac{\partial}{\partial \theta} l(\hat{y}(\boldsymbol{x}|\Theta), y) + 2\lambda_\theta\,\theta\right) \tag{60}$$

This methods requires the definition of three input parameters:

- η, the learning rate. It indicates how wide each step towards the minimum is. This value determines how fast the algorithm goes to convergence. If it is chosen too small, then the convergence can be too slow, if chosen too high, there could be instability problems leading to non convergence.
- λ_θ the regularization terms for each parameter. It is possible to assign different values to different groups of features, based on the available knowledge about them.
- σ^2, a variance term. It is used by the algorithm to initialize the low dimensional vectors of the V matrix. They are in fact drawn from a Normal distribution with zero mean and variance σ^2.

After the initialization phase, an iterative process starts, and for each sample a small step toward the direction of a smaller loss is performed. The process terminates when a stopping criterion, generally based on the quality of the performance, is met. The main steps of the algorithm are reported in Figure 3.5. This class of algorithms is very popular because they are simple to implement, they have a low storage need and, above all, they don't require a lot of computational resources. In particular, for online learning,

this strategy is crucial, since it doesn't require re-estimating the entire model when a new entry is added.

ALGORITHM 1: Stochastic Gradient Descent (SGD)

Input: Training data S, regularization parameters λ, learning rate η, initialization σ
Output: Model parameters $\Theta = (w_0, \mathbf{w}, \mathbf{V})$
$w_0 \leftarrow 0$; $\mathbf{w} \leftarrow (0, \ldots, 0)$; $\mathbf{V} \sim \mathcal{N}(0, \sigma)$;
repeat
  for $(\mathbf{x}, y) \in S$ **do**
    $w_0 \leftarrow w_0 - \eta \left(\frac{\partial}{\partial w_0} l(\hat{y}(\mathbf{x}|\Theta), y) + 2\lambda^0 w_0 \right)$;
    for $i \in \{1, \ldots, p\} \wedge x_i \neq 0$ **do**
      $w_i \leftarrow w_i - \eta \left(\frac{\partial}{\partial w_i} l(\hat{y}(\mathbf{x}|\Theta), y) + 2\lambda^w_{\pi(i)} w_i \right)$;
      for $f \in \{1, \ldots, k\}$ **do**
        $v_{i,f} \leftarrow v_{i,f} - \eta \left(\frac{\partial}{\partial v_{i,f}} l(\hat{y}(\mathbf{x}|\Theta), y) + 2\lambda^v_{f,\pi(i)} v_{i,f} \right)$;
      end
    end
  end
until *stopping criterion is met*;

Figure 3.5: SGD main steps. Figure from [94]

3.2.1.5. Alternating Least Squares (ALS)

The alternating least squares algorithms [97], also called coordinate descent algorithms, is another class of methods that try to solve the minimization problem described above in an iterative way, aimed at finding a local minimum of the cost function. On the contrary of SGD, it considers all the samples at the same time, and for each parameter it computes the optimum value ϑ^*, considering fixed all the others $\Theta/\{\theta\}$. An iterative process is still required since fixing the values of the other parameters it doesn't perform global optimization. In case of least squared loss function and L2 regularization, a close form for this optimum problem can be obtained by exploiting the multilinearity of the FM model [98]:

$$\vartheta^* = argmin_\vartheta \left(\sum_{(\boldsymbol{x},y)\in S} (\hat{y}(\boldsymbol{x}|\Theta) - y)^2 + \sum_{\vartheta\in\boldsymbol{\theta}} \lambda_\vartheta \vartheta^2 \right) = \tag{61}$$

$$argmin_\vartheta \left(\sum_{(\boldsymbol{x},y)\in S} (g_\vartheta(\boldsymbol{x}|\,\Theta/\{\theta\}) + \vartheta h_\vartheta(\boldsymbol{x}|\,\Theta/\{\theta\}) - y)^2 + \sum_{\vartheta\in\boldsymbol{\theta}} \lambda_\vartheta \vartheta^2 \right)$$

and then finding the roots of the derivatives:

$$\vartheta^* = \frac{\vartheta \sum_i {h_\vartheta}^2(x_i) + \sum_i h_\vartheta(x_i)\, e_i}{\sum_i h_\vartheta(x_i)^2 + \lambda_\vartheta} \tag{62}$$

where e_i is the error calculated in the i-th case:

$$e_i = y_i - \hat{y}(\boldsymbol{x}_i|\Theta) \tag{63}$$

The function h_ϑ can be computed as indicated in (42). In case $\theta = v_{l,f}$, the quantity $h_{v_{l,f}}$ can be computed as:

$$h_{v_{l,f}}(\boldsymbol{x}_i) = x_l\left(q_{i,f} - v_{l,f}X_{i,l}\right)$$
$$q_{i,f} = \sum_{l=1}^{p}\left(X_{i,l}v_{l,f}\right) \tag{64}$$

where $Q \in R^{n\times k}$.

Figure 3.6 reports the main step of the ALS algorithm.

Also this method requires some input parameters. In particular, as for SGD, a regularization term, λ_θ , for each parameter and a variance value σ^2 to initialize the low dimensional vectors are required. On the contrary, a learning rate parameter is not necessary, since the method computes the exact optimum minimum at each iteration.

Anyway, in case of classification, it's not possible to use this method because it's not suitable to perform the minimization of the related loss function.

ALGORITHM 2: Alternating least squares (ALS)

Input: Training data S, regularization parameters λ, initialization σ
Output: Model parameters $\Theta = (w_0, \mathbf{w}, \mathbf{V})$
$w_0 \leftarrow 0$; $\mathbf{w} \leftarrow (0, \ldots, 0)$; $\mathbf{V} \sim \mathcal{N}(0, \sigma)$;
repeat
  $\hat{\mathbf{y}} \leftarrow$ predict all cases S;
  $\mathbf{e} \leftarrow \mathbf{y} - \hat{\mathbf{y}}$;
  $w_0 \leftarrow w_0^*$;
  for $l \in \{1, \ldots, p\}$ **do**
    $w_l \leftarrow w_l^*$;
    update e;
  end
  for $f \in \{1, \ldots, k\}$ **do**
    init $q_{\cdot,f}$;
    for $l \in \{1, \ldots, p\}$ **do**
      $v_{l,f} \leftarrow v_{l,f}^*$;
      update e, q;
    end
  end
until *stopping criterion is met*;

Figure 3.6: ALS main steps. Figure from [94]

3.2.1.6. Markov Chain Monte Carlo (MCMC)

Using the probabilistic interpretation of the FMs model, it's possible to exploit some techniques developed in the field of Bayesian inference. Instead of computing point estimates of the parameters, in this case their value is obtained by using some sampling strategies. As previously described, within a probabilistic framework some distributions and related hyperparameters can be introduced in the model. Indicating with Θ_H the set of all the hyperparameters, this is composed of (see Figure 3.2):

$$\Theta_H \coloneqq \{(\mu^0, \lambda^0), (\mu_\pi^w, \lambda_\pi^w), (\mu_{f,\pi}^w, \lambda_{f,\pi}^w) : \forall \pi \in \{1, \dots \Pi\}, \forall f \in \{1, \dots k\}\} \quad (65)$$

The objective of Bayesian methods is to compute for each parameter its posterior distribution. A typical approach in this case is to exploit MCMC techniques. Basically they create a Markov chain (i.e. a sequence of samples, each of them depending only on the previous one) that, after a burn-in phase, converges to the posterior distribution of the parameter. Given a parameter θ, keeping fixed all the other parameters, $\Theta \backslash \{\theta\}$, and the hyperparameters, Θ_H, its posterior distribution has the form of a Gaussian distribution:

$$\theta | X, y, \Theta\{\theta\}, \Theta_H \sim N(\tilde{\mu}_\theta, \tilde{\sigma}_\theta^2) \quad (66)$$

where:

$$\tilde{\sigma}_\theta^2 \coloneqq \left(\alpha \sum_{i=1}^{n} h_\theta(x_i)^2 + \lambda_\theta \right)^{-1} \quad (67)$$

$$\tilde{\mu}_\theta = \tilde{\sigma}_\theta^2 \left(\alpha\theta \sum\nolimits_{i=1}^{n} h_\theta^2(\boldsymbol{x}_i) + \alpha \sum_{i=1}^{n} h_\theta(\boldsymbol{x}_i) e_i + \mu_\theta \lambda_\theta \right) \quad (68)$$

Interestingly to notice, imposing $\alpha = 1$ and $\mu_\cdot = 0$ in this equation, the mean of the posterior distribution is equivalent to the optimum value computed by the ALS algorithm, $\tilde{\mu}_\theta = \theta^*$. Anyway, the assumptions behind the two methods are completely different, since MCMC uses a probability distribution while ALS performs a point estimate.

One of the advantages of the probabilistic interpretation is that the regularization parameters can be directly included in the model. In this way, the algorithm itself can jointly estimate also their value in the learning phase. The basic model must be extended to include hyperpriors distribution over the hyperparameters (see Figure 3.7).

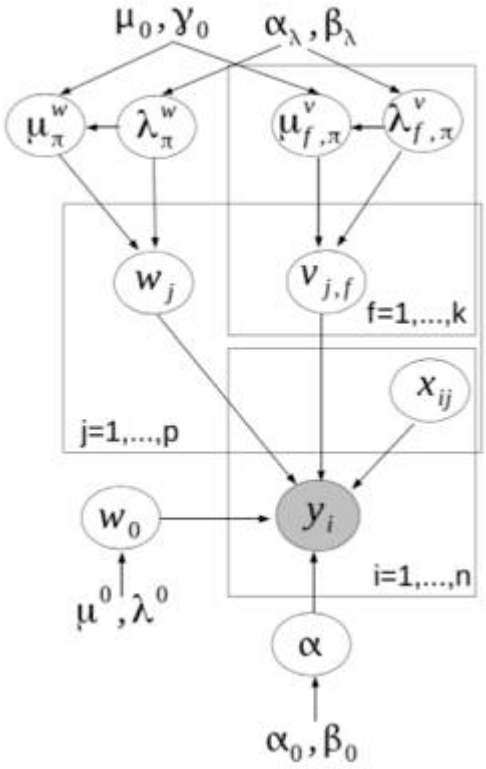

Figure 3.7: probabilistic representation of FM model with hyperpriors. Figure from [94]

Given a parameter θ, the regularization terms λ_θ are Gamma variables, while the μ_θ have a Gaussian probability density distribution:

$$\begin{aligned} &\mu_\pi^w \sim N(\mu_0, \gamma_0 \lambda_\pi^w), \qquad \lambda_\pi^w \sim \Gamma(\alpha_\lambda, \beta_\lambda), \\ &\mu_{f,\pi}^v \sim N\left(\mu_0, \gamma_0 \lambda_{f,\pi}^v\right), \quad \lambda_{f.\pi}^v \sim \Gamma(\alpha_\lambda, \beta_\lambda) \end{aligned} \tag{69}$$

where $\mu_0, \gamma_0, \alpha_\lambda, \beta_\lambda$ are the hyperpriors' parameters.
A Gamma distribution is also used to describe α:

$$\alpha \sim \Gamma(\alpha_0, \beta_0) \tag{70}$$

Within this framework, the values of all hyperparameters Θ_H can be automatically determined by sampling from the related conditional posterior distributions.

For α:

$$\alpha|y, X, \Theta_0, \Theta \sim \Gamma\left(\frac{\alpha_0 + n}{2}, \frac{1}{2}\left[\sum_{i=1}^{n}\left(y_i - \hat{y}(x_i\,|\Theta)\right)^2 + \beta_0\right]\right); \tag{71}$$

For the regularization parameters λ_π:

$$\lambda_\pi|\Theta_0, \Theta_H \setminus \{\lambda_\pi\}, \Theta \sim$$
$$\Gamma\left(\frac{(\alpha_\lambda + p^\pi + 1)}{2}, \frac{1}{2}\left[\sum_{j=1}^{p}\delta(\pi(j) = \pi)\left(\theta_j - \mu_\theta\right)^2 + \gamma_0(\mu_\pi - \mu_0)^2 + \beta_\lambda\right]\right) \tag{72}$$

For the means parameters μ_π:

$$\mu_\pi|\Theta_0, \Theta_H \setminus \{\lambda_\pi\}, \Theta \sim$$
$$N\left((p^\pi + \gamma_0)^{-1}\left[\sum_{j=1}^{p}\delta(\pi(j) = \pi)\theta_j + \gamma_0\mu_0\right], \frac{1}{(p^\pi + \gamma_0)\lambda_\pi}\right) \tag{73}$$

with

$$p^\pi := \sum_{j=1}^{p}\delta(\pi(j)) = \pi) \tag{74}$$

The price to be paid for this model is the definition of the hyperpriors' parameters. Anyway, it seems to be a reasonable cost, because first of all the numbers of these parameters is much smaller than the number of regularization terms. In addition, the MCMC algorithm has demonstrate to be quite insensible to the value of these parameters [94]. The only critical tuning parameter is therefore the variance σ^2 used in the initialization phase, which may affect the convergence speed of the algorithm.

The main steps of the algorithm are reported in Figure 3.8

ALGORITHM 3: Markov Chain Monte Carlo Inference (MCMC)

Input: Training data S, Test data S_{test}, initialization σ
Output: Prediction $\hat{y}_{test}$ for the test cases
$w_0 \leftarrow 0; \boldsymbol{w} \leftarrow (0, \ldots, 0); \boldsymbol{V} \sim N(0, \sigma)$;
$\#_{samples} \leftarrow 0$;
repeat
  $\hat{y} \leftarrow$ predict all cases S;
  $e \leftarrow y - \hat{y}$;
  Update the hyperparameters:
  samples α from its conditional posterior distribution
  for $(\mu_\pi, \lambda_\pi) \in \Theta_H$ **do**
    sample λ_π from its conditional posterior distribution
    sample μ_π from its conditional posterior distribution
  end
  Update the model parameters:
  sample w_0 from $N(\hat{\mu}_{w0}, \hat{\sigma}^2_{w0})$;
  for $l \in \{1,\ldots,\mathrm{p}\}$ **do**
    samples w_l from $N(\hat{\mu}_{wl}, \hat{\sigma}^2_{wl})$;
    update e;
  end
  for $f \in \{1,\ldots,\mathrm{k}\}$ **do**
    init $q_{\cdot f}$;
    for $l \in \{1,\ldots,\mathrm{p}\}$ **do**
      samples $w_{l,f}$ from $N(\hat{\mu}_{wl,f}, \hat{\sigma}^2_{wl,f})$;
      update e, q;
    end
  end
  $\#_{samples} \leftarrow \#_{samples} + 1$;
  $\hat{y}^*_{test} \leftarrow$ predict all cases S_{test};
  $\hat{y}_{test} \leftarrow \hat{y}_{test} + \hat{y}^*_{test}$;
until stopping criterion is met;
$\hat{y}_{test} \leftarrow \frac{1}{\#_{samples}} \hat{y}_{test}$;

Figure 3.8: MCMC main steps. Figure from [94]

Chapter 4

Data fusion techniques

As discussed in Chapter 1, in recent years many fields experienced an incredible growth of the amount of data collected. The biomedical area is one of those characterized by the fastest increase of available information, primary related to the advent of low cost technologies for analyzing molecular data. Very often, different types of complementary information, coming from different data sources, become available for computational analysis. Extracting useful information from this stack of raw measures can be a challenging task, which requires the definition of novel strategies and the development of specific tools.

In recent times, the expression *data fusion* is often used to indicate the process of integrating heterogeneous data, coming from different sources [99]. The basic idea is to combine all the information available in order to improve the prediction performance of a model [100].

In this chapter, the problem of data integration will be addressed. First of all, different strategies of integration will be introduced. Then, two different algorithms, based on matrix factorization techniques, will be described.

4.1. Types of data integration

A common way to discriminate the different data integration strategies is to consider the step of the analysis process, in which the actual integration is performed [101,102]. Three main approaches can be identified (Figure 4.1):

- Early integration: the data fusion is performed at the beginning of the process. In this case, a pre-processing step is required to make the data homogeneous and comparable. After this phase, all the data are integrated in a unified dataset (e.g. a single table with all the features)

and after that standard machine learning techniques can be applied. For example, let's suppose that the aim is to predict the risk for a patient of developing a disease given a series of data sources (generic personal data, clinical exams, somatic mutations, literature…). Using this approach, it is necessary to create a traditional dataset, where all the information is modeled in terms of patients' attributes. After this phase, standard machine learning techniques can be applied. The drawback is that, following this strategy, the peculiar characteristics of each data source are lost. Even if it is possible to capture interactions between features coming from distinct sources, the modular structure of the data is neglected [103].

- Late integration: this strategy is the opposite of early integration. For each data source a different model is trained, and then the results are combined to compute the final output. Using the same example described above, in this case for each type of data a traditional dataset in built and for each of them a different algorithm can be applied (thus giving more flexibility to the model). As can be easily imagined, the challenging part lays in the integration of the partial results given by each method. Regarding the example, an appropriate meta-classifier is needed to merge the risk predictions obtained considering the different components (lifestyle, genetic factors, clinical situation…), in order to come up with a single holistic score. Of course with this approach, it is possible to fully exploit the peculiarities of each dataset, but it is necessary to model the relation of each dataset with the final target of the method. In addition, the integration of the results may be challenging, because it requires a way to weight the contribution of each classifier.
- Intermediate (or partial) integration: this approach consists of performing the integration during the learning of the model. More precisely, the structure of the data is incorporated within the structure of a joint model: in this way the original the information can be exploited for the inference. Using the same example used earlier, in this case an overall disease risk for each patient is computed by considering all the available information at the same time. Of course, the algorithm must be able to exploit both patient-related properties (e.g. personal data, clinical exams, physiological measurements…) and the relation between the different data types (e.g, disease-disease associations, drug-disease effectiveness, gene-disease associations…).

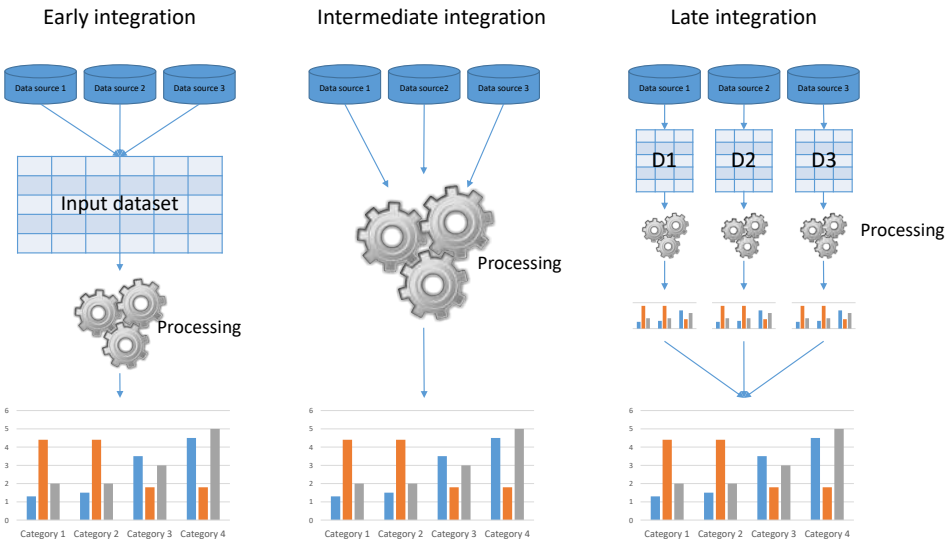

Figure 4.1: Graphical representation of the different types of data integration strategies.

The third strategy, the intermediate integration, is potentially the most accurate, since it retains the structure of the original data. Anyway, the key point of this solution is, of course, the definition of the inference algorithm. Due to the complexity of the problem, it is not trivial to develop a general methodology, independent on the problem to address.
Some classes of algorithms have been proposed to operate an intermediate data fusion. Many of them are just heuristic methods heavily bound to the specific application. But there are also some general approaches such as graphical model-based methods, multiple kernel-based methods and latent factors models. Regarding the first ones, there exist different application of Bayesian hierarchical modeling used to perform predictions using different sets of objects [104]. For example, in order to transfer information coming from different data sources it is possible to place common priors on top of hierarchical models [105]. On the other hand, kernel methods use non-parametric kernel functions to express the similarity between pairs of objects. Different kernels may correspond to different notions of similarity and different kernels can be used for inputs coming from different data sources. In order to perform data fusion, a combinations of the different kernels can then be obtained using linear or more sophisticated non-linear functions [106,107].
In this dissertation, the attention is focused on methods based on latent factors, obtained by performing a joint decomposition of the starting data matrices.

4.2. Approaches based on matrix factorization

In this section, two examples of data fusion algorithms, based on matrix factorization techniques, will be described. The first one, the Tri-factorization algorithm, has been recently developed [103] and successfully applied to many biomedical problem. The second one, Bayesian matrix factorization for data fusion, is a novel method introduced in this thesis. Both of them have been implemented in the context of this dissertation. At the end of the section, a brief comparison between the two methods will be discussed.

4.2.1. Matrix tri-factorization algorithm

The tri-factorization algorithm can be seen as a multi-level extension of the generic matrix decomposition used in the field of recommender systems. This method has been already successfully applied to a variety of problem, such as discovering novel disease-disease associations using molecular data [108], prediction of drug-induced liver injury [109], gene prioritization [110], gene function prediction [111], for drug repurposing for triple negative breast cancer [112] and for multiple protein network alignment [113].

4.2.1.1. Input data

The philosophy behind this method is to structure the available information in matrix form. The basic assumption is that every input data must be interpretable as an interaction between two objects. For example, given microarray data, containing gene expression values for a cohort of patients, the related information can be easily structured in a matrix, where the rows represent the patients, the columns represent the genes and each entry of the matrix represents the expression value for a certain pair (patient, gene). Of course a lot of traditional machine learning datasets can be easily modeled in this way, except those characterized by higher order interactions. In this case, tensors need to be used and the complexity of the model drastically increases.

The Tri-factorization algorithm distinguishes two types of input matrices [103]: the relation matrices, $R_{ij} \in \mathbb{R}^{n_i x n_j}$, and the constraint matrices, $\theta_i \in \mathbb{R}^{n_i x n_i}$. The first type refers to the case of interactions between two different types of objects (e.g. patient-gene interactions). The algorithm requires the entries to be bound in the [0,1] interval, where 1 indicates a very strong interaction, while 0 represents the absence of interaction or the lack of knowledge about it. The constraint matrices, instead, are used to model relationships between objects of the same type (e.g. patient-patient similarities or gene-gene interactions). The entries of the θ_i matrices must

be bound in the [-1,1] interval, where -1 indicates a must-link, i.e. a very strong association, while 1 represents a cannot-link, i.e. a nearly impossible association. Due to the bounded intervals, a pre-processing phase is required to rescale the interaction values inside the appropriate limits.

The central point of Tri-factorization method is that the multiple data sources to integrate in the model must share some of the object types involved in the analysis. So, for example, in case of bioinformatics data, it is possible to include different types of interactions for an object of type "gene": patient-gene expression, gene-protein, gene-miRNA, gene-gene… In turn, all the other objects may have other interactions: e.g. for patients, patient-drug, patient-disease, patient-miRNA expression… In practice, each R_{ij} matrix, relates the two objects types, i and j, that are involved in other relations.

The set of R_{ij} can be then used to define a comprehensive block matrix, $R \in \mathbb{R}^{NxN}$, with $N = \sum_i n_i$, containing all the available associations:

$$R = \begin{bmatrix} 0 & R_{12} & \cdots & R_{1r} \\ R_{21} & 0 & \cdots & R_{2r} \\ \vdots & \vdots & \ddots & \vdots \\ R_{r1} & R_{r2} & \cdots & 0 \end{bmatrix} \tag{75}$$

In order to obtain compatible dimensions, the rows of R_{ij} must be made equal to the number of the objects of type i included in the model and in the same way the columns of R_{ij} must be made equal to the number of the objects of type j. The final R matrix will have null block matrices on the main diagonal, because that type of information is contained in the θ_i matrices. Some other blocks may be null, because the related associations are missing (or meaningless). The R matrix can be symmetrical, therefore to each R_{ij} is associated a $R_{ji} = R_{ij}^t$, or non-symmetrical. In this latter case, two different kinds of information, relating the same two types of objects, can be included in the model.

As regards the θ_i matrices, the algorithm allows taking into account multiple interaction sources for each object. Indicating with f the maximum cardinality of the θ_i matrices, similarly to R, it is possible to define f block diagonal $\theta^{(f)}$ matrices, containing $\theta_i^{(f)}$ (if existing) as i-th block on the diagonal:

$$\theta^{(f)} = \begin{bmatrix} \theta_1^{(f)} & 0 & \cdots & 0 \\ 0 & \theta_2^{(f)} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \theta_r^{(f)} \end{bmatrix} \tag{76}$$

where r is the number of different object types included in the model.

In the same way of R_{ij}, for dimensional compatibility, in θ_i there exist a row and a corresponding column for each unique element of type i.

4.2.1.2. Method description

In the typical application contexts, both R and θ are characterized by very high sparseness, because, given the complete list of unique elements introduced in the model, just few interactions are known. In the same way as recommender systems, the objective of the method is to fill the gap in order to discover novel interesting unknown associations between two specific types of objects. This is achieved by factorizing the starting R matrix into the product of three terms:

$$\hat{R} \approx GSG^t \tag{77}$$

where:

- $G \in \mathbb{R}^{NxK}$ is a non-negative block diagonal matrix, and block $G_i \in \mathbb{R}^{n_i x k_i}$ is related to the i-th object type, and $K = \sum_i k_i$. The k_i terms, also called ranks, define the dimension of the latent factors for the i-th object type. In fact, as typically operated by the decomposition methods, the Tri-factorization performs a dimensionality reduction, with the objective of revealing hidden structures in the data. For this reason, k_i has to be chosen much smaller than the associated $n_i$dimension. For each type of objects, a different rank parameter can be defined, in general depending on the sparseness of the associated relational matrices. Each row of G_i is a low dimensional vector, related to a specific element of type i. Its values indicate the weight of the corresponding latent factor for the specific element.

$$G = \begin{bmatrix} G_1^{n_1 x k_1} & 0 & \cdots & 0 \\ 0 & G_2^{n_2 x k_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & G_r^{n_r x k_r} \end{bmatrix} \tag{78}$$

- $S \in \mathbb{R}^{KxK}$ is a squared block matrix. It has null blocks on the main diagonal and for all the $S_{ij} \in \mathbb{R}^{k_i x k_j}$ for which the corresponding R_{ij} is null. This matrix plays the role of modeling the associations between the latent factors. Each $S_{ij} \in \mathbb{R}^{k_i x k_j}$ in particular, put in relation the latent features of the i-th type of objects with the latent features of the j-th type of objects. In

practice it represents a compressed version of the related R_{ij} in a smaller space, defined by the latent factors.

$$S = \begin{bmatrix} 0 & S_{12}^{k_1 x k_2} & \cdots & S_{1r}^{k_1 x k_r} \\ S_{21}^{k_2 x k_1} & 0 & \cdots & S_{2r}^{k_2 x k_r} \\ \vdots & \vdots & \ddots & \vdots \\ S_{r1}^{k_r x k_1} & S_{r2}^{k_r x k_2} & \cdots & 0 \end{bmatrix} \tag{79}$$

Therefore, the basic idea behind the method is to operate a joint decomposition of all the starting matrices. This results in a set of two types of matrices: G_i, with the latent factors for the i-th object, and S_{ij} with the pairwise relation, in the latent factor space, between object i and object j. Figure 4.2 depicts an example of Tri-factorization schema with three different types of objects.

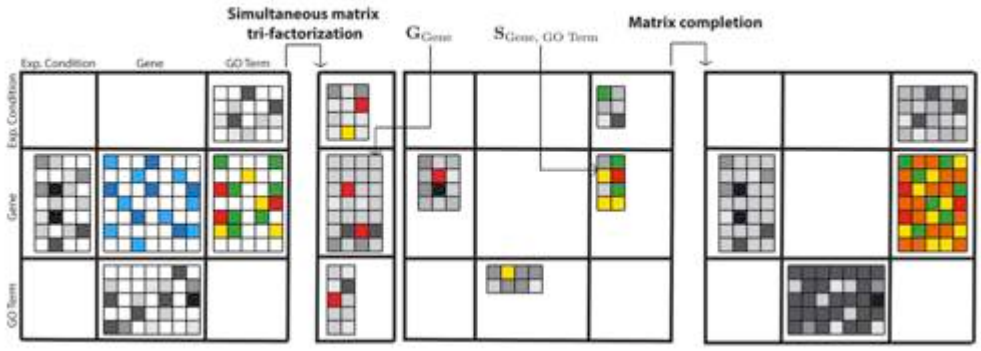

Figure 4.2: Example of Tri-factorization for three types of objects and five different data sources. Figure from [108]

Regarding the learning process of the model, the estimate of the values of the G and S matrices can be expressed as an optimization problem, based on the minimization of a cost function:

$$\min_{G \geq 0} J(G;S) = \sum_{R_{ij}} \left\| R_{ij} - G_i S_{ij} G_j^t \right\|^2_{Frobenius} + \sum_{f=1}^{\max_i f_i} tr(G^t \theta^{(f)} G) \tag{80}$$

where tr(.) denotes the trace of the matrix.

This cost function is composed of two distinct terms. The first one represents the reconstruction error, computed as the sum, over all the starting R_{ij}, of the squared Frobenius norms of the difference between the original matrix and the reconstructed one. The second part, instead, is related to the constraint matrices. This is very important because it penalizes the cost function based on the must-link and cannot-link. In this

way, it reduces the risk of overfitting, especially for those types of objects characterized by a small number of relation matrices.

This optimization problem cannot be solved in closed form. Anyway, in [103] an iterative method is proposed. After a proper initialization of the G_i factors, an alternate optimization of G and S is performed. So, keeping G fixed, S is updated, and then, keeping S fixed, G is updated. The update rules for the two types of matrices are obtained by computing the roots and the partial derivative of J, fixing the other matrix.

For the S matrix, the update rule is:

$$S \leftarrow (G^t G)^{-1} G^t R G (G^t G)^{-1} \tag{81}$$

For G, a multiplicative update rule is derived:

$$G \leftarrow G \circ \sqrt{\frac{(RGS)^+ + G(SG^tGS)^- + \sum_f((\theta^{(f)})^- G)}{(RGS)^- + G(SG^tGS)^+ + \sum_f((\theta^{(f)})^+ G)}} \tag{82}$$

where $\circ$ indicates the Hadamard product (i.e. the element by element product). $X^+$is a matrix whose generic entry $x_{i,j}^+$ is equals to $x_{i,j}$ if $x_{i,j} > 0$ and 0 otherwise, while $X^-$is a matrix whose generic entry $x_{i,j}^-$ is equals to $-x_{i,j}$ if $x_{i,j} < 0$ and 0 otherwise. For this reason X^+ and X^- are both non-negative.

This operation is repeated until the algorithm converges to a local minimum of the cost function J. A possible stopping criterion is based on the reconstruction error of a target matrix. For example, if the objective is to estimate new relationships between objects of type i and j, a reasonable solution is to monitor the following norm:

$$err = \left\| R_{ij} - G_i S_{ij} G_j^t \right\|^2_{Frobenius} \tag{83}$$

When the difference between this quantity in two consecutive iterations goes below a certain threshold, the algorithm stops. For computational reasons, the assessment of the error can be evaluated only after a certain number of iterations.

In [103], the mathematical correctness of the method is reported. In addition, the demonstration of the fact that J is nonincreasing, using the reported update rules for G and S, is shown.

4.2.1.3. Parameters choice

A crucial aspect of the method is the choice of the factorization ranks. As well as for the other matrix decomposition methods, a high number may

lead to overfitting, while a too small number may not be enough to capture all the information. There is no general consensus about how to select these values. A grid search approach was proposed in [103], based on the computation of the model performance testing, for each k_i, a certain number of values from a predefined interval. In order to reduce the number of tests to try, they proposed a sort of bisection method. For each parameter, they start with the midpoint and the border of the interval, and then they repeat the process by selecting the subinterval giving the best result. The research can be stopped when some criterion is met, for example when the cophenetic correlation coefficient starts to decrease [103].

Another strategy, adopted for this work, is to select the rank parameters on the basis of the total number of interactions available for each type of object. This can be easily computed considering, for a certain object type i, the overall associations, N_i, modeled by all the related relation matrices R_{ij} and R_{ji} $\forall$ j. Afterwards, the obtained value is scaled by a proper factor λ:

$$k_i = \frac{N_i}{\lambda} \tag{84}$$

In this way it is possible to reduce the number of parameters to tune just to a single value. A set of reasonable λ values can be tested, choosing the one giving the best performance.

4.2.1.4. Inference of new associations

After the learning phase, the computed factors can be employed for prediction purposes. First of all a relational target matrix R_{ij} must be selected: it indicates the objects among which new associations want to be discovered. In the same way of collaborative filters, the target matric can be approximate by the product of the related latent factors:

$$\hat{R}_{ij} \approx G_i S_{ij} G_j^t \tag{85}$$

While the starting matrix R_{ij} was sparse, the reconstructed one is dense, meaning that the gaps are filled with a certain value. A simple solution to identify new associations is to find the new entries in the reconstructed matrix with a value higher than an absolute threshold: higher values should indicate stronger interaction levels.

More sophisticate techniques can be employed to identify new associations in a more robust way. For example, it is possible to relate the values of the new predicted interactions with those of the originally known association. This can be computed in row-centric perspective or in the dual column-centric perspective [103].

Following the row-centric rule, a new interaction is considered as significant if its value in the reconstructed matrix is higher than the average of the entries of the same row, for which an association was known since the original relation matrix. In formulas:

$$\hat{R}_{ij}(p,q) > \frac{1}{\left|A(o_p^i, \varepsilon_j)\right|} \sum_{o_m^j \in A(o_p^i, \varepsilon_j)} \hat{R}_{ij}(p,m) \tag{86}$$

where p and q are the indices of the entry, $A(o_p^i, \varepsilon_j)$ is the set all the object of type j related to object o_p^i, of type i. A criticism of this approach is that if the entire row was null, it is impossible to compute that mean. In this case, an absolute threshold, for example the global mean, can be used.

A dual rule, the column-centric one, can operate the same computation on the columns.

It is also possible to combine the two rules in order to obtain a score value indicating the strength of the newly estimated associations. For example, after applying the row-centric rule, the distribution of the previously known associations in the same column can be computed. The strength of the predicted interaction can be therefore evaluated using its position in that distribution, i.e. by computing its inverse percentile in the distribution. A threshold can then be used to filter the results, keeping just the strongest interactions. Of course, as mentioned above, this operation can be critical if none of the values in the column was known at the beginning.

The described procedure is useful for extract information in case of interactions between objects of different types. However, in case of objects of the same type, a direct reconstruction of the interaction matrix is not possible. In fact, the R_{ii} blocks in the R matrix are null, while the interaction information for objects of the same type is modeled in the constraint matrices. In this case, some tricks have to be applied.

A solution to this problem, proposed in [108], is to exploit the G_i latent factors. Since the method operates a dimensionality reduction, each latent factor will be associated to a (hidden) holistic characteristic for that object type. Therefore, in principle, a sort of clustering can be performed by associating each element to a class depending on the highest component of its associated low dimensional vector. In practice, for each row of G_i the maximum value is computed and the related element is associated to that category. Two elements are considered similar, and therefore interacting, if they belong to the same class. A binary connectivity matrix C can be computed to summarize all the identified associations. Figure 4.3 shows an example of connectivity matrix construction starting from the related latent factor G_i.

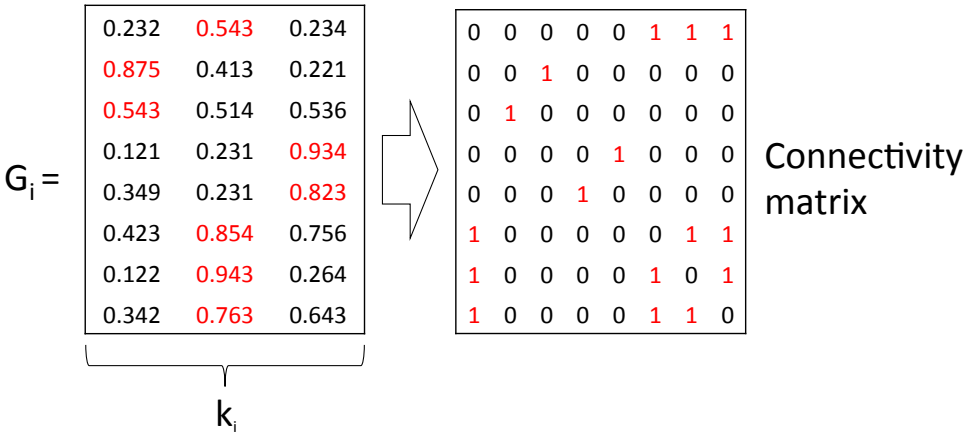

Figure 4.3: Example of construction of the connectivity matrix from a factor G_i.

4.2.1.5. Implementation details

The Tri-factorization algorithm has been implemented in the context of this thesis. Matlab and Octave code was used for all the steps. The first important choice was the selection of the initialization method for G_i. Some of the solution described in [103] were tested, in particular a first attempt tried to populate the G_i matrix with columns extracted from the related $R_{ij}\ \forall\ j$. Due to the high sparseness of the original matrices, very often this strategy led to instability problem during the learning phase. More sophisticated techniques, like applying SVD and the concatenation of the $R_{ij}\ \forall\ j$ and keeping the top left singular vectors, were tried. In any case, some instability problems were observed, due to an incontrollable growth of the values in the G_i during the training step. In fact, even if the cost function J has been demonstrated to be nonincreasing, the two terms defining it may growth in opposite directions. So, too high values in G_i may determine a high value of the reconstruction error, balanced by very negative values of the constraints part. For this reason, a random uniform initialization has been chosen. In particular, to avoid the problem described above, small values were used, typically in the range [0,10^-3].

The update rule for the S matrix requires computing a matrix inversion. This operation can be really burdensome from the computational point of view. For this reason, instead of computing the value of $y = (G^tG)^{-1}G^t$ using an explicit inversion, the associated linear system, $(G^tG)y = G^t$, is solved.

Due to the random initialization, the results of each run of the method can be different. For this reason, multiple repetitions of the algorithm are computed, in order to obtain more robust solutions. Since they are

independent from each other, this operation can be easily parallelized. For each repetition, a connectivity matrix C is determined for the target matrix. If the target is a relation matrix, C is computed on the basis of the row- and column-centric rule (with an appropriate threshold), otherwise the clustering method based on G_i factors is applied. Once all the repetitions are completed, a final consensus matrix $\bar{C}$ is computed by averaging all the obtained connectivity matrices.

4.2.2. Bayesian matrix factorization for data fusion

The problem of data fusion can be addressed also in a probabilistic fashion. Within the framework of latent factors models, and following a number of strategies that applied probabilistic extensions to matrix factorization [114–118], a novel method for data integration, based on probabilistic matrix factorization has been developed and implemented. The basis of the method is represented by the Bayesian matrix factorization model presented in [55] and described in Chapter 2. Of course, in order make it suitable to perform data fusion, it has been extended with the aim of performing a joint decomposition of all the starting matrices.

4.2.2.1. Input data

As well as for the Tri-factorization algorithm, the input data are supposed to have a matrix representation. The first step of the process is therefore to represent all the knowledge, coming from the different data sources, in form of relation matrices, each one formalizing the known associations between pairs of distinct elements. In order to propagate the information, some elements must be involved in multiple associations with different types of objects. To make the matrices comparable, the entries are normalized in the interval [0,1], where 0 represents the absence of (known) association, and 1 represents a very strong association. On the contrary of Tri-factorization, the associations between objects of the same type cannot be directly integrated in the model, unless exploiting a weak effect given by the hyperparameters setting.

4.2.2.2. Model description

The single matrix factorization considers the generic entry of a matrix as a stochastic variable. As for the other latent factor models, the decomposition is aimed at computing, for each element involved in the analysis, a low dimensional vector, representing its position in the latent factors space. However, in this probabilistic framework, the latent factors are model as random variables drawn from an appropriate distribution. Their values are used to define the parameters of the distribution

characterizing the entries of the matrix. In case of data fusion, the latent factor characterizing an element has to be related to the overall set of its relationships, even if coming from the different sources. On the contrary of the Tri-factorization algorithm, all the low dimensional vectors must be characterized by the same dimension k. The choice of this rank parameter is of course critical, also because it has to balance the different levels of sparseness characterizing the different objects.

Each entry in an observation matrix is considered as drawn from a Gaussian distribution, whose parameters depend on the related low dimensional vectors. For example, if U_i and V_j the low dimensional vectors for objects of type i and j and R_{ij} is the related interaction matrix, then:

$$p\left(R_{ij}\middle|U_i, V_j, \alpha_1, \alpha_2\right) = N(R_{ij}|U_i^t V_j, ({\alpha_1}^{I_{ij}} * {\alpha_2}^{1-I_{ij}})^{-1}) \quad (87)$$

where I_{ij} is a variable, indicating if R_{ij} is observed ($I_{ij} = 1$) or not (I_{ij}=0). In the first case, the α_1 precision parameter is used. Otherwise another precision parameter is utilized. While the first one plays the same role of the α used in the single matrix model, the second one is used to model the unknown relationships (i.e. the zero entries). The idea is to represent them as known observations but with less precision in comparison with the actual known observations. The motivation behind the introduction of this parameter is that the basic schema, taking in account just known associations, can be prone to overfitting, leading to a perfect reconstruction of the original matrices. This can make the method useless, because it doesn't allow highlighting novel interesting associations. The situation is particularly critical for very sparse matrices, which are very common in the application contexts for which the method was designed. For this reason it's important to suitably tune the α_2 parameter in order to allow an adequate flexibility to the model.

The other distributions are the same specified in [55] for the single matrix factorization. Therefore, for low dimensional vectors Gaussian distributions are chosen and Gaussian-Wishart distributions for the hyperparameters Starting from these assumptions, a possible way to overcome the problem of representing the interactions between pairs of objects of the same type is to exploit the hyperpriors' parameters. In particular, the W scale matrix characterizing the Wishart distribution can be interpreted as a condensed version of this type of information, represented in the latent factors space. Anyway, the algorithm has proved to be quite insensitive to the value of these parameters, therefore unless a very strong structure is present in the data, the effects of this step are negligible.

Figure 4.4 shows an example of the model for three different types of objects and three relation matrices.

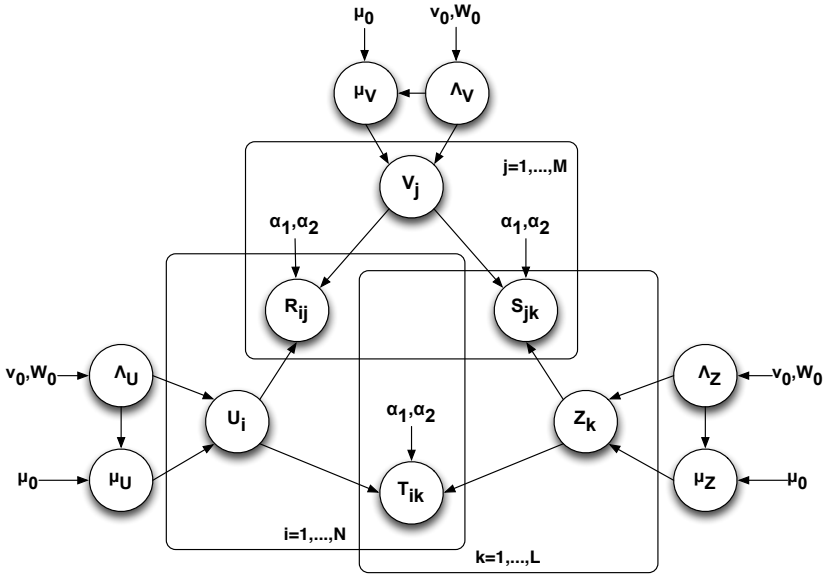

Figure 4.4: Graphical representation of the Bayesian matrix factorization model for data fusion. In this case, three different objects and three relation matrices are used.

4.2.2.3. Parameters estimation though MCMC

As regards the predictive distribution, as well as for the single matrix decomposition, to get the exact solution it would be necessary to compute a complex posterior distribution by integrating over all the parameters and hyperparameters. Since the analytical derivation of this distribution is intractable, a different strategy has to be employed. The solution adopted was to use a MCMC approach using on Gibbs sampling.

The main idea is to build a Markov chain that, through an iterative sampling process, converges to a stationary distribution that approximates the true posterior distribution.

In this particular case, each iteration consists of two main steps:

- During the first phase, a new value for every hyperparameter is independently drawn from the related Gaussian-Wishart distribution, conditioned on the current values of all the low dimensional vectors related to that particular type of object. The choice of conjugate distributions allows to compute a closed form solution, as reported in [55]. For example, for U:

$$p(\mu_U, \Lambda_U | U, \mu_0, \beta_0, W_0, \nu_0) = N(\mu_U | \mu_0^*, (\beta_0^* \Lambda_U)^{-1}) W(\Lambda_U | W_0^*, \upsilon_0^*) \quad (88)$$

with:

$$\mu_0^* = \frac{\beta_0\mu_0 + N\overline{U}}{\beta_0 + N}, \beta_0^* = \beta_0 + N, \upsilon_0^* = \upsilon_0 + N, \overline{U} = \frac{1}{N}\sum_i^N U_i$$
$$[W_0^*]^{-1} = W_0^{-1} + \sum_{i=1}^N U_i U_i^t + \frac{\beta_0 N}{\beta_0 + N}(\mu_0 - \overline{U})(\mu_0 - \overline{U})^t \quad (89)$$

Where N is the number of elements of type i.

- After that, for every different object a new vector is drawn from the related distribution, conditioned on both the values of the known observations and the values of the hyperparameters. The example for U is:

$$U_i^{t+1} \sim p(U_i|R, V^t, Z^t, \mu_U^t, \Lambda_U^t) \quad (90)$$

where t represent the previous iteration. This full conditional distribution is still a Gaussian distribution, depending on the current values of the low dimensional vectors of all the other types of objects for which a relation matrix exists. Using again the example of U:

$$p(U_i|R, T, V, Z, \mu_U, \Lambda_U, \alpha_1, \alpha_2) = N(U_i|\mu_i^*, [\Lambda_i^*]^{-1}) \quad (91)$$

with:

$$\Lambda_i^* = \Lambda_U + \sum_{j=1}^M \left[V_j V_j^T\right] * {\alpha_1}^{I_{ij}} * {\alpha_2}^{1-I_{ij}} + \sum_{k=1}^L [Z_k Z_k^T] * {\alpha_1}^{I_{ik}} * {\alpha_2}^{1-I_{ik}} \quad (92)$$

and:

$$\mu_i^* = [\Lambda_i^*]^{-1}\left(\begin{array}{c}\sum_{j=1}^M \left[V_j(\alpha_1 R_{ij})^{I_{ij}}(\alpha_2\varepsilon)^{1-I_{ij}}\right] + \\ + \sum_{k=1}^L [Z_k(\alpha_1 T_{ik})^{I_{ik}}(\alpha_2\varepsilon)^{1-I_{ik}}] + \Lambda_U\mu_U\end{array}\right) \quad (93)$$

One type of object at a time is updated, using the current estimate of the low dimensional vectors of all the others. In this way, the simultaneous factorization of all the available data matrices is

performed. Since the elements belonging to the same type of object don't depend on each other, it is possible to parallelize this operation by independently sampling the single low dimensional vectors.

After a burn-in phase, the algorithm converges to the posterior distribution of the unknown variables:

$$p(\widehat{obs}_{ij}|obs) \approx \frac{1}{B}\sum_{b=1}^{B} p(\widehat{obs}_{ij}|parent_i^{(b)}, parent_j^{(b)}) \quad (94)$$

Choosing an arbitrary large number, B, of the last samples it is possible to reliably compute sufficient statistics. After the learning phase, each element is associated to a set of vectors, one for each sampling, with its representation in the latent vectors space.

4.2.2.4. Inference from predictive distribution

Starting from the set of low dimensional vectors extracted during the learning phase, it is possible to compute the related posterior distribution of the predicted interaction for each possible pair of elements. Using an approach similar to the one employed by the Tri-factorization algorithm, a certain target relation matrix, representing the interactions between two specific types of object, can be fully reconstructed. In correspondence of the known associations the reconstructed value should be similar to the original one, but in correspondence of the new potentially relevant relationships the distribution of values should be considerably distant from the zero value. In order to identify the most promising interactions, different strategies can be adopted. For example, a simple threshold on the mean of the posterior distribution may be enough to filter the results, keeping just the highest. Otherwise, more sophisticated methods can be applied. The advantage of the probabilistic approach, in fact, is that instead of a single value (or a bunch of values in case multiple repetitions), an entire probability distribution is available. For this reason, many statistical properties can be associated to each interaction, as for example a measure of uncertainty based on the dispersion of the samples.

4.2.3. Comparison of the two methods

After the description of these two methods, it is worthwhile to highlight the similarities and the differences between them.

Both the methods rely on matrix factorization techniques in order to represent the available information in a low dimensional space. They both can integrate different data sources, whose knowledge is represented in matrix form. However, while the Tri-factorization may handle natively

associations between objects of the same type, the Bayesian method can only integrate them in and indirect way, by properly tuning its parameters.

Another difference is determined by the rank parameters. While for Tri-factorization a different rank parameter must be specified for each type of objects, in the Bayesian method the same dimension characterizes all the latent factors allowing less flexibility to the model.

Regarding the predictions, Bayesian factorization provides a more robust and complete interpretation of the results with respect to other approaches like the Tri-factorization algorithm. The probabilistic assumptions behind the model, in fact, allow giving an estimate of the uncertainty of the predicted associations. Of course the drawback is that this approach imposes a model (in this case Gaussian) to the data, even if it is not always a correct assumption.

From the computational point of view, both methods exploit iterative algorithms in order to estimate the model's factors. While Tri-factorization requires in each step to use the entire dataset, the Bayesian method can exploit its conditional independence properties to perform independent sampling from the distributions, making it suitable to be parallelized.

An example of application of the Bayesian method, as compared with Tri-factorization, is reported in Appendix 1.

Chapter 5

Data fusion in myeloid neoplasms

In this chapter, two different applications of previously described matrix factorization techniques are presented. Each case study is focused on a different blood cancer, both belonging to the myeloid neoplasms category: the myelodysplastic syndroms (MDS) for the first analysis, and the acute myeloid leukemia (AML) for the second one. The available data and the purposes of the two analyses were different, but both of them exploited matrix factorization-based algorithms to reach their objective.

5.1. Myelodysplastic syndromes

The term myeloid neoplasms indicates a category of blood tumors originating from myeloid stem cells [119]. In normal conditions, the bone marrow produces blood stem cells, which are immature in the beginning and become mature over time. After the differentiation and maturation phase, these cells originate most of the blood cell types: red blood cells, platelets and granulocytes (a type of white cells).

Myeloid neoplasms are clonal diseases of these hematopoietic stem cells. The causes are typically genetic and epigenetic alterations, resulting in an abnormal activity of key processes such as self-renewal, proliferation and impaired cell differentiation [120,121].

In 2016 the World Health Organization (WHO) revised the classification of myeloid neoplasms and acute leukemia [122]. This categorization was based on a large number of characteristics: morphology, cytochemistry, immunophenotype, genetics, and clinical features. The classification is periodically updated to include the newly available information, in order to better characterize the different subgroups of tumors. Five main types of malignancies are distinguished:

- Myeloproliferative neoplasms (MPN)

- Myeloid/lymphoid neoplasms with eosinophilia and rearrangement of PDGFRA, PDGFRB, or FGFR1, or with PCM1-JAK2
- Myelodysplastic/myeloproliferative neoplasms (MDS/MPN)
- Myelodysplastic syndromes (MDS)
- Acute myeloid leukemia (AML) and related neoplasms

Each of these categories can be further differentiated on the basis of several factors, in order to better characterize different disease subtypes.

In this first case study, the attention is focused on Myelodysplastic syndromes (MDS). Patients affected by this malignancy present morphologic dysplasia (i.e. an abnormal development and differentiation) in hematopoietic cells and peripheral cytopenia (i.e. reduction of the number of produced cells) [122]. The same effects can be the result of different factors, therefore the diagnosis of MDS is still challenging. The revised classification is based on a more precise morphologic interpretation and cytopenia assessment, and above all it heavily includes genetic information (easily available thanks to the new cheaper technologies) for the diagnosis and the classification. Therefore, a study focused on the analysis of the genetic network characterizing this type of disorders seems to be an interesting starting point to better understand and treat the pathology.

5.1.1. Problem description

Within the context of MDS and myeloid neoplasms in general, the analysis of genetic machinery underlying the pathology is currently a hot topic, as described in [122]. Many studies have been carried on, from gene expression profiling [123] of MDS patient, to the analysis of the different somatic mutations affecting the stems cells [124–126], to the combination of mutation data with expression data to increase the performance in outcome prediction [127].

Under this perspective, this dissertation presents an analysis focused on the integration of different types of data sources. The final aim was to discover novel interesting gene-gene interactions characterizing MDS patients. For this purpose, the Tri-factorization algorithm, described in Chapter 4, has been properly set up and employed. Many different data, in particular molecular data, were included in the model, mainly retrieved from public data sources. The objective of the study was to evaluate the possibly of exploiting this information in order to highlight unknown genetic interactions characterizing the pathology.

5.1.2. Available data

Different types of data were collected. The starting point was a dataset of mutations provided by the Department of Hematological Oncology of Policlinico San Matteo (Pavia, Italy). The other information instead came from publically accessible databases.

Going into the details, the types of data employed were:

- Somatic mutation data. This information was the result of a mutation screening conducted on patients affected by MDS. For each mutation, different data were collected: chromosome, gene, starting position, ending position, sequence of reference and alternative sequence introduced by the mutation. Most of these mutations were point mutations with the alteration of a single base.
- Pathways data. Pathways provide useful information to include in the model. They, in fact, represent the biological processes occurring inside a cell; therefore their knowledge can be helpful in the analysis of the tumor cell mechanisms. The data source utilized was KEGG, from which different kinds of information about all the modeled pathways were collected: the genes, the interaction between different pathways and the direct associations with diseases.
- Gene data. Genes represent the focus of the study, so many forms of associations were included in the model for this type of object. First of all, each mutation was mapped to its associated gene. Then, already known human gene-gene interactions, coming from the public database BioGRID [16], were obtained. Different kinds of interactions are held by this database: direct physical binding of two proteins (produced by the related genes), genetic interaction or, more generally, co-existence in a stable complex. BioGRID curates the results from the experiments published in scientific journals, therefore for each pair of genes, multiple values may be recorded. Another source of information about genes is, as mentioned above, KEGG, since for each pathway a set of gene can be involved in the biological process. In addition, it is also possible to related genes and diseases. A curated set of human gene-disease associations were obtained from the public database DisGeNET [128]. In particular, this database provides a score that ranges from 0 to 1, indicating the evidence level. This value is computed by taking into account the number and type of data sources as well as the number of publications supporting that association.
- Disease data. This is another important type of object, since different diseases may share some disorder at the cellular level. As mentioned above, it was possible to obtain the associations between genes and disease from DisGeNET. Then, all the diseases altering each pathway were extracted from KEGG. In addition the

associations between human diseases were obtained, from the Disease Ontology [11]. Since it is an ontology, it has a semantic structure. Therefore, it is possible to exploit this hierarchy to estimate the distance/similarity between two diseases, based on the number of steps needed to reach one disease from the other one on the hierarchy.

5.1.3. Preprocessing and matrices construction

Given all the available data, they needed an appropriate pre-processing in order to be used by the Tri-factorization algorithm. Five type of objects were included in the model:

- 6462 diseases
- 255 patients
- 761 mutations
- 10513 genes
- 383 pathways.

Figure 5.1 shows the schema utilized for the Tri-factorization, with the related data sources.

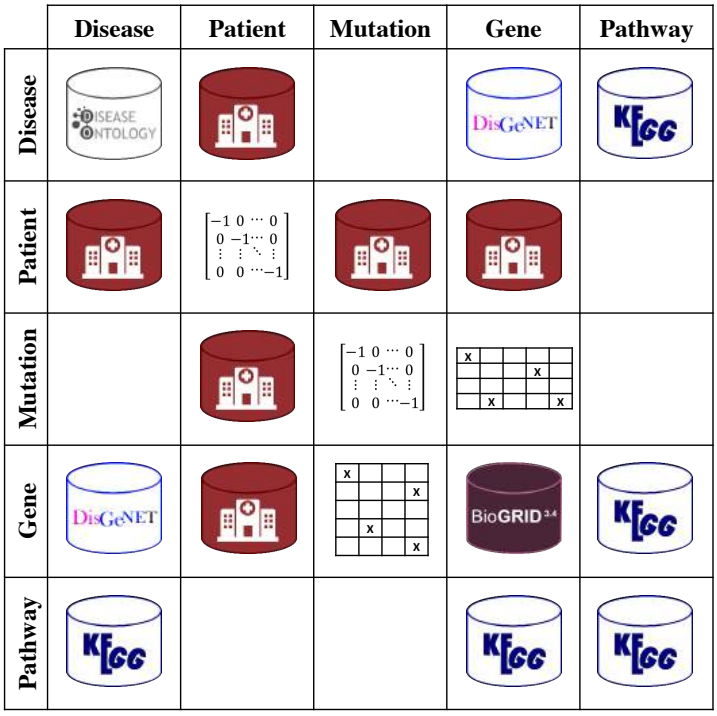

Figure 5.1: Adopted Tri-factorization schema.

The complete R matrix is symmetrical, therefore for each pair of objects just one type of data was used. In the following, the construction of each matrix will be described:

- Disease-disease constraint matrix: it contains the associations for each pair of diseases. It has been obtained using the semantic structure of the diseases in the Disease Ontology. In particular, the constraint between two diseases is set to -0.8^n, where n is the length of the shortest path between corresponding terms in the Disease Ontology hierarchy (i.e. the minimum number of steps to reach one disease from the other one). This value is negative, since for constraint matrices, negative values represent similarities. In total, 35201 interactions were included in the matrix
- Disease-patient: the patients considered in the analysis were only the 255 coming from the San Matteo hospital. The same disease, the MDS, characterized all of them. For this reason, this matrix is completely empty, except for one row, the MDS one, completely full of 1 to indicate the association.
- Disease-gene: from DisGeNet, 13281 associations of this kind were obtained. A transcoding of the diseases, from UMLS to DO, was needed, since the data provided by DisGeNet don't contain the DO terms used in the other disease-related matrices. Since the DisGeNet score is already in the interval [0,1], no further processing was required.
- Disease-pathway: 605 associations were found from KEGG. Also in this case, a transcoding of the diseases, from MESH and OMIM to DO, was needed.
- Patient-patient: since no further information about the patients was available, this matrix is completely empty except for the main diagonal, which is set to -1 to indicate that each patient is equal to himself.
- Patient-mutation: the data from San Matteo hospital were used. To each mutation a unique identifier was assigned by concatenating 5 data (chromosome, starting position, ending position, sequence of reference and alternative sequence). Therefore this matrix indicates for each patient all the related mutations, with the value 1 in the corresponding entry. The number of measured mutations was 778, out of 761 unique mutations. This means that only few mutations are shared by different patients,
- Patient-gene: since most of the mutations are unique, this matrix is important because it denotes, for each patient, which genes are mutated. Also in this case, of course, the number of non-zeros entries is equal to 778.
- Mutation-mutation: as for patient-patient this matrix is completely empty except for the main diagonal, which is set to -1 to denote that each mutation is equal to itself.

- Mutation-gene: it simply represents the mapping of the mutations over the associated gene.
- Gene-gene: it contains gene-gene interactions from BioGrid. The raw data needed a preprocessing step, since, as mentioned above, for each pair of genes, the related association may appear multiple times. For this reason, denoting with x the number of times a certain pair appears, its score is determined by:

$$f(x) = -\frac{1}{2} * \left(1 + \frac{\ln(x)}{\ln(x_{max})}\right) \tag{95}$$

It means that the range goes from -0.5 to -1, using a logarithmic function. 24809 unique interactions were at last included in the model

- Gene-pathway: from KEGG, it contains mapping of the genes inside the pathways. This matrix is characterized by 25345 entries.
- Pathway-pathway: again from KEGG, it contains 1957 entries with value -1, representing associations between pathways.

Figure 5.2 summarizes the dimensions of the data used in the analysis.

	6462	255	761	10513	383
6462	35201	255	0	13281	605
255	255	255	778	778	0
761	0	778	778	778	0
10513	13281	778	778	24809	25345
383	605	0	0	25345	1957

Figure 5.2: dimensions of the problem.

Of course all the matrices were arranged in order to have compatible dimensions and the same ordering of the elements.

5.1.4. Tri-factorization setting

After the preparation of the input matrices, the parameters of the Tri-factorization algorithm had to be set. After some empirical attempts, the

ranks of the latent factors were chosen using a scale factor equals to 200. It means that, each G_i matrix had a number of columns equals to the number of interactions (included in the relation matrices) of all the elements of type i, divided by 200. The final ranks for the five objects are reported in Table 5.1

Table 5.1: Values of the rank parameter for the different objects

Object type	Rank
Disease	71
Gene	201
Mutation	8
Pathway	130
Patient	9

As explained in the beginning, the target matrix in this case was the gene-gene interactions matrix. Since it is not possible to directly reconstruct it using by multiplication, a connectivity matrix was defined from the G_i factor associated to genes as described in Chapter 4.

The convergence of the algorithm was monitored by measuring the norm of the reconstruction error of a relation matrix, specifically the gene-mutation matrix. The algorithm stopped when the difference between two consecutive norms was under the threshold 10^{-5} or after a maximum number of iterations of 10,000.

The number of repetitions to build the final matrix was set to 50, in order to reduce the effects of random initialization. After that, a global consensus matrix was built, averaging all the 50 connectivity matrices.

5.1.5. Results

Starting from the consensus matrix, some analyses on the results were conducted. First of all, in order to filter the number of the newly predicted gene-gene interactions, just those identified in every repetition were kept (i.e. those with the value 1 in the corresponding entry of the consensus matrix). This operation led to the identification of 323 gene-gene interactions. Three of them were already included in the starting matrix from BioGRID, the others, instead, were novel predicted pairs. On the basis of these results, a set of analyses was carried on in order to evaluate the relevance of the found interactions in the context of the specified problem.

The complete list of 323 gene-gene interactions can be found in Appendix 2.

5.1.5.1. Genetic interaction networks

Starting from the predicted pairs, the related genetic interaction networks have been constructed, by linking all the genes according to their predicted interactions. Figure 5.3 shows the most important networks (i.e. those characterized by the largest number of interacting genes). Some of the links are red: they represent already known associations coming from BioGRID. They were included to link distinct subnetworks.

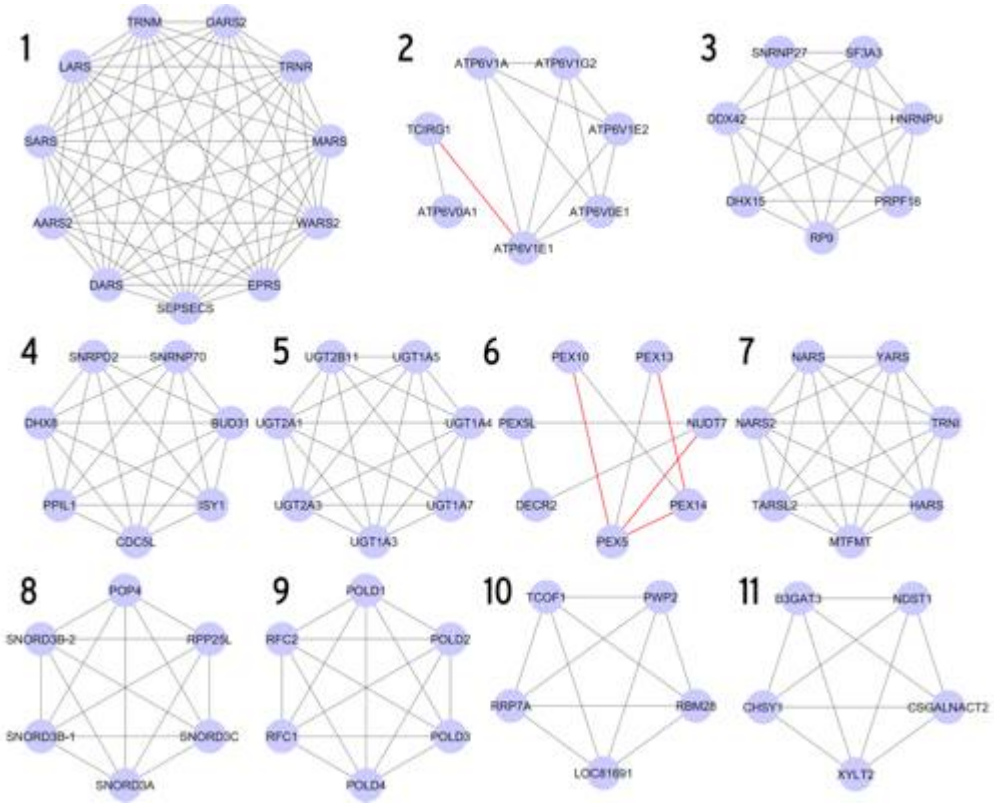

Figure 5.3: most important subnetworks identified from the consensus.

Figure 5.4 shows all the other minor networks.

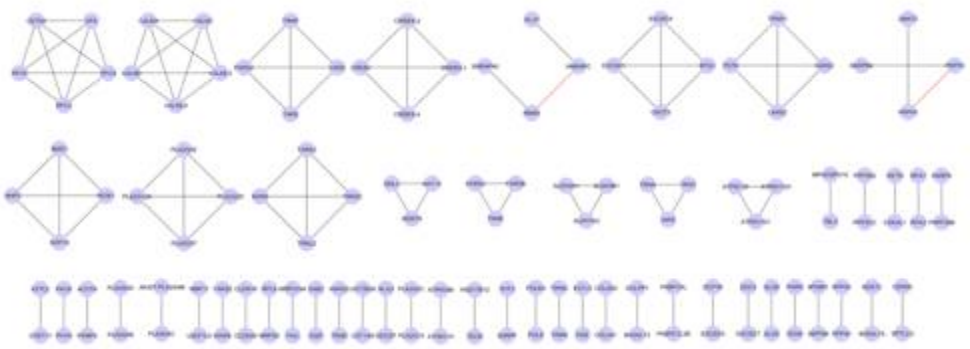

Figure 5.4: minor subnetworks identified from the consensus.

5.1.5.2. Enrichment analysis with Reactome

Some of the subnetworks, in particular those characterized by at least 7 genes, were further investigated. An enrichment analysis was carried on, in order to individuate the biological processes that are most influenced by the set of genes represented in each subnetwork. In order to do that, the pathways from another curated database, Reactome [14], were used. For each subnetwork, all the pathways were tested, computing the number of genes of the network also present in the pathway. The goal was to evaluate if the set of genes was over-represented in a significant way in the pathway. An exact Fisher test, based on hypergeometric distribution, was performed to determine the probability of obtaining the same number of genes by chance. This p-value was then corrected for false discovery rate using the Benjamini-Hochberg correction. Then, a threshold of 0.001 was applied to the adjusted p-values to select just the most significant.

For 4 of the considered subnetworks, significant associations with pathways were found. They are depicted in Figures 5.5-5.8, where to a bigger node corresponds a lower p-value and where the width of the link between nodes indicates the number of genes shared by the related pathway.

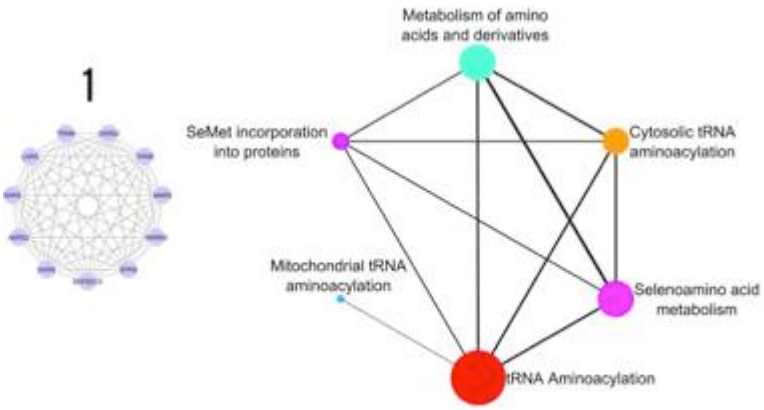

Figure 5.5: results of the enrichment analysis conducted on subnetwork 1.

The largest subnetwork (Figure 5.5), results in different pathways, in particular related to metabolism and tRNA aminoacylation. While the first term is very generic, specific aminoacyl tRNA synthetases are connected to the etiology of several diseases including cancer [129]. Particularly interesting is the pathway associated to mitochondrial tRNA, because MDS is often associated with mitochondrial dysfunctions. In particular, disorders caused by mitochondrial DNA mutations, were found to be associated with MDS in many studies [130].

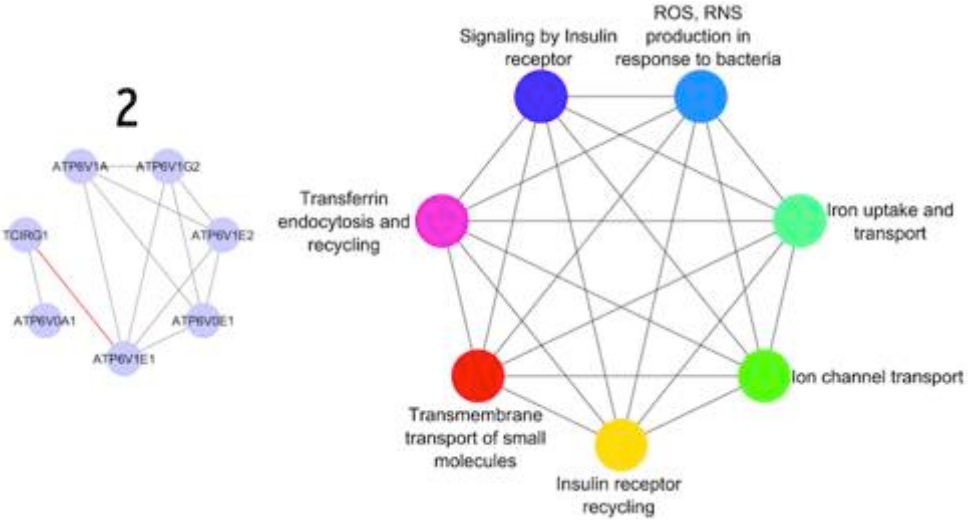

Figure 5.6: results of the enrichment analysis conducted on subnetwork 2.

In the second subnetwork (Figure 5.6), two processes are particularly interesting: the iron uptake and transport and the ion channel transport. In fact, patients with MDS may express an iron overload [131,132], while the ion channels are studied generically in the field of blood tumors like leukemia, since several ion channels and pumps, and other transport mechanisms are often altered [133].

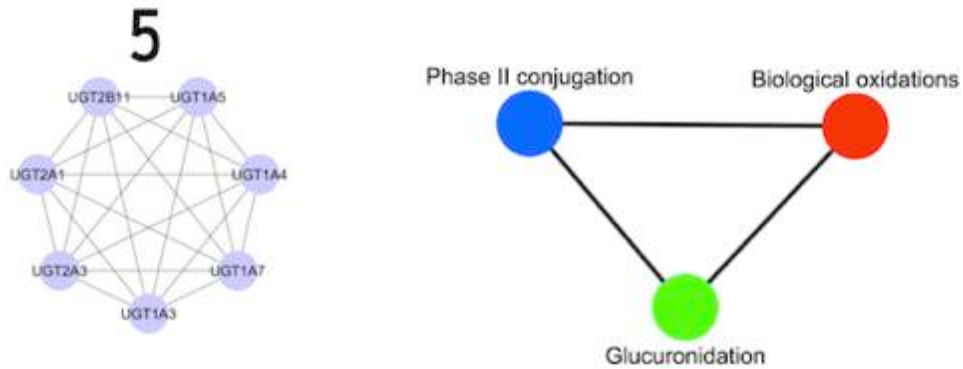

Figure 5.7: results of the enrichment analysis conducted on subnetwork 5.

The fifth subnetwork (Figure 5.7) is interesting due to the oxidation process. Since mitochondria are a main source of biological oxidation and energy transformation [134], this may refer again to the mitochondrial DNA alterations often present in MDS patients.

Figure 5.8: results of the enrichment analysis conducted on subnetwork 7.

The seventh subnetwork (Figure 5.8), instead, refers again to the tRNA aminoacylation, as already discussed for the first subnetwork.

5.1.5.3. KEGG pathways analysis

Another type analysis was conducted using the pathways from the KEGG database. First of all, still from KEGG, 14 genes with known associations with MDS were selected. Then, all the KEGG pathways involving at least one of those genes were extracted. For each of them, a count was made by considering the number of pairs, from the 323 predicted by the algorithm, for which both the genes were included in the pathway. The results are shown in Table 5.2

Table 5.2: pathways related to MDS genes and relative number of pairs found.

Pathway	Seed genes	# pairs found
VEGF signaling pathway	NRAS	1
Ras signaling pathway	NRAS	17
HTLV-I infection	NRAS,TP53	8
Insulin signaling pathway	NRAS	10
Spliceosome	U2AF1,SF3B1,SRSF2	49
Axon guidance	NRAS	1

Sphingolipid signaling pathway	TP53,NRAS	1
Long-term potentiation	NRAS	11
Peroxisome	IDH2,IDH1	5
Melanogenesis	NRAS	16
Alcoholism	NRAS	16
Amyotrophic lateral sclerosis (ALS)	TP53	1
Huntington's disease	TP53	6
Rap1 signaling pathway	NRAS	10
Neurotrophin signaling pathway	NRAS,TP53	10
Natural killer cell mediated cytotoxicity	NRAS	1
Oxytocin signaling pathway	NRAS	11
Metabolic pathways	IDH2,IDH1,DNMT3A	87
Tuberculosis	JAK2	13
Cholinergic synapse	NRAS,JAK2	6
T cell receptor signaling pathway	NRAS	1
B cell receptor signaling pathway	NRAS	1
PI3K-Akt signaling pathway	NRAS,JAK2,TP53	6
Hepatitis B	NRAS,TP53	6
Hepatitis C	TP53,NRAS	1
Glioma	NRAS,TP53	10
Wnt signaling pathway	TP53	1
Prostate cancer	NRAS,TP53	6
Estrogen signaling pathway	NRAS	16
MAPK signaling pathway	TP53,NRAS	1
GnRH signaling pathway	NRAS	10
Longevity regulating pathway	NRAS,TP53	6
Viral carcinogenesis	NRAS,TP53	6
Tight junction	NRAS	1

Very interestingly, a lot of pairs were found for metabolic pathways (which is a very generic term) and Spliceosome, which is known to be related to the MDS [135,136]. Same thing for the Ras signaling pathway, characterized by hyperactivation in MDS [137] and Rap1 signaling pathway, that plays a crucial role in the pathogenesis of some hematologic malignancies [138]. Also the Akt signaling pathway, was proved to be strongly related to MDS [139]. Some of the pathways are instead related to the other branch of myeloid tumors, the lymphatic ones. For example, the HTLV-I is a human virus type responsible of T-cell leukemia [140]. The T cell receptor signaling pathway, the B cell receptor signaling pathway and the Wnt signaling pathway were found deregulated in studies on MDS [141]. Finally, some other pathways are generically related to cancer (Glioma, Prostate cancer, Viral carcinogenesis). This means that many of the gene-gene pairs predicted by the algorithm are co-present in biological processes more or less strictly related to MDS.

5.1.5.4. Protein-protein interactions

Starting from the discovered gene-gene associations it is possible to analyze the related protein-protein interactions. A public database, STRING [15], contains protein-protein interactions supported by different types of evidence: Neighborhoods, Gene Fusion events, Co-occurrence events, Co-expression data, Experimental data, Database information and Text-mining association.

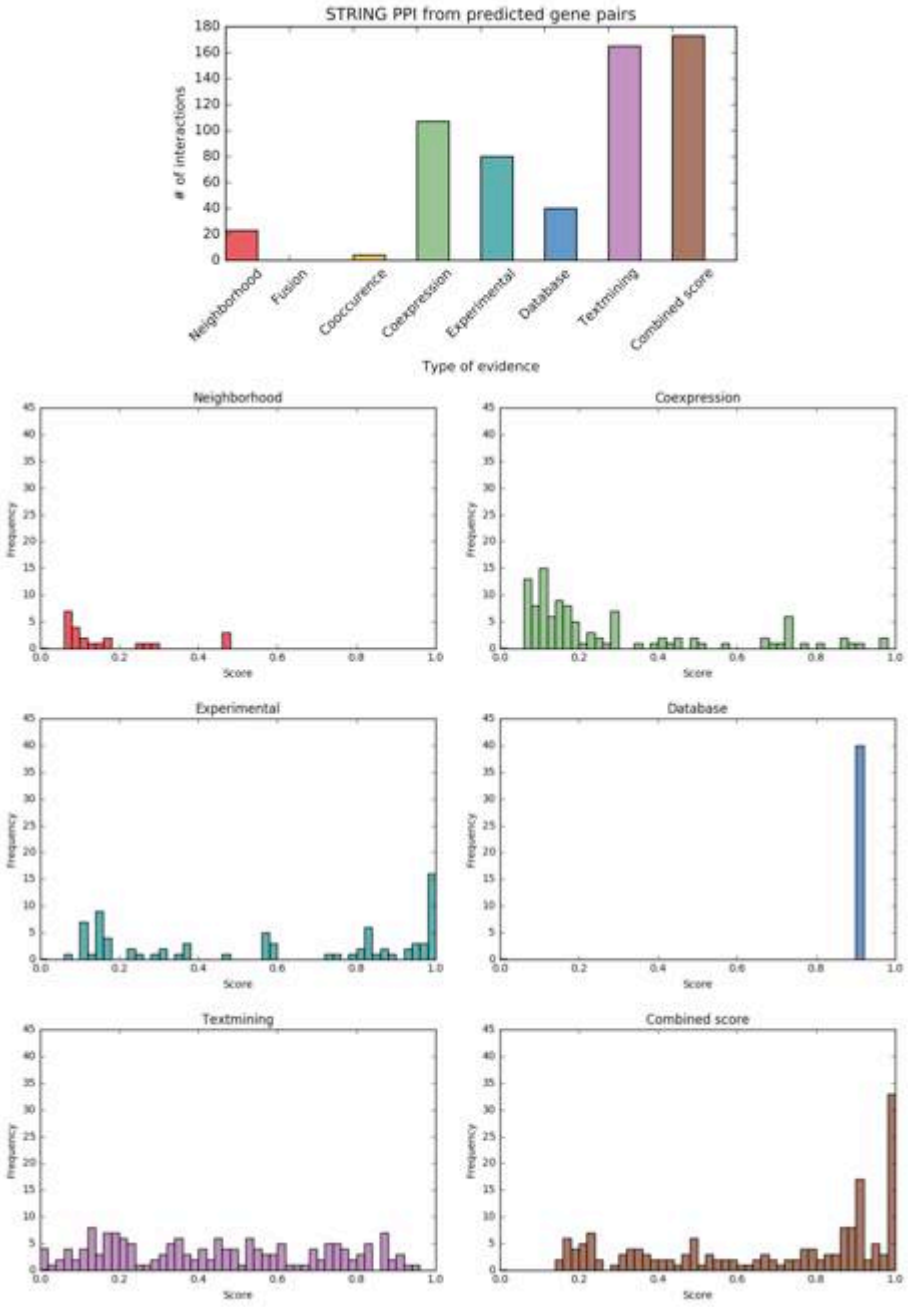

Figure 5.9: representation of summary statistics about the discovered protein-protein interactions. The distribution of the found interactions on the basis of the evidence source is shown on top. For each category of evidence, the distribution of the related scores is reported.

For each category of evidence, STRING assigns a score value, denoting the confidence level about that interaction. If multiple types of evidence are available for the same pair, a combined score is computed from the different scores.

Given the 320 novel gene-gene interactions predicted by the Tri-factorization method, 173 of them were already known as protein-protein interactions in STRING. Even more interestingly, more than half of them (98/173) have a combined score greater than 0.7, which is the level indicated by the STRING curators as high confidence [142].

This means that the associations found by the method are, in general, real interactions and not just random associations.

In addition, the single scores can be analyzed. No occurrences were found from Gene Fusion events, and just a few from Neighborhoods and Co-occurrence events. Many came from text mining, with scores ranging more or less uniformly in the entire interval [0,1]. Many co-expression data were found, even if with generally low scores. Very interestingly, instead, many pairs had an experimental origin, in general with high score, and some interactions came from Database information with very high scores. Figure 5.9 illustrates all the described characteristics. The complete list of scores for each pair can be found in Appendix 2.

5.1.5.5. Co-expression analysis

A further validation analysis was conducted using expression gene data. From the Gene Expression Omnibus data repository [143], the data about four studies were obtained. In particular:

- GSE4619: 55 patients with MDS and 11 controls
- GSE19429: 183 patients with MDS and 17 controls
- GSE58831: 157 patients with MDS and 17 controls
- GSE13159: 73 controls

The raw expression data were normalized using Robust Multichip Average (RMA). After that, for each study and for each of the two categories of samples (cases and controls), the co-expression between all the pairs of genes was computed using the absolute value of the Pearson correlation coefficient. For each gene pair, just two co-expresssion values were considered, one for cases and one for controls. These values were computed by taking the maximum value across all the experiments. Afterward, some comparisons between the values distributions were made. First of all, for each of the two groups, cases and controls, the overall distribution, computed on all the gene pairs, was compared with the distribution obtained by considering only the co-expression values associated to the 323 predicted pairs.

Figure 5.10 shows the results for the cases, while Figure 5.11 shows the distributions for the controls. In seems clear, especially for the cases, that

the global distributions are different from the distributions of the predicted pairs. This highlights the fact that the pairs found by the algorithm are not obtained by chance, but they represent actual interactions.

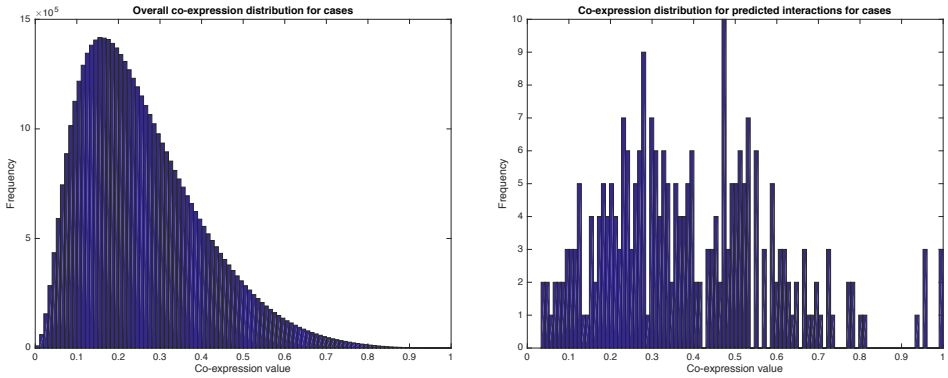

Figure 5.10: distribution of co-expressions for cases. On the left the overall distribution, containing all the co-expression values. On the right, the distribution with only the predicted pairs.

In order to evaluate the difference between the two distributions, two statistical tests were used: the Kolmogorov-Smirnov test and the Mann-Whitney U test. For cases, both of them rejected the null hypothesis of equal distributions with p-values $1.081878 * 10^{-19}$ and $8.961556 * 10^{-28}$ respectively.

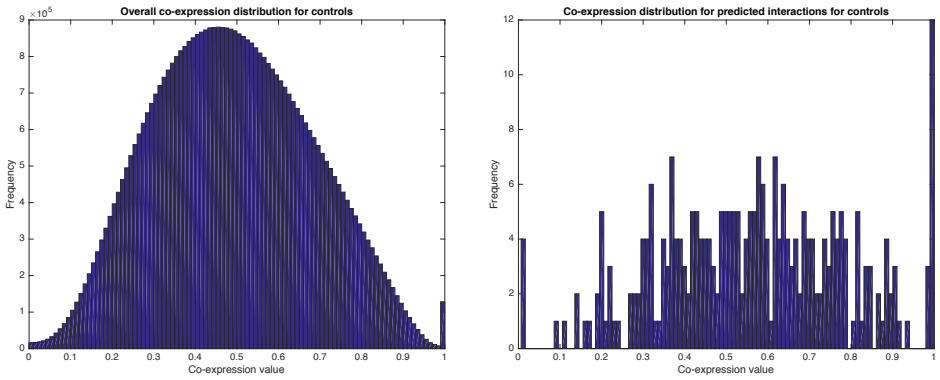

Figure 5.11: distribution of co-expressions for controls. On the left the overall distribution, containing all the co-expression values. On the right, the distribution with only the predicted pairs.

For controls, both of them still rejected the null hypothesis of equal distributions, even if with higher p-values: $5.935023 * 10^{-6}$ for Kolmogorov-Smirnov and $4.410006 * 10^{-8}$ for the Mann-Whitney U test.

This means that the pairs pointed out by the method are relevant for all the samples, even if the p-values suggest a more strict connection with cases. For this reason, another test was carried on, this time to compare cases and controls regarding the related distributions of the predicted pairs. Again, Kolmogorov-Smirnov and the Mann-Whitney U tests were used. Also in this case, the null hypothesis of equal distributions was rejected with p-values $6.676560 * 10^{-11}$ for Kolmogorov-Smirnov and $1.099889e * 10^{-12}$ for the Mann-Whitney U test.

A final test in this direction was conducted. In particular, a Wilcoxon signed rank test was used to evaluate if the differences of the paired co-expression values in cases and controls came from a distribution with zero median. This hypothesis was rejected with a very low p-value, $4.111207 * 10^{-18}$.

All these results suggest that, even if the set of predicted pairs are relevant in all samples (cases and controls) there is still a noteworthy distinction between the two distributions, meaning that most of them are specifically associated to the investigated pathology.

5.1.5.6. Conclusions

The gene-gene interactions resulting from the Tri-factorization algorithm were analyzed under different points of view. Considering the pathway connected to them, many interesting relation with MDS were found. In particular, some known information, not included in the model, was rediscovered using all the other sources of data. Also from the associated protein level, strong evidences of interactions were found. In addition, the co-expression levels of the predicted pairs of genes showed a significant difference with respect to the overall co-expression levels, and between cases and controls. For these reasons, the results are promising and they suggest to further investigating these findings, possibly with an appropriate experimental study.

5.2. Acute myeloid leukemia

The acute myeloid leukemia is one of the five main types of myeloid neoplasms [122]. It is the most common myeloid leukemia, with a prevalence of 3.8 cases per 100,000, deeply increasing to 17.9 cases per 100,000 for people older than 65 years. The median age of onset is 70 years, and it affects more men than women (3:2 proportion) [144]. It is characterized by an increase in the number of myeloid cells in the marrow (blasts), which lose the ability to differentiate normally and arrest their maturation, often leading to hematopoietic insufficiency [145]. This loss leads to fatal infection, bleeding, or organ infiltration, typically, in the absence of treatment, within 1 year of diagnosis. Chemotherapy treatments have been developed, particularly successful for younger adults, but in case

of elderly patients the median survival times are still short, typically only a few months. Genetic defects are believed to play a key role in determining the response to chemotherapy and outcome [144]. In fact, AML is characterized by many genotypic variants, therefore a study focused on this aspect may lead to more accurate classifications and treatments for the pathology.

5.2.1. Problem description

Within this framework, the problem addressed in this work is to build a predictive model, having the survival time as target, and using AML genetic mutations, integrated with some other basic information such as age and gender, as features. In particular, a factorization machine (FM) model (described in Chapter 3) was used. The reason of this choice is related to the fact that AML outcomes are often linked to genetic abnormalities. Therefore, FMs allow to model also interactions between input variables, whose strength can be interpreted on the basis of the values of the related parameters. A classification problem was defined, aimed at identifying patients with high probability of a short survival time, based in particular on mutation data. The classification performance of the algorithm can be seen a measure of the quality of model itself and in particular its ability in capturing the hidden structures inside the data, while the main focus of the work is related to the interpretation of the parameters of the model.

5.2.2. Available data

All the data used for this case study came from the AML cohort (200 patients) hosted by The Cancer Genome Atlas (TCGA) [146]. For this study, the following information was used:

- Somatic mutations. In particular, 4 type of information were used: chromosome, gene, stating position and ending position
- Vital status: dead or alive
- Days to death: number of days of survival (if dead)
- Days to last follow-up
- Age at initial pathological diagnosis
- Gender
- Race

5.2.3. Data preprocessing

Since the objective of the study was to perform a classification based on survival time, first of all a threshold was chosen to discriminate the patients on the basis of the severity of the pathology. In this context, the cases

represent people died before that time threshold, while controls are people still alive or at least survived for a longer time than the threshold. Figure 5.12 shows the histogram of the survival times. To get a balanced dataset, 1 year was chosen as threshold. Some patients had to be discarded, since the time passed from the diagnosis was lesser than 1 year, therefore it was impossible to establish the related survival time. After this step, the number of obtained cases was 80, while the number of controls was 88.

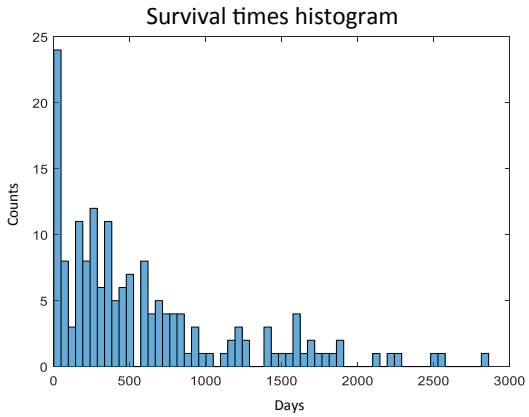

Figure 5.12: distribution of the survival times.

Regarding the somatic mutations, a unique identifier for each of them was obtained by combining 3 data: chromosome, starting position and ending position. After that, we evaluated how many mutations were shared by several patients. Unfortunately, almost all the mutations were unique, making useless their usage in the model. Therefore another strategy was adopted, by taking into account the gene on which the mutations occurred. So, for each patient, the number of mutations on each gene was computed. Some statistics were evaluated on this dataset. As reported in Table 5.3, most of the patients have multiple mutations, in general more than 3.

Table 5.3: Number of mutations per patient.

# Patients	# Mutations
2	0
7	1
5	2
8	3
146	4+

Even if the majority of the patients have at most one mutation per gene, some of them have multiple alterations on the same gene, as indicated in Table 5.4.

Table 5.4: maximum number of mutations on the same gene per patient.

# Patients	Maximum number of mutations on the same gene
2	0
134	1
30	2
2	3
0	4+

The problem is that, as reported in Table 5.5, most of the genes are mutated in just one patient, thus making impossible to share information across the samples. For this reason, a feature selection was performed, to keep just those genes mutated in at least 2 patients. For each of them, the related value corresponds to the number of mutation of that gene in that particular sample.

Table 5.5: number of times the same gene is mutated across all patients.

# Mutated genes	# Occurrences
1408	1
190	2
31	3
39	4+

Regarding the other types of data, two dummy variables were used for both gender (male and female) and race (white and black). In addition, the variable age was included in the model.

5.2.4. Factorization machines setting

For this study, the libFM tool [94] was used. The setting of the FM algorithm requires choosing some parameters. First of all, a learning

method (among SGD, ALS and MCMC) must be selected. Then, the related parameters have to be set. Since the focus of the work was centered on the interpretation of the model's parameters, more than on the classification performance, the input parameters were chosen to maximize the predictive capabilities of the method. These were measured using a 8-fold cross validation, where the number 8 was chosen in order to have at least 20 samples in each test set. Different combinations of the input parameters were tested, measuring the classification performance in terms of AUC (Area Under the ROC Curve), balanced accuracy, sensitivity and specificity. The best set of parameters for each of the three learning algorithms is reported in Table 5.6.

Table 5.6: best set of parameters for each of the three algorithms.

Method	Rank k	Std σ	# Iterations	Reg params λ	Learning rate η
MCMC	2	0.01	500	NA	NA
ALS	6	0.001	1,000	0.01	NA
SGD	12	0.001	12	0.01	0.0003

The best performance, under all the points of view, was obtained by the MCMC method (Table 5.7), which was thus chosen for the following analysis.

Table 5.7: classification performance for the best configuration of the three methods. AUC with its 95% confidence intervals, balanced accuracy, sensitivity and specificity are reported.

Method	AUC	AUC CI		Balanced acc	Sensitivity	Specificity
MCMC	0.67386	0.60856	0.73917	0.65398	0.6375	0.67045
ALS	0.60455	0.56228	0.64681	0.58125	0.5375	0.625
SGD	0.57273	0.4538	0.69165	0.52216	0.4875	0.55682

An important role for MCMC is played by the number of iterations. Figure 3.13 shows the trend of the AUC values depending on the number of iterations of the algorithm. It is clear that at least 500 iterations are needed to reach the convergence. To avoid the burn-in phase, just the last 100 out of the 500 iterations were considered. The libFM tool was modified in order to output all the parameters extracted from each single iteration. In this way, for each parameter the entire distribution (once reached convergence) was available for the subsequent analysis.

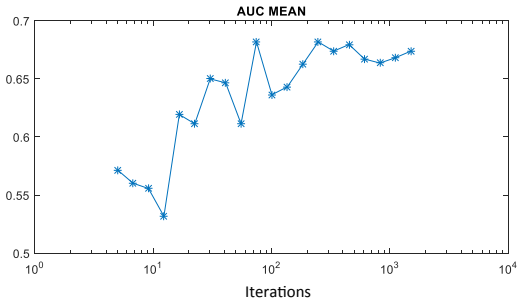

Figure 5.13: AUC values depending on the number of iterations of MCMC algorithm.

5.2.5. Results

The post-processing phase was focused on the analysis of the predicted interactions between the mutated genes. Thanks to the interpretability property of the FMs model, the low dimensional vectors associated to each input features can be exploited to compute an interaction value between all pairs of variables. In this case, only the vectors related to mutated genes where considered. Through the dot product between each pair of vectors, the correlated interaction value was computed, one for each of the 100 iterations kept after the burn-in phase. Therefore, at the end of this process, for each pair of genes, the related distribution of the interaction score was available. For every distribution, a t-test was used to evaluate if the interaction differed from 0 in a significant way. For the test, the significance threshold for the p-value was set to $\alpha = 10^{-5}$. In this way, a set of possible interesting associations was obtained.

Due to the random initialization step, different runs of the algorithm yielded quite different lists of interesting associations, with a typical overlap of about 20%. For this reason, multiple runs of the algorithm were performed. In particular, 1,000 repetitions were carried out, resulting in 1,000 different lists of interesting associations. Then, to each interaction a score value was assigned, based on the number of times (out of 1,000) that particular pair resulted to be significant based on the t-test. Figure 5.14 shows the distribution of these scores. It is immediately cleat that just a few of the overall interactions are characterized by high score values. Therefore, only the most recurring ones were selected. This was obtained by keeping only the pairs with a score higher than the 99-th percentile of the distribution of all the scores.

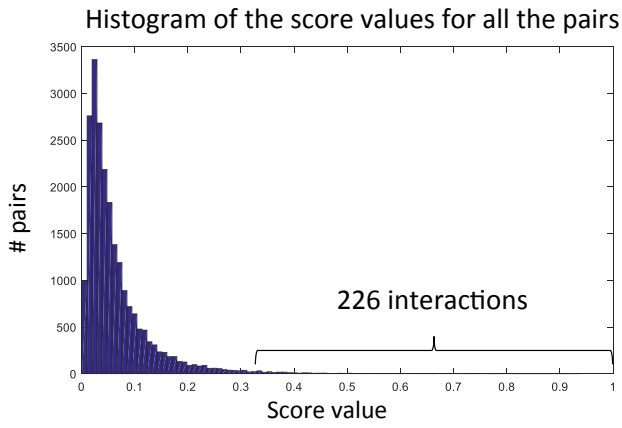

Figure 5.14: histogram of the score values for all the gene pairs. The interaction over the 99-th percentile are pointed out.

After this step, only 226 interactions survived for the subsequent steps. For all of them, the related protein-protein interactions were considered. In particular, the combined score from STRING was computed. The complete set of interactions with the associated STRING score is reported in Appendix 3. The overall mean value of all the scores is 0.4484, which falls into the interval of medium confidence defined by STRING curators [142].

5.2.5.1. Literature analysis

In order to evaluate the results, a literature analysis was conducted using PubMed. It is a search engine for publications focused on biomedical topics. All the articles containing in their abstract or text one on the pairs predicted by the algorithm were extracted. To filter out the results, only those publications with at least a MeSH term related to AML were kept. For 12 of the predicted pairs, an article at least was found (Table 5.8).

Table 5.8: gene pairs found in PubMed in association with AML and related STRING combined score.

Gene1	Gene2	PPI combined score
DNMT3A	IDH2	0.606000
NRAS	TP53	0.536000
RUNX1	TP53	0.518000
CEBPA	FLT3	0.487000
CEBPA	RUNX1	0.481000
FLT3	RUNX1	0.471000
FLT3	NRAS	0.456000
IDH1	TP53	0.412000
KIT	NRAS	0.363000

KRAS	NRAS	0.358000
KIT	TP53	0.357000
PHF6	PTPN11	0.352000

The related publications were then analyzed to find experimental support about the interaction of the associated pairs. For some of the couples of genes no studies were found focused on the synergic effect of multiple mutations on survival. The fact that more genes are co-cited in the same paper may relate to the fact that the patients had one of those mutations, but not both.

Anyway, interesting results were found for some of the reported pairs. For example, NRAS-TP53 and FLT-NRAS are clinically very relevant, since RAS, FLT3, and TP53 have important interactions in AML, and therefore examining these genes in the same cohort of patients may provide information about patterns of genetic disruption of the pathology [147].

Another interesting association is CEBPA-RUNX1, since it seems that a mutation on one of them can influence the expression of the other, interfering with the cell differentiation [148].

In other cases, the direct effects of paired mutations were evaluated. This is the case of RUNX1 and TP53, for which co-occurring mutations are considered as significant for the development of the AML [149].

In other cases, additive effects were investigated for paired mutations of genes (DNMT3A-IDH2 [150], CEBPA-FLT3 [151,152]) and validated on in vivo on mice in case of FLT3-RUNX1 [153,154]

These results suggest the validity of the approach used in this study, which may represent a promising starting point for more focused and targeted experiments to evaluate the synergic effects of paired mutations on genes.

Chapter 6

Conclusions and future works

Data integration represents one of the most interesting challenges for digital medicine. The availability of increasingly larger amount of data is pushing researchers and companies to investigate novel techniques to extract useful information by combining data coming from different sources, with the aim of performing prediction, interpolation and analogical reasoning. The characteristics of biomedical data make this operation really challenging, since very heterogeneous information is available.

In this dissertation, some solutions to the data integration problem were discussed. In particular, the attention was focused on a specific class of methods, characterized by the exploitation of matrix factorization techniques. The mathematical properties of these methods allow bringing out intrinsic characteristics of the data. In fact, by operating a dimensionality reduction, the latent structures of an input matrix can be revealed by projecting the data into a low dimensional space.

This property can be used in a machine learning framework in order to capture the interaction effects between variables. The Factorization Machines model utilizes this strategy to perform classification and regression. An application of this method has been described in this work. The input data came from a public data repository (TCGA) and contained information about patients affected by acute myeloid leukemia (AML). In particular, mutation data, in addition to some other personal information (age, gender, race) were used by the method. The algorithm was trained, using the severity of the disease, determined on the basis of the survival time, as target class. The algorithm was tuned in order to maximize the classification performance, while the analysis of the results was focused on the interpretation of the model's parameters. In fact, due to the usage of a matrix decomposition technique, the parameters of the model can provide insights about the strength of the pairwise associations between the input variables. In particular, for the considered case study, the objective was to evaluate possible synergic effects of paired mutations on different genes,

with respect to the pathology under investigation. Interestingly, some of the pairs of mutated genes pointed out by the method were already known in literature to play an important role in the development of the disease. For some of them in particular, combined effects had already been suggested.

In case of multiple heterogeneous data, matrix factorization techniques can play an important role to simultaneously fuse all the information in a joint model. All the input data have to be expressed in form of relation matrices, each of them modeling the pairwise interactions between two types of objects. If some of these objects are shared by different matrices, a joint decomposition of all the matrices allows summarizing the information associated to each element in a low dimensional vector, expressing its characteristics in a latent features space. Two different approaches have been described in this thesis, the Tri-factorization method and the Bayesian probabilistic matrix factorization method. While the first one is an already published technique, the second one has been developed in the context of this thesis. In addition, both of them have been implemented from scratch.

Regarding the Tri-factorization method, in this dissertation an application of the algorithm has been presented. The starting data employed for this case study came from a local hospital, and consisted of mutation data of people affected by myelodysplastic syndromes (MDS). In this case, the objective of the study was to identify new gene-gene interactions characterizing the pathology, in order to better understand its underlying molecular mechanisms. Other kinds of data were retrieved from public datasets, concerning different types of objects: genes, mutations, pathways and diseases. All of them were modeled using appropriate matrices and pre-processed to be used by the Tri-factorization algorithm. After an appropriate tuning of the parameters, the method was applied to the input data. The result was a list of potentially interesting gene-gene interactions, which were analyzed from different points of view. The pathways related to those pairs revealed interesting relations with MDS. In particular, some known information, not included in the model, was rediscovered using all the other sources of data. Moreover, moving to proteins, strong evidences of interactions were found. In addition, a co-expression analysis on MDS patients showed a significant correlation of the overall set of found interactions with respect to the disease.

In conclusion, this work shows how, thanks to the application of factorization-based methods, it is possible to effectively exploit the peculiar characteristic of some types of biomedical data in order to extract useful information. Of course, a deeper validation step is needed to better understand the quality of the results and the direction to take to improve the performance. In particular, targeted experiments would be important to establish the validity of the discoveries. Anyway, under this perspective, the presented methods may be considered as a reliable data driven strategy for the definition of new research hypothesis.

Appendix 1

Example of application of the Bayesian factorization for data fusion

In this appendix, an application to real data of the Bayesian matrix factorization method for data fusion, described in Chapter 4, is presented. The data sourced employed for the analysis were gene expression data and miRNA expression data coming from The Cancer Genome Atlas (TCGA) [155]. For each gene and miRNA, the associated expressions valued were rescaled in the [0,1] interval. A third matrix was built using gene-miRNA interactions coming from MirTarBase [156]. Only the experimentally validated interactions with strong evidence were kept in the analysis. For computational reasons, the number of genes was filtered, keeping just those with at least one known strong gene-miRNA interaction in MirTarBase. At the end of the pre-processing phase, the final data consisted of 183 patients, 760 miRNAs and 2197 genes, thus defining the dimensions of the three matrices. The target of this experiment was the gene-miRNA interaction matrix, which was very sparse, with a density of 0.003. To evaluate the performances of the method, 10% of the entries from the target matrix were selected and used as test set. The known interactions inside this set were then put to zero to create the input matrix. The number of iterations was set to 1000, using the last 500 for the reconstruction of the target matrix. Specificity, sensitivity and the Matthews correlation coefficient (MCC) were chosen to evaluate the performance. In order to compute them, the resulting matrix was binary discretized using a 0.5 cutoff.

The results, obtained with different configurations parameters are shown in Table A1.1. A grid search was performed, using two different values of the rank parameter k (10 and 50), four different values for the parameter α_1 (1,10,100,1000), while we set the value of α_2 as one and two orders of magnitude lesser than α_1. The results show that the specificity is always very high, thus reducing the number of false positive for this very imbalanced problem. On the contrary, the specificity is highly dependent on the values of the parameters. In particular, low values of α_1 tend to reconstruct the input matrix, reducing the possibility of highlighting

unknown interactions, while very high values may introduce too much uncertainty. For k =50 the results were always better than with k =10, at the price of a much slower execution.

To get a comparison, on the same dataset was applied the Tri-factorization algorithm described in Chapter 4. Due to the random initialization, 15 repetitions were performed. The threshold for the sopping criterion was set to 10^-5, but in any case a maximum of 1000 iterations of the optimization algorithm were allowed. The rank parameters were set for each type of object by scaling the number of the known interactions with a factor 200. A consensus matrix was then obtained by averaging the 15 output matrices of each repetition. On this matrix the same statistics computed for the Bayesian method were evaluated. The results show a very high specificity (0.99999) but a very low sensitivity (0.0451), with MCC= 0.2079. Therefore, it seems that the Bayesian method is more flexible, allowing a greater control thanks to its parameters.

Table A1.1: Results for different settings of the parameters

α_1	α_2	k	Specificity	Sensitivity	MCC
1000	10	50	0.9845	0.7130	0.3036
1000	100	50	0.9974	0.2708	0.2608
1000	10	10	0.9503	0.4838	0.1133
1000	100	10	0.9956	0.0794	0.0633
100	1	50	0.9879	0.8087	0.3801
100	10	50	0.9979	0.2419	0.2553
100	1	10	0.9455	0.5794	0.1310
100	10	10	0.9949	0.1354	0.1008
10	0.1	50	0.9417	0.6931	0.1532
10	1	50	0.9962	0.0794	0.0681
10	0.1	10	0.9409	0.5921	0.1282
10	1	10	0.9960	0.0614	0.0511
1	0.01	50	0.9323	0.6372	0.1287
1	0.1	50	0.9964	0.0199	0.0155
1	0.01	10	0.9424	0.4946	0.1067
1	0.1	10	0.9963	0.0199	0.0152

Since the Bayesian method models the data using probabilistic distributions, it is also possible to compute an estimate of the uncertainty for each pairwise interaction. Figure A1.1. shows an example of two different posterior distributions, both of them denoting the presence of an association due to a distribution mean greater than 0.5. However, while the figure on the left reveals a high uncertainty related to that pair, because the related distribution is very broad, the figure on the right is characterized by a very tight distribution, suggesting a lower uncertainty for that interaction.

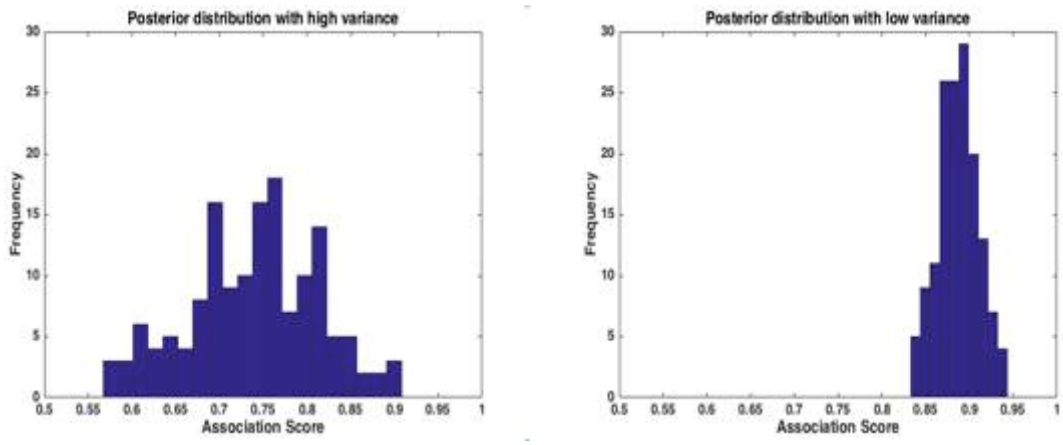

Figure A1.1: Example of two different posterior distributions. On the left there is an example of predicted association characterized by high uncertainty, on the right an example of predicted association characterized by low uncertainty

Appendix 2

Additional results for MDS case study

All the 320 new gene-gene interactions discovered by the Tri-factorization algorithm and their associated combined score (if present) from STRING are reported in Table A2.1

Table A2.1: list of all the 320 new gene-gene interactions their associated combined score (if present) from STRING. The pairs characterized by high significance (combined score>0.7) are marked in light blue.

Gene1	Gene2	PPI combined score
DARS	LARS	0.999
DARS	EPRS	0.999
DARS	MARS	0.999
RFC4	RFC5	0.999
RFC4	RFC3	0.999
EPRS	LARS	0.999
EPRS	MARS	0.999
SNRPD2	SNRNP70	0.999
ATP6V1A	ATP6V1E1	0.999
ATP6V1A	ATP6V1E2	0.999
NHP2	NOP10	0.999
EXOSC1	EXOSC4	0.999
LARS	MARS	0.999
RFC5	RFC3	0.999
POLD1	POLD2	0.999
POLD1	POLD3	0.999
POLD2	POLD3	0.999
UGT1A4	UGT1A7	0.999
UGT1A4	UGT1A3	0.999
UGT1A7	UGT1A3	0.999
RPA1	RPA3	0.999
SNRPD2	CDC5L	0.998
POLE4	POLE	0.997
POLD1	POLD4	0.996
MPHOSPH10	TBL3	0.996
POLD2	POLD4	0.995
POLD3	POLD4	0.993
RPL9	MRPS5	0.990

ATP6V1H	ATP6V0B	0.989
WDR75	NAT10	0.988
ATP6V1D	ATP6V1G1	0.984
ATP6V1E1	ATP6V0E1	0.981
ATP6V1D	ATP6V1G3	0.980
BUD31	CDC5L	0.978
DHX8	BUD31	0.964
ATP6V1E2	ATP6V0E1	0.964
B3GAT3	CSGALNACT2	0.959
RPP40	RPP30	0.955
ATP6V1A	ATP6V0E1	0.947
RFC1	POLD1	0.945
RFC1	RFC2	0.941
GNL3	NAT10	0.933
DHX15	DDX42	0.929
ISY1	CDC5L	0.919
RFC1	POLD2	0.918
CHSY1	CSGALNACT2	0.915
NARS	YARS	0.915
SF3A3	HNRNPU	0.915
PRPF8	SNRPB2	0.914
RRP7A	PWP2	0.913
ATP6V1G2	ATP6V1E1	0.912
TARSL2	YARS	0.911
RFC1	POLD3	0.910
CLDN18	CLDN20	0.907
TCIRG1	ATP6V0A1	0.905
ATP6V1G3	ATP6V1G1	0.903
SNRNP40	RBMX	0.902
RFC1	POLD4	0.900
LARS	SARS	0.900
ATP6V1E1	ATP6V1E2	0.900
DHX8	CDC5L	0.899
EPRS	AARS2	0.898
ALG8	ALG5	0.898
BUD31	ISY1	0.894
MARS	SARS	0.888
SNRPD2	BUD31	0.887
MARS	WARS2	0.886
PPP3R2	PPP3CC	0.880
EPRS	DARS2	0.879
MARS	AARS2	0.876
GNL3	WDR75	0.875
YARS	HARS	0.870
LARS	AARS2	0.870
PWP2	RBM28	0.869
NARS2	YARS	0.865
PRPF6	ZMAT2	0.861
TARSL2	HARS	0.853
DHX8	ISY1	0.853
PPIL1	BUD31	0.847
TARSL2	NARS	0.836
HSD17B12	TECR	0.836
NARS	HARS	0.835
DARS	SARS	0.818
EPRS	SARS	0.815
DHX15	PRPF18	0.792

LARS	WARS2	0.792
ATP6V1G2	ATP6V1E2	0.781
DARS2	AARS2	0.780
SARS	WARS2	0.776
WARS	YARS2	0.764
B3GAT3	XYLT2	0.761
CERS5	SPTLC2	0.760
PLA2G12A	PLA2G2D	0.752
EPRS	WARS2	0.745
ATP6V1A	ATP6V1G2	0.736
VARS	TARS	0.736
BMS1	RIOK1	0.716
LARS	DARS2	0.700
DHX8	SNRNP70	0.691
RARS	TARS2	0.676
SARS	SEPSECS	0.668
DARS2	MARS	0.667
YARS	MTFMT	0.654
EXOSC1	CNOT3	0.652
DARS2	SARS	0.623
DARS2	WARS2	0.600
SNRNP70	CDC5L	0.593
DHX15	SF3A3	0.584
SARS	AARS2	0.577
NHP2	BMS1	0.575
RFC2	POLD1	0.550
NDST1	XYLT2	0.544
XAB2	AQR	0.538
MTMR1	INPP5B	0.534
CSGALNACT2	XYLT2	0.520
HNRNPC	SLU7	0.516
ALG3	DDOST	0.498
DARS	AARS2	0.496
VARS2	LARS2	0.495
CHSY1	XYLT2	0.490
SF3A3	PRPF18	0.489
DARS	WARS2	0.484
DDX42	SF3A3	0.475
PLA2G12A	PLA2G2E	0.472
EXTL3	DSE	0.465
RFC2	POLD2	0.455
CHSY1	B3GAT3	0.434
B3GAT3	NDST1	0.420
NARS2	HARS	0.400
PLA2G12A	PLA2G2F	0.400
DARS	DARS2	0.393
PARS2	VARS	0.388
PARS2	TARS	0.375
BUD31	SNRNP70	0.375
RFC5	XPA	0.372
SNRNP70	ISY1	0.358
RFC2	POLD4	0.353
EXTL2	CHST11	0.352
TARSL2	NARS2	0.345
NDST2	B3GALT6	0.337
NARS2	NARS	0.327
RFC2	POLD3	0.325
POP4	RPP25L	0.323
RFC4	XPA	0.318

TARSL2	MTFMT	0.307
RRP7A	RBM28	0.304
RFC3	XPA	0.281
FARSB	EARS2	0.259
PPIL1	CDC5L	0.250
CETN2	XPA	0.234
SNRPD2	ISY1	0.232
ALDH3A1	ALDH1A3	0.232
DHX8	SNRPD2	0.231
RBM28	LOC81691	0.222
HARS	MTFMT	0.222
RETN	CDKAL1	0.221
DDX42	PRPF18	0.217
NHP2	RIOK1	0.217
PPIL1	ISY1	0.210
CREB3L2	CREB3L4	0.202
DCP1B	EXOSC5	0.200
CREB3L1	CREB3L4	0.198
NUDT7	PEX5L	0.186
PWP2	LOC81691	0.183
ATP6V1G2	ATP6V0E1	0.183
NDST1	CSGALNACT2	0.172
PPIL1	DHX8	0.168
PPIL1	SNRPD2	0.168
CHSY1	NDST1	0.165
DDX42	HNRNPU	0.165
PPIL1	SNRNP70	0.162
RRP7A	LOC81691	0.158
DECR2	NUDT7	0.150
AARS2	SEPSECS	Not in STRING
ALDH1A3	ALDH3B1	Not in STRING
ALDH3A1	ALDH3B1	Not in STRING
BMS1	NOP10	Not in STRING
BTG1	CNOT3	Not in STRING
CALM1	CALML3	Not in STRING
CALM1	CALML5	Not in STRING
CALM2	CALML3	Not in STRING
CALM2	CALM3	Not in STRING
CALM2	CALML5	Not in STRING
CALM2	CALM1	Not in STRING
CALM3	CALML3	Not in STRING
CALM3	CALML5	Not in STRING
CALM3	CALM1	Not in STRING
CALML5	CALML3	Not in STRING
CETN2	RFC5	Not in STRING
CETN2	RFC3	Not in STRING
CREB3	CREB3L2	Not in STRING
CREB3	CREB3L1	Not in STRING
CREB3	CREB3L4	Not in STRING
CREB3L2	CREB3L1	Not in STRING
DARS	TRNM	Not in STRING
DARS	SEPSECS	Not in STRING
DARS	TRNR	Not in STRING
DARS2	SEPSECS	Not in STRING
DARS2	TRNR	Not in STRING
DDX42	RP9	Not in STRING
DDX42	SNRNP27	Not in STRING
DECR2	PEX5L	Not in STRING

DHX15	RP9	Not in STRING
DHX15	HNRNPU	Not in STRING
DHX15	SNRNP27	Not in STRING
DOLPP1	B4GALT3	Not in STRING
EARS2	TRNF	Not in STRING
EDC3	EXOSC7	Not in STRING
EPRS	TRNM	Not in STRING
EPRS	SEPSECS	Not in STRING
EPRS	TRNR	Not in STRING
EXOSC1	BTG1	Not in STRING
EXOSC4	CNOT3	Not in STRING
EXOSC4	BTG1	Not in STRING
FARSB	TRNF	Not in STRING
FAU	MRPS18A	Not in STRING
HARS2	TRND	Not in STRING
HNRNPU	RP9	Not in STRING
HNRNPU	PRPF18	Not in STRING
JMJD7-PLA2G4B	PLA2G4C	Not in STRING
LARS	TRNM	Not in STRING
LARS	SEPSECS	Not in STRING
LARS	TRNR	Not in STRING
MARS	SEPSECS	Not in STRING
NARS	MTFMT	Not in STRING
NARS	TRNI	Not in STRING
NARS2	MTFMT	Not in STRING
NARS2	TRNI	Not in STRING
PABPC4L	PABPC1L2B	Not in STRING
PAOX	PHYH	Not in STRING
PARN	DDX6	Not in STRING
PARS2	TRNP	Not in STRING
PLA2G2C	PLA2G10	Not in STRING
PLA2G2D	PLA2G2F	Not in STRING
PLA2G2E	PLA2G2D	Not in STRING
PLA2G2E	PLA2G2F	Not in STRING
PLA2G4B	PLA2G4D	Not in STRING
POP4	SNORD3A	Not in STRING
POP4	SNORD3C	Not in STRING
PRPF18	RP9	Not in STRING
PSTK	TRNS1	Not in STRING
PSTK	LARS2	Not in STRING
PSTK	VARS2	Not in STRING
PXMP4	ACOT8	Not in STRING
RARS	TRNS2	Not in STRING
RARS	TRNL2	Not in STRING
RFC4	CETN2	Not in STRING
RIOK1	NOP10	Not in STRING
RPP25L	SNORD3A	Not in STRING
RPP25L	SNORD3C	Not in STRING
RRP7A	TCOF1	Not in STRING
SF3A3	RP9	Not in STRING
SNORD3A	SNORD3C	Not in STRING
SNORD3B-1	RPP25L	Not in STRING
SNORD3B-1	SNORD3A	Not in STRING
SNORD3B-1	POP4	Not in STRING
SNORD3B-1	SNORD3C	Not in STRING
SNORD3B-1	SNORD3B-2	Not in STRING
SNORD3B-2	RPP25L	Not in STRING
SNORD3B-2	POP4	Not in STRING

SNORD3B-2	SNORD3C	Not in STRING
SNORD3B-2	SNORD3A	Not in STRING
SNRNP27	RP9	Not in STRING
SNRNP27	PRPF18	Not in STRING
SNRNP27	HNRNPU	Not in STRING
SNRNP27	SF3A3	Not in STRING
SNRPA	PRPF38B	Not in STRING
SYF2	SNRPF	Not in STRING
TARS2	TRNS2	Not in STRING
TARS2	TRNL2	Not in STRING
TARSL2	TRNI	Not in STRING
TCOF1	PWP2	Not in STRING
TCOF1	LOC81691	Not in STRING
TCOF1	RBM28	Not in STRING
TRNA	TRNY	Not in STRING
TRNA	IARS	Not in STRING
TRNI	MTFMT	Not in STRING
TRNI	HARS	Not in STRING
TRNM	WARS2	Not in STRING
TRNM	DARS2	Not in STRING
TRNM	SEPSECS	Not in STRING
TRNM	MARS	Not in STRING
TRNM	SARS	Not in STRING
TRNM	TRNR	Not in STRING
TRNM	AARS2	Not in STRING
TRNN	TRNQ	Not in STRING
TRNP	TARS	Not in STRING
TRNP	VARS	Not in STRING
TRNR	WARS2	Not in STRING
TRNR	SEPSECS	Not in STRING
TRNR	MARS	Not in STRING
TRNR	SARS	Not in STRING
TRNR	AARS2	Not in STRING
TRNS1	VARS2	Not in STRING
TRNS1	LARS2	Not in STRING
TRNS2	TRNL2	Not in STRING
TRNY	IARS	Not in STRING
UGT1A4	UGT1A5	Not in STRING
UGT1A5	UGT1A7	Not in STRING
UGT1A5	UGT1A3	Not in STRING
UGT2A1	UGT2B11	Not in STRING
UGT2A1	UGT1A4	Not in STRING
UGT2A1	UGT1A5	Not in STRING
UGT2A1	UGT1A7	Not in STRING
UGT2A1	UGT1A3	Not in STRING
UGT2A3	UGT2A1	Not in STRING
UGT2A3	UGT2B11	Not in STRING
UGT2A3	UGT1A4	Not in STRING
UGT2A3	UGT1A5	Not in STRING
UGT2A3	UGT1A7	Not in STRING
UGT2A3	UGT1A3	Not in STRING
UGT2B11	UGT1A4	Not in STRING
UGT2B11	UGT1A5	Not in STRING
UGT2B11	UGT1A7	Not in STRING
UGT2B11	UGT1A3	Not in STRING
UGT2B28	UGT1A9	Not in STRING
WARS2	SEPSECS	Not in STRING
WARS2	AARS2	Not in STRING
WBP11	U2AF1L5	Not in STRING

YARS	TRNI	Not in STRING

Appendix 3

Additional results for AML case study

All the 226 gene-gene interactions discovered by the FM model and their associated combined score from STRING are reported in Table A3.1

Table A3.1: list of all the 226 gene-gene interactions their associated combined score from STRING. The pairs associated to mesh terms related to leukemia are marked in light blue.

Gene1	Gene2	PPI combined score
DNMT3A	NPM1	0.941000
DNMT3A	NRAS	0.806000
CEBPA	TET2	0.797000
DNMT3A	TET2	0.759000
DNMT3A	FLT3	0.751000
CEBPA	DNMT3A	0.724000
FLT3	NPM1	0.716000
BCOR	DNMT3A	0.701000
DNMT3A	MT-CO2	0.695000
DNMT3A	MTUS2	0.678000
DNMT3A	MEGF8	0.676000
DNMT3A	PTCH1	0.675000
DNMT3A	TP53	0.651000
DNMT3A	RAD21	0.642000
NPM1	TET2	0.640000
NPM1	TP53	0.637000
CEBPA	NRAS	0.624000
DNMT3A	TET1	0.621000
DNMT3A	RUNX1	0.619000
FLT3	TP53	0.618000
CRISPLD1	DNMT3A	0.614000
NRAS	TET2	0.610000
NPM1	NRAS	0.608000
MT-CO2	NRAS	0.607000
DNMT3A	IDH2	0.606000
CEBPA	MT-CO2	0.591000
CEBPA	NPM1	0.584000
IDH1	NPM1	0.581000

MT-CYB	NPM1	0.576000
DNMT3A	TTN	0.574000
NPM1	RUNX1	0.563000
NRAS	TTN	0.562000
MT-CO2	NPM1	0.552000
TET2	TP53	0.547000
DNMT3A	MAGI2	0.542000
DNMT3A	PTPRN	0.542000
FLT3	TET2	0.538000
NRAS	TP53	0.536000
C10orf28	DNMT3A	0.533000
NPM1	RAD21	0.525000
CEBPA	TP53	0.522000
MT-CO2	TET2	0.522000
RUNX1	TP53	0.518000
MTUS2	NRAS	0.516000
BCOR	NPM1	0.515000
DNMT3A	IDH1	0.515000
CEBPA	IDH2	0.510000
DNMT3A	LOC100130211	0.505000
DNMT3A	PTPRT	0.505000
CBL	NPM1	0.504000
CEBPA	TTN	0.504000
IDH2	NPM1	0.504000
NPM1	TET1	0.503000
IDH2	NRAS	0.501000
IDH2	TET2	0.498000
MEGF8	NPM1	0.497000
DNMT3A	PHF6	0.494000
DNMT3A	FAM57B	0.493000
RUNX1	TET2	0.493000
CEBPA	FLT3	0.487000
MTUS2	NPM1	0.485000
FAM57B	NRAS	0.484000
NPM1	PTCH1	0.484000
NRAS	RUNX1	0.484000
CEBPA	RUNX1	0.481000
DNMT3A	KIT	0.480000
NRAS	RAD21	0.479000
FLT3	IDH1	0.478000
CBL	DNMT3A	0.475000
IDH1	KCNT1	0.474000
FLT3	RUNX1	0.471000
NPM1	PHF6	0.465000
KRT79	NPM1	0.464000
NRAS	PTCH1	0.463000
DNMT3A	KRT79	0.460000
DNMT3A	MT-CYB	0.459000
IDH2	TP53	0.459000
MEGF8	NRAS	0.457000
FLT3	NRAS	0.456000
KIT	NPM1	0.456000
CEBPA	ENSG00000211459	0.455000
LOC100130211	NPM1	0.453000
DNMT3A	PTPN11	0.452000
PHF6	TET1	0.452000
TET2	TTN	0.451000
PTPRT	TTN	0.449000
CRISPLD1	NRAS	0.447000

IDH1	TET2	0.446000
DSCAM	PTPRT	0.444000
MT-CO2	TP53	0.441000
CROCC	DNMT3A	0.437000
MT-CO2	TTN	0.437000
BCOR	NRAS	0.436000
DNMT3A	LOC152845	0.436000
DSCAM	TTN	0.435000
BCOR	FLT3	0.433000
DNMT3A	U2AF1	0.433000
DNMT3A	FAM47A	0.432000
IDH1	IDH2	0.427000
IDH2	RUNX1	0.427000
NRAS	PTPRT	0.427000
DST	KIT	0.426000
CEBPA	MTUS2	0.425000
DNMT3A	FLG	0.425000
FLT3	IDH2	0.423000
DNMT3A	NTRK3	0.422000
DNMT3A	WT1	0.422000
DNMT3A	DSCAM	0.421000
DNMT3A	KRAS	0.421000
IDH2	TTN	0.421000
DNMT3A	TCEAL3	0.420000
DNMT3A	LOC730032	0.419000
DNMT3A	PLCE1	0.419000
NPM1	TTN	0.418000
MT-CO2	RUNX1	0.414000
MTUS2	TET2	0.414000
CEBPA	TCEAL3	0.412000
IDH1	TP53	0.412000
IDH1	NRAS	0.411000
KIT	TET2	0.410000
MTUS2	TTN	0.407000
SCML2	TET2	0.404000
CEBPA	IDH1	0.403000
CEBPA	FAM57B	0.401000
DNMT3A	PRPF4B	0.400000
CEBPA	MEGF8	0.399000
CEBPA	CRISPLD1	0.398000
MEGF8	TET2	0.398000
DNMT3A	NMUR2	0.397000
DNMT3A	SEMA4A	0.397000
NRAS	PTPRN	0.397000
DNMT3A	KCNK13	0.395000
DNMT3A	MIR142	0.394000
DNMT3A	SPEN	0.394000
NPM1	PTPN11	0.393000
DNMT3A	PCDHA13	0.392000
KRAS	TET2	0.392000
DNMT3A	PCDHB18	0.391000
DNMT3A	STAG2	0.391000
DST	TET2	0.391000
IDH2	MT-CO2	0.390000
IDH2	MTUS2	0.390000
NRAS	PLCE1	0.386000
FLT3	MT-CO2	0.385000
NPM1	PTPRN	0.383000
FLT3	KRT79	0.382000

CEBPA	PTCH1	0.381000
CEBPA	WT1	0.381000
MT-CO2	MTUS2	0.381000
C10orf28	MAGI2	0.379000
FLT3	KIT	0.379000
MAGI2	NPM1	0.379000
MTUS2	TP53	0.379000
BCOR	TET2	0.378000
DNMT3A	GBP4	0.378000
PKD1L2	TET2	0.378000
SMC3	TTN	0.378000
CROCC	TP53	0.377000
DNMT3A	SYT15	0.377000
NRAS	NTRK3	0.377000
MTUS2	RUNX1	0.376000
NRAS	TCEAL3	0.376000
FAM57B	IDH2	0.375000
IDH2	KRAS	0.375000
PTCH1	TET2	0.374000
RUNX1	TTN	0.374000
CEBPA	SMC3	0.373000
KIT	WT1	0.373000
DNMT3A	OR4H12P	0.372000
CEBPA	KIT	0.371000
FAM57B	TET2	0.371000
NPM1	U2AF1	0.371000
NPM1	WT1	0.371000
TP53	TTN	0.371000
C10orf28	NPM1	0.370000
CEBPA	KRAS	0.369000
CEBPA	PTPRT	0.369000
NRAS	SMC3	0.369000
MT-CO2	PTCH1	0.367000
NRAS	PTPN11	0.367000
CHD4	DNMT3A	0.366000
C10orf28	NRAS	0.365000
DSCAM	NRAS	0.365000
DNMT3A	MED12	0.363000
KIT	NRAS	0.363000
FLT3	PHF6	0.361000
FLT3	TET1	0.361000
MEGF8	MT-CO2	0.361000
NRAS	STAG2	0.361000
CRISPLD1	NPM1	0.360000
FLT3	RAD21	0.359000
HYDIN	NRAS	0.359000
CEBPA	RAD21	0.358000
FCGBP	TET2	0.358000
KRAS	NRAS	0.358000
CRISPLD1	MT-CO2	0.357000
IDH1	RUNX1	0.357000
KIT	TP53	0.357000
NRAS	OR4H12P	0.357000
CBFB	DNMT3A	0.356000
LOC100130211	WT1	0.356000
ASXL1	DNMT3A	0.355000
CEBPA	PKD1L2	0.354000
CEBPA	PTPN11	0.354000
CRISPLD1	TET2	0.354000

CEBPA	PKHD1	0.353000
DNMT3A	SUZ12	0.353000
TET2	U2AF1	0.353000
ADCY5	DNMT3A	0.352000
MAGI2	NRAS	0.352000
PHF6	PTPN11	0.352000
DNMT3A	SMC3	0.350000
NMUR2	NRAS	0.350000
BCOR	CEBPA	0.348000
KRAS	NPM1	0.348000
NRAS	PHF6	0.347000
BCOR	TP53	0.346000
CEBPA	FAM47A	0.345000
FAM47A	NRAS	0.344000
FLT3	MTUS2	0.344000
NTRK3	TP53	0.344000
ATG16L1	DNMT3A	0.343000
CBL	NRAS	0.343000
FAM57B	NPM1	0.343000
MTUS2	PTCH1	0.343000
NRAS	TET1	0.343000

References

1. Bellazzi R. Big data and biomedical informatics: a challenging opportunity. Yearb Med Inform. 2014;9: 8–13. doi:10.15265/IY-2014-0024
2. de la Torre Díez I, Cosgaya HM, Garcia-Zapirain B, López-Coronado M. Big Data in Health: a Literature Review from the Year 2005. J Med Syst. Springer US; 2016;40: 209. doi:10.1007/s10916-016-0565-7
3. Fang R, Pouyanfar S, Yang Y, Chen S-C, Iyengar SS. Computational Health Informatics in the Big Data Age: a survey. ACM Comput Surv. 2016;49: 1–36. doi:10.1145/2932707
4. Sun J, Reddy C. Big data analytics for healthcare. SIAM Int Conf Knowl Discov data …. 2013; 1525. doi:10.1145/2487575.2506178
5. Gargis AS, Kalman L, Berry MW, Bick DP, Dimmock DP, Hambuch T, et al. Assuring the quality of next-generation sequencing in clinical laboratory practice. Nat Biotechnol. 2012;30: 1033–1036. doi:10.1038/nbt.2403
6. Rigden DJ, Fernández-Suárez XM, Galperin MY. The 2016 database issue of nucleic acids research and an updated molecular biology database collection. Nucleic Acids Res. 2016;44: D1–D6. doi:10.1093/nar/gkv1356
7. Benson DA, Cavanaugh M, Clark K, Karsch-Mizrachi I, Lipman DJ, Ostell J, et al. GenBank. Nucleic Acids Res. 2013;41. doi:10.1093/nar/gks1195
8. Kulikova T, Akhtar R, Aldebert P, Althorpe N, Andersson M, Baldwin A, et al. EMBL Nucleotide Sequence Database in 2006. Nucleic Acids Res. 2007;35. doi:10.1093/nar/gkl913
9. The UniProt Consortium. UniProt: a hub for protein information. Nucleic Acids Res. 2015;43: D204-12. doi:10.1093/nar/gku989
10. Bairoch A, Apweiler R. The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. Nucleic Acids Res. 2000;28: 45–48. doi:10.1093/nar/28.1.45
11. Schriml LM, Mitraka E. The Disease Ontology: fostering interoperability between biological and clinical human disease-related data. Mammalian Genome. 2015. pp. 584–589. doi:10.1007/s00335-015-9576-9
12. Hamosh A, Scott AF, Amberger JS, Bocchini CA, McKusick VA. Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. Nucleic Acids Res. 2005;33. doi:10.1093/nar/gki033
13. Kanehisa M, Goto S, Kawashima S, Okuno Y, Hattori M. The KEGG

resource for deciphering the genome. Nucleic Acids Res. 2004;32: D277-80. doi:10.1093/nar/gkh063
14. Fabregat A, Sidiropoulos K, Garapati P, Gillespie M, Hausmann K, Haw R, et al. The reactome pathway knowledgebase. Nucleic Acids Res. 2016;44: D481–D487. doi:10.1093/nar/gkv1351
15. Szklarczyk D, Franceschini A, Wyder S, Forslund K, Heller D, Huerta-Cepas J, et al. STRING v10: Protein-protein interaction networks, integrated over the tree of life. Nucleic Acids Res. 2015;43: D447–D452. doi:10.1093/nar/gku1003
16. Chatr-Aryamontri A, Breitkreutz BJ, Oughtred R, Boucher L, Heinicke S, Chen D, et al. The BioGRID interaction database: 2015 update. Nucleic Acids Res. 2015;43: D470–D478. doi:10.1093/nar/gku1204
17. Coletti MH, Bleich HL. Medical Subject Headings Used to Search the Biomedical Literature. J Am Med Informatics Assoc. 2001;8: 317–323. doi:10.1136/jamia.2001.0080317
18. O'Connor K, Pimpalkhute P, Nikfarjam A, Ginn R, Smith KL, Gonzalez G. Pharmacovigilance on twitter? Mining tweets for adverse drug reactions. AMIA Annu Symp Proc. 2014;2014: 924–33.
19. Martin Sanchez F, Gray K, Bellazzi R, Lopez-Campos G. Exposome informatics: considerations for the design of future biomedical research information systems. J Am Med Inform Assoc. 2014;21: 386–90. doi:10.1136/amiajnl-2013-001772
20. Sinha A, Hripcsak G, Markatou M. Large Datasets in Biomedicine: A Discussion of Salient Analytic Issues. J Am Med Informatics Assoc. 2009;16: 759–767. doi:10.1197/jamia.M2780
21. Andreu-Perez J, Poon CCY, Merrifield RD, Wong STC, Yang GZ. Big Data for Health. IEEE J Biomed Heal Informatics. 2015;19: 1193–1208. doi:10.1109/JBHI.2015.2450362
22. Altman RB, Ashley EA. Using "Big Data" to Dissect Clinical Heterogeneity. Circulation. 2015;131.
23. Desmond-Hellmann S, Sawyers CL. Toward Precision Medicine: Building a Knowledge Network for Biomedical Research and a New Taxonomy of Disease [Internet]. The National Academies Press. 2011. doi:10.17226/13284
24. Boulakia SC, Lair S, Stransky N, Graziani S, Radvanyi F, Barillot E, et al. Selecting biomedical data sources according to user preferences. Bioinformatics. 2004. doi:10.1093/bioinformatics/bth949
25. Searls DB. Data integration: challenges for drug discovery. Nat Rev Drug Discov. 2005;4: 45–58. doi:10.1038/nrd1608
26. Louie B, Mork P, Martin-Sanchez F, Halevy A, Tarczy-Hornoch P. Data integration and genomic medicine. Journal of Biomedical Informatics. 2007. pp. 5–16. doi:10.1016/j.jbi.2006.02.007
27. Banerjee S, Roy A. Linear Algebra and Matrix Analysis. 2009.
28. Trefethen LN, Bau III D. Numerical linear algebra. Numer Linear Algebr with Appl. 1997;12: 361. doi:10.1137/1.9780898719574
29. Demmel JW. Applied Numerical Linear Algebra [Internet]. Society for Industrial and Applied Mathematics SIAM. 1997. p. 436.

doi:10.1137/1.9781611971446
30. Businger PA, Golub GH. Algorithm 358: singular value decomposition of a complex matrix [F1, 4, 5]. Commun ACM. 1969;12: 565. doi:10.1145/363235.363249
31. Kalman D. A Singularly Valuable Decomposition: The SVD of a Matrix. Coll Math J. 1996;27: 2–23. doi:10.2307/2687269
32. Golub GH, Van Loan CF. Matrix Computations. Physics Today. 1996. p. 48. doi:10.1063/1.3060478
33. Golub G, Kahan W. Calculating the Singular Values and Pseudo-Inverse of a Matrix. Journal of the Society for Industrial and Applied Mathematics Series B Numerical Analysis. 1965. pp. 205–224. doi:10.1137/0702016
34. Biglieri E, Yao K. Some properties of singular value decomposition and their applications to digital signal processing. Signal Processing. 1989;18: 277–289. doi:10.1016/0165-1684(89)90039-X
35. Rahmati M, Sadri MS, Naeini MA. FPGA Based Singular Value Decomposition for Image Processing Applications. 2008 Int Conf Appl Syst Archit Process. 2008; 185–190. doi:10.1109/ASAP.2008.4580176
36. Rufai AM, Anbarjafari G, Demirel H. Lossy image compression using singular value decomposition and wavelet difference reduction. Digit Signal Process. 2014;24: 117–123. doi:10.1016/j.dsp.2013.09.008
37. Bhatnagar G, Jonathan Wu QM. Selective image encryption based on pixels of interest and singular value decomposition. Digit Signal Process. 2012;22: 648–663. doi:10.1016/j.dsp.2012.02.005
38. Jolliffe I, Jolliffe, Ian. Principal Component Analysis. Wiley StatsRef: Statistics Reference Online. Chichester, UK: John Wiley & Sons, Ltd; 2014. doi:10.1002/9781118445112.stat06472
39. Mahoney MW, Drineas P. CUR matrix decompositions for improved data analysis. Proc Natl Acad Sci U S A. National Academy of Sciences; 2009;106: 697–702. doi:10.1073/pnas.0803205106
40. Williams C, Seeger M. Using the Nyström Method to Speed Up Kernel Machines. Advances in Neural Information Processing Systems 13. 2001. pp. 682–688. doi:10.1017/CBO9781107415324.004
41. Frieze A, Kannan R, Vempala S. Fast Monte-Carlo algorithms for finding low-rank approximations. Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat No98CB36280). 2013. pp. 370–378. doi:10.1109/SFCS.1998.743487
42. Dhillon IS, Sra S. Generalized Nonnegative Matrix Approximations with Bregman Divergences. Adv Neural Inf Process Syst. 2005;19: 283–290. doi:10.1.1.72.5975
43. Lee D, Seung H. Algorithms for non-negative matrix factorization. Adv neural Inf Process …. 2001; 556–562. doi:10.1109/IJCNN.2008.4634046

44. Taslaman L, Nilsson B. A framework for regularized non-negative matrix factorization, with application to the analysis of gene expression data. PLoS One. Public Library of Science; 2012;7: e46331. doi:10.1371/journal.pone.0046331
45. Ding C, He X, Simon HD. On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering. Proc fifth SIAM Int Conf Data Min. 2005; 606–610. doi:10.1137/1.9781611972757.70
46. Pauca VP, Piper J, Plemmons RJ. Nonnegative matrix factorization for spectral data analysis. Linear Algebra Appl. 2006;416: 29–47. doi:10.1016/j.laa.2005.06.025
47. Berry MW, Browne M, Langville AN, Pauca VP, Plemmons RJ. Algorithms and applications for approximate nonnegative matrix factorization. Comput Stat Data Anal. 2007;52: 155–173. doi:10.1016/j.csda.2006.11.006
48. Guan N, Tao D, Luo Z, Yuan B. Online nonnegative matrix factorization with robust stochastic approximation. IEEE Trans Neural Networks Learn Syst. 2012;23: 1087–1099. doi:10.1109/TNNLS.2012.2197827
49. Luo X, Zhou M, Xia Y, Zhu Q. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. IEEE Trans Ind Informatics. 2014;10: 1273–1284. doi:10.1109/TII.2014.2308433
50. Nielsen FÅ, Balslev D, Hansen LK. Mining the posterior cingulate: Segregation between memory and pain components. Neuroimage. 2005;27: 520–532. doi:10.1016/j.neuroimage.2005.04.034
51. Kim H, Park H. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. Bioinformatics. 2007;23: 1495–1502. doi:10.1093/bioinformatics/btm134
52. Gao Y, Church G. Improving molecular cancer class discovery through sparse non-negative matrix factorization. Bioinformatics. 2005;21: 3970–3975. doi:10.1093/bioinformatics/bti653
53. Alexandrov LB, Nik-Zainal S, Wedge DC, Campbell PJ, Stratton MR. Deciphering Signatures of Mutational Processes Operative in Human Cancer. Cell Rep. 2013;3: 246–259. doi:10.1016/j.celrep.2012.12.008
54. Salakhutdinov R, Mnih A. Probabilistic Matrix Factorization. Proc Adv Neural Inf Process Syst 20 (NIPS 07). 2007; 1257–1264. doi:10.1145/1390156.1390267
55. Salakhutdinov R, Mnih A. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. Proc 25th Int Conf Mach Learn - ICML '08. 2008; 880–887. doi:10.1145/1390156.1390267
56. Gelman A, Carlin JB, Stern HS, Rubin DB. Bayesian Data Analysis. Chapman Texts in Statistical Science Series. 2004. doi:10.1007/s13398-014-0173-7.2
57. Jordan MI, Ghahramani Z, Jaakkola TS, Saul LK. Introduction to variational methods for graphical models. Mach Learn. 1999;37:

183–233. doi:10.1023/A:1007665907178
58. Kolda TG, Bader BW. Tensor Decompositions and Applications. SIAM Rev. 2008;51: 455–500. doi:10.1137/07070111X
59. Sidiropoulos ND, Bro R, Giannakis GB. Parallel factor analysis in sensor array processing. IEEE Trans Signal Process. 2000;48: 2377–2388. doi:10.1109/78.852018
60. Muti D, Bourennane S. Multidimensional filtering based on a tensor approach. Signal Processing. 2005. pp. 2338–2353. doi:10.1016/j.sigpro.2004.11.029
61. Fitzgerald D, Cranitch M, Coyle E. Non-negative Tensor Factorisation for Sound Source Separation. Acoust Speech Signal Process 2006 ICASSP 2006 Proceedings 2006 IEEE Int Conf. 2005;5: V--V. doi:10.1109/ICASSP.2006.1661360
62. Sun J, Papadimitriou S, Yu PS. Window-based tensor analysis on high-dimensional and multi-aspect streams. Proceedings - IEEE International Conference on Data Mining, ICDM. 2006. pp. 1076–1080. doi:10.1109/ICDM.2006.169
63. Sun J, Tao D, Faloutsos C. Beyond Streams and Graphs: Dynamic Tensor Analysis. Proc 12th ACM SIGKDD Int Conf Knowl Discov data Min - KDD '06. 2006; 374. doi:10.1145/1150402.1150445
64. Kolda TG, Bader BW, Kenny JP. Higher-order web link analysis using multilinear algebra. Proceedings - IEEE International Conference on Data Mining, ICDM. 2005. pp. 242–249. doi:10.1109/ICDM.2005.77
65. De Vos M, De Lathauwer L, Vanrumste B, Van Huffel S, Van Paesschen W. Canonical decomposition of ictal scalp EEG and accurate source localisation: Principles and simulation study. Comput Intell Neurosci. 2007;2007. doi:10.1155/2007/58253
66. De Vos M, Vergult A, De Lathauwer L, De Clercq W, Van Huffel S, Dupont P, et al. Canonical decomposition of ictal scalp EEG reliably detects the seizure onset zone. Neuroimage. 2007;37: 844–854. doi:10.1016/j.neuroimage.2007.04.041
67. Carroll JD, Chang JJ. Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition. Psychometrika. 1970;35: 283–319. doi:10.1007/BF02310791
68. Harshman R a. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. UCLA Work Pap Phonetics. 1970;16: 1–84. Available: http://www.psychology.uwo.ca/faculty/harshman/wpppfac0.pdf
69. Kolda T. Orthogonal Tensor Decompositions. SIAM J Matrix Anal Appl. 2001;23: 243–255. doi:10.1137/S0895479800368354
70. Håstad J. Tensor rank is NP-complete. J Algorithms. 1990;11: 644–654. doi:10.1016/0196-6774(90)90014-6
71. Tucker LR. Some mathematical notes on three-mode factor analysis. Psychometrika. 1966;31: 279–311. doi:10.1007/BF02289464
72. Bader BW, Kolda TG. Efficient MATLAB Computations with Sparse and Factored Tensors. SIAM J Sci Comput. 2008;30: 205–231.

doi:10.1137/060676489
73. Kroonenberg P, Leeuw DJ. Principal Component Analysis of 3-Mode Data by Means of Alternating Least-Squares Algorithms. Psychometrika. 1980;45: 69–97. doi:Doi 10.1007/Bf02293599
74. Lü L, Medo M, Yeung CH, Zhang Y-C, Zhang Z-K, Zhou T. Recommender systems. Phys Rep. 2012;519: 1–49. doi:10.1016/j.physrep.2012.02.006
75. Bobadilla J, Ortega F, Hernando A, Gutiérrez A. Recommender systems survey. Knowledge-Based Syst. 2013;46: 109–132. doi:10.1016/j.knosys.2013.03.012
76. Sivapalan S, Sadeghian A, Rahnama H, Madni AM. Recommender systems in e-commerce. World Automation Congress Proceedings. 2014. pp. 179–184. doi:10.1109/WAC.2014.6935763
77. Ricci F, Rokach L, Shapira B. Introduction to Recommender Systems Handbook. Recommender Systems Handbook. 2011. pp. 1–35. doi:10.1007/978-0-387-85820-3_1
78. Bennett J, Lanning S. The Netflix Prize. KDD Cup Work. 2007; 3–6. doi:10.1145/1562764.1562769
79. Yoneya T, Mamitsuka H. PURE: a PubMed article recommendation system based on content-based filtering. Genome Inform. 2007;18: 267–276. Available: http://www.ncbi.nlm.nih.gov/pubmed/18546494
80. Jamali M, Ester M. A matrix factorization technique with trust propagation for recommendation in social networks. Proc fourth ACM Conf Recomm Syst - RecSys '10. 2010; 135–142. doi:10.1145/1864708.1864736
81. Hussein AS, Omar WM, Li X, Ati M. Efficient Chronic Disease Diagnosis prediction and recommendation system. 2012 IEEE-EMBS Conference on Biomedical Engineering and Sciences, IECBES 2012. 2012. pp. 209–214. doi:10.1109/IECBES.2012.6498117
82. Vlahu-Gjorgievska E, Trajkovik V. Towards collaborative health care system model - COHESY. 2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, WoWMoM 2011 - Digital Proceedings. 2011. doi:10.1109/WoWMoM.2011.5986197
83. Bellogín A, Cantador I, Díez F, Castells P, Chavarriaga E. An empirical comparison of social, collaborative filtering, and hybrid recommenders. ACM Trans Intell Syst Technol. 2013;4: 1–29. doi:10.1145/2414425.2414439
84. Terveen L, Hill W. Beyond recommender systems: Helping people help each other. HCI New Millenn. 2001; 487–509. doi:10.1.1.26.2437
85. Burke R. Hybrid web recommender systems. Adapt web. 2007; 377–408. doi:10.1007/978-3-540-72079-9_12
86. Shi YU, Larson M, Hanjalic A. Collaborative Filtering beyond the User-Item Matrix : A Survey of the State of the Art and Future Challenges. ACM Comput Surv. 2014;47: 1–45. doi:http://dx.doi.org/10.1145/2556270

87. Elahi M, Ricci F, Rubens N. A survey of active learning in collaborative filtering recommender systems. Computer Science Review. 2016. pp. 29–50. doi:10.1016/j.cosrev.2016.05.002
88. Koren Y, Bell R. Advances in collaborative filtering. Recommender Systems Handbook, Second Edition. 2015. pp. 77–118. doi:10.1007/978-1-4899-7637-6_3
89. Linden G, Smith B, York J. Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Comput. 2003;7: 76–80. doi:10.1109/MIC.2003.1167344
90. Koren Y, Bell R, Volinsky C. Matrix Factorization Techniques for Recommender Systems. Computer (Long Beach Calif). 2009;42: 42–49. doi:10.1109/MC.2009.263
91. Koren Y. Factorization meets the neighborhood. Proceeding 14th ACM SIGKDD Int Conf Knowl Discov data Min - KDD 08. 2008; 426. doi:10.1145/1401890.1401944
92. Koren Y. Collaborative filtering with temporal dynamics. Proc KDD '09. 2009; 447–456. doi:10.1145/1557019.1557072
93. Rendle S. Factorization machines. Proceedings - IEEE International Conference on Data Mining, ICDM. 2010. pp. 995–1000. doi:10.1109/ICDM.2010.127
94. Steffen R. Factorization Machines with libFM. TIST. 2012;3: 1–22. doi:10.1145/2168752.2168771
95. Brereton RG, Lloyd GR. Support vector machines for classification and regression. Analyst. 2010;135: 230–267. doi:10.1039/b918972f
96. Bottou L. Stochastic Gradient Descent Tricks. Neural Networks: Tricks of the Trade. 2012;1: 421–436. doi:10.1007/978-3-642-35289-8
97. Zachariah D, Sundin M, Jansson M, Chatterjee S. Alternating least-squares for low-rank matrix reconstruction. IEEE Signal Process Lett. 2012;19: 231–234. doi:10.1109/LSP.2012.2188026
98. Rendle S, Gantner Z, Freudenthaler C, Schmidt-Thieme L. Fast context-aware recommendations with factorization machines. Proc 34th Int ACM SIGIR Conf Res Dev Inf - SIGIR '11. 2011; 635. doi:10.1145/2009916.2010002
99. Hall DLDL, Member S, Llinas J. An introduction to multisensor data fusion. Proc IEEE. 1997;85: 6–23. doi:10.1109/5.554205
100. Boström H, Andler SF, Brohede M, Johansson R, Karlsson A, Laere J Van, et al. On the Definition of Information Fusion as a Field of Research. IKI Tech Reports. 2007; 1–8. doi:HS-IKI-TR-07-006
101. Greene D, Cunningham P. A matrix factorization approach for integrating multiple data views. Machine Learning and Knowledge Discovery in Databases. 2009. pp. 423–438. doi:10.1007/978-3-642-04180-8
102. Potamianos A, Perakakis M. Multimodal Processing and Interaction [Internet]. Multimedia Systems and Applications Series. 2008. doi:10.1007/978-0-387-76316-3
103. Zitnik M, Zupan B. Data Fusion by Matrix Factorization. IEEE Trans

Pattern Anal Mach Intell. 2015;37: 41–53. doi:10.1109/TPAMI.2014.2343973
104. Han S, Liao X, Carin L. Cross-Domain Multitask Learning with Latent Probit Models. Proc 29th Int Conf Mach Learn. 2012; 1463–1470.
105. Yang H, He J. Learning with Dual Heterogeneity : A Nonparametric Bayes Model. KDD. 2014; 582–590. doi:10.1145/2623330.2623727
106. Lanckriet GRG, De Bie T, Cristianini N, Jordan MI, Noble WS. A statistical framework for genomic data fusion. Bioinformatics. 2004;20: 2626–35. doi:10.1093/bioinformatics/bth294
107. Varma M, Babu BR. More generality in efficient multiple kernel learning. Proc 26th Annu Int Conf Mach Learn - ICML '09. 2009;2009: 1–8. doi:10.1145/1553374.1553510
108. Zitnik M, Janjic V, Larminie C, Zupan B, Przulj N. Discovering disease-disease associations by fusing systems-level molecular data. Sci Rep. 2013;3: 3202. doi:10.1038/srep03202
109. Žitnik M, Zupan B. Matrix factorization-based data fusion for drug-induced liver injury prediction. Syst Biomed. 2014;8130: 37–41. doi:10.4161/sysb.29072
110. Žitnik M, Nam EA, Dinh C, Kuspa A, Shaulsky G, Zupan B. Gene Prioritization by Compressive Data Fusion and Chaining. PLoS Comput Biol. 2015;11. doi:10.1371/journal.pcbi.1004552
111. Zitnik M, Zupan B. Matrix factorization-based data fusion for gene function prediction in baker's yeast and slime mold. Pac Symp Biocomput. 2014; 400–411. doi:10.1142/9789814583220_0038
112. Vitali F, Cohen LD, Demartini A, Amato A, Eterno V, Zambelli A, et al. A Network-Based Data Integration Approach to Support Drug Repurposing and Multi-Target Therapies in Triple Negative Breast Cancer. PLoS One. 2016;11: e0162407. doi:10.1371/journal.pone.0162407
113. Gligorijević V, Malod-Dognin N, Prulj N. Fuse: Multiple network alignment via data fusion. Bioinformatics. 2016;32: 1195–1203. doi:10.1093/bioinformatics/btv731
114. Gonen M, Kaski S. Kernelized Bayesian Matrix Factorization. IEEE Trans Pattern Anal Mach Intell. 2014;36: 2047–2060. doi:10.1109/TPAMI.2014.2313125
115. Bayar B, Bouaynaya N, Shterenberg R. Probabilistic non-negative matrix factorization: theory and application to microarray data analysis. J Bioinform Comput Biol. 2014;12: 1450001. doi:10.1142/S0219720014500012
116. Xu Z, Yan F, Qi Y. Bayesian nonparametric models for multiway data analysis. IEEE Trans Pattern Anal Mach Intell. 2015;37: 475–487. doi:10.1109/TPAMI.2013.201
117. Li J, Tao D. A bayesian hierarchical factorization model for vector fields. IEEE Trans Image Process. 2013;22: 4510–4521. doi:10.1109/TIP.2013.2274732
118. Shashanka M, Raj B, Smaragdis P. Probabilistic latent variable models as nonnegative factorizations. Comput Intell Neurosci. 2008;2008: 947438. doi:10.1155/2008/947438

119. PDQ Adult Treatment Editorial Board. Chronic Myelogenous Leukemia Treatment (PDQ®): Health Professional Version. In: PDQ Cancer Information Summaries [Internet]. 2002. Available: http://www.cancer.gov/types/leukemia/hp/cml-treatment-pdq
120. Ntziachristos P, Mullenders J, Trimarchi T, Aifantis I. Mechanisms of Epigenetic Regulation of Leukemia Onset and Progression. Adv Immunol. 2013;117: 1–38. doi:10.1016/B978-0-12-410524-9.00001-3
121. Shih AH, Abdel-Wahab O, Patel JP, Levine RL. The role of mutations in epigenetic regulators in myeloid malignancies. Nat Rev Cancer. 2012;12: 599–612. doi:10.1038/nrc3343
122. Arber DA, Orazi A, Hasserjian R, Thiele J, Borowitz MJ, Le Beau MM, et al. The 2016 revision to the World Health Organization classification of myeloid neoplasms and acute leukemia. Blood. 2016. pp. 2391–2405. doi:10.1182/blood-2016-03-643544
123. Theilgaard-Mönch K, Boultwood J, Ferrari S, Giannopoulos K, Hernandez-Rivas JM, Kohlmann A, et al. Gene expression profiling in MDS and AML: potential and future avenues. Leukemia. 2011;25: 909–20. doi:10.1038/leu.2011.48
124. Tartaglia M, Niemeyer CM, Fragale A, Song X, Buechner J, Jung A, et al. Somatic mutations in PTPN11 in juvenile myelomonocytic leukemia, myelodysplastic syndromes and acute myeloid leukemia. Nat Genet. 2003;34: 148–150. doi:10.1038/ng1156
125. Harada H, Harada Y, Niimi H, Kyo T, Kimura A, Inaba T. High incidence of somatic mutations in the AML1/RUNX1 gene in myelodysplastic syndrome and low blast percentage myeloid leukemia with myelodysplasia. Blood. 2004;103: 2316–2324. doi:10.1182/blood-2003-09-3074
126. Papaemmanuil E, Cazzola M, Boultwood J, Malcovati L, Vyas P, Bowen D, et al. Somatic SF3B1 mutation in myelodysplasia with ring sideroblasts. N Engl J Med. 2011;365: 1384–95. doi:10.1056/NEJMoa1103283
127. Gerstung M, Pellagatti A, Malcovati L, Giagounidis A, Porta MG Della, Jädersten M, et al. Combining gene mutation with gene expression data improves outcome prediction in myelodysplastic syndromes. Nat Commun. 2015;6: 5901. doi:10.1038/ncomms6901
128. Piñero J, Queralt-Rosinach N, Bravo À, Deu-Pons J, Bauer-Mehren A, Baron M, et al. DisGeNET: A discovery platform for the dynamical exploration of human diseases and their genes. Database. 2015;2015. doi:10.1093/database/bav028
129. Park SG, Schimmel P, Kim S. Aminoacyl tRNA synthetases and their connections to disease. Proc Natl Acad Sci U S A. 2008;105: 11043–9. doi:10.1073/pnas.0802862105
130. Wang H, Li Y, Wu J, Guo S. Mitochondrial tRNA mutations in patients with myelodysplastic syndromes. Mitochondrial DNA. Taylor & Francis; 2015; 1–3. doi:10.3109/19401736.2015.1022760
131. Gattermann N, Rachmilewitz EA. Iron overload in MDS-pathophysiology, diagnosis, and complications. Annals of

Hematology. 2011. pp. 1–10. doi:10.1007/s00277-010-1091-1
132. Wells RA, Leber B, Buckstein R, Lipton JH, Hasegawa W, Grewal K, et al. Iron overload in myelodysplastic syndromes: A Canadian consensus guideline. Leukemia Research. 2008. pp. 1338–1353. doi:10.1016/j.leukres.2008.02.021
133. Arcangeli a., Pillozzi S, Becchetti a. Targeting Ion Channels in Leukemias: A New Challenge for Treatment. Curr Med Chem. 2012;19: 683–696. doi:BSP/CMC/E-Pub/2012/056 [pii]
134. Gupta M, Madkaikar M, Rao VB, Mishra A, Govindaraj P, Thangaraj K, et al. Mitochondrial DNA variations in myelodysplastic syndrome. Ann Hematol. 2013;92: 871–876. doi:10.1007/s00277-013-1706-4
135. Visconte V, Makishima H, Maciejewski JP, Tiu R V. Emerging roles of the spliceosomal machinery in myelodysplastic syndromes and other hematological disorders. Leukemia. 2012;26: 2447–54. doi:10.1038/leu.2012.130
136. Yoshida K, Sanada M, Shiraishi Y, Nowak D, Nagata Y, Yamamoto R, et al. Frequent pathway mutations of splicing machinery in myelodysplasia. Nature. 2011;478: 64–9. doi:10.1038/nature10496
137. Niimi H, Harada H, Harada Y, Ding Y, Imagawa J, Inaba T, et al. Hyperactivation of the RAS signaling pathway in myelodysplastic syndrome with AML1/RUNX1 point mutations. Leukemia. 2006;20: 635–44. doi:10.1038/sj.leu.2404136
138. Reuter CW, Morgan M a, Bergmann L. Targeting the Ras signaling pathway: a rational, mechanism-based treatment for hematologic malignancies? Blood. 2000;96: 1655–1669.
139. Follo MY, Mongiorgi S, Bosi C, Cappellini A, Finelli C, Chiarini F, et al. The Akt/mammalian target of rapamycin signal transduction pathway is activated in high-risk myelodysplastic syndromes and influences cell survival and proliferation. Cancer Res. 2007;67: 4287–4294. doi:10.1158/0008-5472.CAN-06-4409
140. Bellon M, Lepelletier Y, Hermine O, Nicot C. Deregulation of microRNA involved in hematopoiesis and the immune response in HTLV-I adult T-cell leukemia. Blood. 2009;113: 4914–4917. doi:10.1182/blood-2008-11-189845
141. Pellagatti A, Cazzola M, Giagounidis A, Perry J, Malcovati L, Della Porta M, et al. Deregulated gene expression pathways in myelodysplastic syndrome hematopoietic stem cells. Leukemia. 2010;24: 756–764. doi:10.1038/leu.2010.31
142. von Mering C, Jensen LJ, Snel B, Hooper SD, Krupp M, Foglierini M, et al. STRING: Known and predicted protein-protein associations, integrated and transferred across organisms. Nucleic Acids Res. 2005;33. doi:10.1093/nar/gki005
143. Barrett T, Wilhite SE, Ledoux P, Evangelista C, Kim IF, Tomashevsky M, et al. NCBI GEO: Archive for functional genomics data sets - Update. Nucleic Acids Res. 2013;41. doi:10.1093/nar/gks1193
144. Estey E, Döhner H. Acute myeloid leukaemia. Lancet. 2006;368: 1894–1907. doi:16/S0140-6736(06)69780-8

145. Löwenberg B, Downing JR, Burnett A. Acute myeloid leukemia. N Engl J Med. 1999;341: 1051–1062. doi:10.1056/NEJM199909303411407
146. Cancer T, Atlas G. Genomic and Epigenomic Landscapes of Adult De Novo Acute Myeloid Leukemia The Cancer Genome Atlas Research Network. N Engl J Med. 2013;368: 2059–74. doi:10.1056/NEJMoa1301689
147. Stirewalt DL, Kopecky KJ, Meshinchi S, Appelbaum FR, Slovak ML, Willman CL, et al. FLT3, RAS, and TP53 mutations in elderly patients with acute myeloid leukemia. Blood. 2001;97: 3589–3595. doi:10.1182/blood.V97.11.3589
148. Zhang J, Song L-P, Huang Y, Zhao Q, Zhao K-W, Chen G-Q. Accumulation of hypoxia-inducible factor-1 alpha protein and its role in the differentiation of myeloid leukemic cells induced by all-trans retinoic acid. Haematologica. 2008;93: 1480–7. doi:10.3324/haematol.13096
149. Nakao M, Horiike S, Fukushima-Nakase Y, Nishimura M, Fujita Y, Taniwaki M, et al. Novel loss-of-function mutations of the haematopoiesis-related transcription factor, acute myeloid leukaemia 1/runt-related transcription factor 1, detected in acute myeloblastic leukaemia and myelodysplastic syndrome. Br J Haematol. 2004;125: 709–719. doi:10.1111/j.1365-2141.2004.04966.x
150. Kroeze LI, Aslanyan MG, Van Rooij A, Koorenhof-Scheele TN, Massop M, Carell T, et al. Characterization of acute myeloid leukemia based on levels of global hydroxymethylation. Blood. 2014;124: 1110–1118. doi:10.1182/blood-2013-08-518514
151. Shih L-Y, Liang D-C, Huang C-F, Wu J-H, Lin T-L, Wang P-N, et al. AML patients with CEBPalpha mutations mostly retain identical mutant patterns but frequently change in allelic distribution at relapse: a comparative analysis on paired diagnosis and relapse samples. Leukemia. 2006;20: 604–609. doi:10.1038/sj.leu.2404124
152. Scholl S, Fricke H-J, Sayer HG, Höffken K. Clinical implications of molecular genetic aberrations in acute myeloid leukemia. J Cancer Res Clin Oncol. 2009;135: 491–505. doi:10.1007/s00432-008-0524-x
153. Kao HW, Liang DC, Wu JH, Kuo MC, Wang PN, Yang CP, et al. Gene Mutation Patterns in Patients with Minimally Differentiated Acute Myeloid Leukemia. Neoplasia (United States). 2014;16: 481–488. doi:10.1016/j.neo.2014.06.002
154. Yuan Y, Zhou L, Miyamoto T, Iwasaki H, Harakawa N, Hetherington CJ, et al. AML1-ETO expression is directly involved in the development of acute myeloid leukemia in the presence of additional mutations. Proc Natl Acad Sci U S A. 2001;98: 10398–10403. doi:10.1073/pnas.171321298
155. Home - The Cancer Genome Atlas - Cancer Genome - TCGA [Internet]. Available: https://cancergenome.nih.gov/
156. Hsu S Da, Lin FM, Wu WY, Liang C, Huang WC, Chan WL, et al.

MiRTarBase: A database curates experimentally validated microRNA-target interactions. Nucleic Acids Res. 2011;39. doi:10.1093/nar/gkq1107