Trend Prediction Using Fashion Datasets

MASTER THESIS

Ahana Mallik

Supervisors:
Dr. Mourad Khayati
Prof. Philippe Cudré-Mauroux

Universiy of Bern

November, 2019

Declaration of Authorship

I, Ahana Mallik

Supervisors:
Dr. Mourad Khayati
Prof. Philippe Cudré-Mauroux , declare that this thesis titled, "Trend Prediction Using Fashion Datasets" and the work presented in it are my own. I confirm that:

- This work was done wholly while in candidature for Master degree at this University.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.

Acknowledgements

I am very grateful to Dr. Mourad Khayati and Ines Arous for the help to understand main concepts of the algorithms and complete my thesis. I am also thankful to Prof. Philippe Cudre-Mauroux for giving me an opportunity of doing Master Thesis in the department of eXascale Infolab.

Contents

List of Figures

Abstract

Recommender systems aim to help users access and retrieve relevant information about items from large collections of item pool, by automatically finding products or services of interest based on observed evidence of the user's preferences. For many reasons, user preferences are difficult to guess, so Recommender systems have a considerable variance or difference in their success ratio in estimating the user's future interests or preferences. In this economic world customer preferences or interests drift over time, the users easily get attracted towards different new products. The item profiles or characteristics also drift over time. In order to keep track of user preferences the Recommender systems generate meaningful recommendations to a collection of users for items or products that might interest them. Recommender systems need to be temporal aware so that they suggest users items that fit their purpose of need. For example a Recommender system must be such designed so that it recommend products like winter creams in winter and sunscreen in summer or rain coats in monsoon. Otherwise, the users will receive recommendations for wrong items which will not fit their purpose. There are several Recommender systems available which are capable of capturing the dynamic user behaviour and item profiles. In our thesis we have implemented three different time and visually aware Recommender models in the context of Amazon fashion data. Our proposed models are namely TVBPRPlus, Timesvd++ and FARIMA. We have performed several experiments to verify the efficiency and accuracy of these models. We have also compared the models under certain settings to determine which model performs better than the other models. The main focus of our thesis is future trend prediction of a product(s) with the help of these prediction techniques. For this purpose, we have designed a graphical tool to visualize the predicted results generated by applying these different prediction models.

Keywords: Recommender system, time series analysis, Trend Prediction, Collaborative Filtering, Temporal and Visual dynamics of data, Concept Drift, Feature extraction.

Chapter 1

Introduction

The increasing importance of the Web as a medium for electronic and business transactions has served as a driving force for the development of Recommender systems. Web enables users to provide feedback about their likes or dislikes. For example, we can consider a simple scenario of a content provider such as FlikKart.com or Amazon.com, where the users are able to easily provide feedback with a simple click of a mouse. The users provide feedback in the form of ratings, in which users select numerical values from a specific evaluation system (For example five-star rating system) that specify their likes and dislikes about various items. There are other forms of feedback which are easier to collect in a web centered paradigm. For example we can consider a scenario of a user buying or browsing an item, this may be viewed as an endorsement for that item. Such forms of feedback are commonly used by online merchants such as Amazon.com or flipKart.com, and the collection of this type of data is completely effortless in terms of the work required of a customer. The job of Recommender systems is to utilize these various sources of data to infer customer interests. The entity to which the recommendation is provided is referred to as the user, and the product being recommended is also referred to as an item. Therefore, recommendation analysis is most commonly based on the previous interaction between users and items. Since, past interests or likes of the users are often good indicators of their future choices. This type of recommendation process is known as **Collaborative Filtering.** We will be discussing about this recommendation method in details in the later sections. There are some exceptions such as the case of knowledge-based Recommender systems, in which the recommendations are suggested based on user-specified requirements rather than the past history of the users.

The basic principle of recommendations is that dependencies or interactions exist between users and items. For example, a user who is interested in washing machine or refrigerator is more likely to be interested in other household appliances rather than in a cosmetic product. The user-item dependencies can be learned in a data-driven manner from the ratings matrix, and the resulting model is used to make predictions for target users. The larger the number of rated items that are available for a user, the easier it is to make robust predictions about the future behaviour or preference of the user. There are many different learning models which can be used to accomplish this task. For example there are neighbourhood models. This family belongs to a broader class of models, referred to as Collaborative Filtering. The term Collaborative Filtering refers to the use of ratings from multiple users in a collaborative way to predict unknown ratings. In practice, Recommender systems can be more complex and data-rich, with a wide variety of auxiliary data types. For example, in content-based Recommender systems, the content plays a primary role in the recommendation process, in which the ratings of users and the attributes of

items are used in order to make predictions. The basic idea is that user interests can be modelled on the basis of properties of the items they have rated in the past. A different framework is that of knowledge-Based Recommender systems, in which users interactively specify their interests, and the user specification is combined with domain knowledge to provide recommendations. In advanced models, contextual data, such as temporal information, external knowledge, location information, social information, or network information, may be used.

In real world scenarios the customer preferences change over time. There are several factors responsible for drifting customer preferences. Some of these factors are emergence of new items in the market, seasonal changes, specific holidays or localized factors like change in family culture. There are also some other reasons responsible for changing customer interests for example a user may change his/her personal liking towards an item. In order to explain this case we take the example of a random user (for example Bob) who used to like Jeans of **"Denim"** is now more interested towards Jeans of **"Lee"**. Sometimes users also change his/her rating style. For example a user who used to give rating "3" as neutral marking may now give "2" as neutral marking. The items characteristics also change over time. When new items with added features or new visual styles emerge into the market the old items lose their popularity. For example a particular visual style or visual feature of a product for example, the visual appearance of **"Nike"** shoes may be dominant during a certain fashion epoch but it might lose its popularity down the time-line. Thus, during different fashion epochs different visual styles of items gain/lose their preferences. This phenomenon of drifting user choices and item properties is commonly known as **Concept Drift** [**Koren10**]. It is very important for Recommender systems to capture the temporal and visual dynamics of data. So, that they always provide updated data to the customers. According to the paper [**Koren10**] there three different ways of handling the Concept Drift. The first way is **Instance Selection** approach where only the recent instances are given importance for example the time window based approach which only considers recent instances. The second approach is **Instance Weighing** approach where the different instances are weighted based on their relevance and then taken into consideration. The third approach is called **Ensemble Learning** where a group of predictors produce a final predicted result. The predictors which are more relevant in recent instances receive higher values.

In our thesis we have worked with three different prediction models. These models are capable of uncovering the temporal and visual dynamics associated with data. For example our implemented model Timesvd++ is capable of tracking the time changing behaviour of users and items throughout the lifespan of data. User preferences changes as new fashionable products emerges for example, a trendy watch of **"Casio"**. Emergence of new items also causes item profiles to change dynamically. These dynamics of data are captured by Timesvd++. Our next model known as TVBPRPlus is capable of capturing the dynamically changing visual features of different items during different fashion epochs. With the help of TVBPRPlus model we are able to compute the visual scores of different items. This further helps us to determine which item is more popular compared to other items during a certain epoch. The visual scores also plays a key role in finding visually similar items. The items or products with same visual score are considered similar to each other. These visually similar items are placed near to each other in visual space. For example there can be different categories of Metallic watches or Sports shoes which

belong to different brands and are rated by different users but they may be visually similar (for example items with common visual features) based on their visual scores. TVBPRPlus also helps to determine which visual style of a product is more dominant than others during a certain epoch under different visual dimensions. For example a special type of women dress or women sunglasses may become most visually Representative during a fashion epoch under a specific visual dimension. Our first two models are based on **Product Level Modelling**. There are certain drawbacks of Product Level Modelling such as lack of sufficient data to predict future preferences of customers. So, we have used **Feature Level Modelling** through our third model Fourier Assisted ARIMA (FARIMA). In this approach we extract explicit features in a specific product domain from users textual reviews. Thus, we are able to eliminate the problem of data insufficiency. FARIMA provides us the ability to predict future trend of products through time series analysis. The expanded feature space is also helpful for cold start users who have rated very few items.

1.1 Organization of the Thesis

We have organized our work in following chapters. Chapter 2 provides description about similar work. In chapter 3 we discuss about background of Recommender systems and describe different evaluation techniques applied on our models. We describe our implemented models in details in Chapter 4. We provide information regarding the Challenges of Recommender systems in Chapter 5. We evaluate the performances of our models in Chapter 6. In Chapter 7 we present the graphical tool to visualize future predicted results applying the different prediction techniques. In Chapter 8 we discuss the various difficulties and challenges of our thesis. In Chapter 9 we conclude our work and finally in Chapter 10 we present the future scope of our thesis.

1.2 Motivation of our work

Fashion is an interesting domain where different products such as clothes, accessories, shoes etc become popular at different time periods. The users are always in search of new fashionable items. With the emergence of the web, the electronic market has expanded and the customers are provided with increasing range of options. Web helps the users to easily search for new fashionable products. But users feel good when fashion items are recommended without asking. For example let us consider a certain user who has purchased some products such as jeans, tees etc. It is more interesting for the user when the system in future recommends him fashionable products such as shoes, watches. This has motivated us to design recommender models which will help users to know about the future trend of different fashion products. Moreover, designing Recommender systems will provide a scalable way of personalizing content for users with an expanded product space. This expanded product space will allow the users to browse through interesting items and select items of choice.

The fashion items lose or gain their attractiveness over time. Certain items become popular at specific time period but later they lose their importance. For example fashion products like sunscreen, sunglasses and T-shirts are more preferred by users during summer, whereas products such as jackets and fur coats are more popular during winter. Thus, fashion dynamics evolve over time. Capturing these temporal dynamics of fashion data is really challenging. A lot of research work has been

done in this field [**HeM16**][**Koren10**][**Zhang0ZLLZM15**]. We are highly motivated by these research works. So, in our thesis we have developed recommender systems which captures the temporal dynamics of fashion products.

1.3 Goal of our thesis

The main focus of our thesis is to develop three different prediction models namely, TVBPRPlus, Timesvd++ and FARIMA and compare them under different input conditions. We have also designed a graphical tool as an additional goal to visualize the predicted results of the models for different fashion products.

Chapter 2

Similar Work

There has been a lot of work done by many researchers for capturing the temporal/visual dynamics of data and determining the future user preferences towards different items. There are some approaches which only uncovers the temporal changes in data such as Timesvd++ [**Koren10**]. There are also techniques which captures both visual and temporal variations of data for example TVBPRPlus [**HeM16**]. These approaches are based on the concept of Collaborative Filtering. They consider the explicit user feedback (past user purchase history) for determining the future preferences of users over the time. Besides, these approaches we also have models which are based on Feature Level Modelling. These models extract explicit features of a particular item domain from text reviews of users [**Zhang0ZLLZM15**]. In this way this type of an approach predicts the future trends of different products through time series analysis. Fourier Assisted ARIMA (FARIMA) is based on Feature Level Modelling [**Zhang0ZLLZM15**]. With the help of this type of a model we are able to overcome the data insufficiency problem of previous models.

2.1 Work Related to Temporal and Visual Dynamics of data

A lot of work done in incorporating visual and temporal dynamics on implicit feedback data sets where visual signals or styles are given more importance than star ratings. The decision that visual styles are given more importance compared to star ratings is due to the fact that new visual styles evolve on a regular basis which is more closely reflected in purchase choices than in ratings. For example users generally purchase products only when they are attracted towards the visual appearance or visual style of the items. For example a random user (say Alice) will usually buy a product (Fancy Bags or Fancy dress) only when she is convinced by the visual attractiveness of the product. Thus, the variations in ratings can be explained by the non-visual dynamics but the variations in purchases are a combination of both the visual and non-visual dynamics. Thus, visual features of a product plays an important role in the buying pattern of users. As described in the paper [**HeM16**], Recommender systems are build which will estimate the users personalized ranking based purchase histories. These Recommender systems are well equipped to capture the evolving fashion styles during different fashion epochs. In the paper [**HeM16**] a set of users and items are considered where each user from user set provides implicit feedback on items which belongs to the item set at a particular time and also each item is associated with an image. Considering the above data for each user a time dependent personalized ranking is generated of those items which the user has rated. Efficient methods are described in the paper [**HeM16**], which captures the raw images of items to learn the visual styles which are evolving temporally and predict the user preferences. The paper describes the concept of Collaborative

Filtering (CF). CF was first introduced by pan et all to work in scenarios where positive feedback is available, the unknown feedbacks are sampled as negative instances and matrix factorization methods are applied. This method is explained by Hu et all[**HeM16**] where varying confidence levels are assigned to different feedbacks and factorized to produce the resultant weighted matrix. This model is classified as point wise method. Rand et all later introduced pair wise method where the concept of Bayesian Personalized Rank (BPR) was proposed. The paper[**HeM16**] describes developing scalable Recommender systems on top of product images and user feedback to capture user preferences and fashion styles which drifts temporally. CNN features are described in the paper[**HeM16**] for modelling visual dynamics. For example Ruining He is his work [**HeM16**], describes visually aware Recommender systems such as TVBPRPlus which are temporally evolving, scalable, personalized and interoperable. According to his paper [**HeM16**] the user/item visual factors are considered as a function of time. The items gain or lose attractiveness in different fashion epochs under various visual dimensions . This is called **Temporal Attractiveness Drift**. The customers weigh visual dimensions differently as fashion changes over time. For example users may pay less attention to a dimension which describes colourfulness as they have become more tolerant to bright colours. This is called **Temporal weighing drift** [**HeM16**]. The user preferences are affected by both the outside fashion trends and their own personal tastes. This is called **Temporal Personal Drift** and this concept is also described in the paper. Here the user preferences are considered as a function of time, where a deviation of user opinion at time particular time from his mean feedback date or time is taken into account. Ruining He also describes the concept of temporally evolving **visual bias**[**HeM16**]. The visual bias at a particular time is considered as an inner product of visual features and bias vector. Various factors can cause an item to be purchased at different time periods This is called **per-item temporal dynamics** [**HeM16**], where the stationary item bias is replaced by a time factor. In case of data sets where category tree is available we incorporate **per-category temporal dynamics** [**HeM16**]. Here a temporal sub-category bias term is introduced to account for drifting user preferences. We have considered the previous work of these researchers in order to develop time and visual aware Recommender systems for predicting the future trend of product.

2.2 Work Related to Temporal Dynamics of data

It is known from many research works that Temporal dynamics relates to the idea of *Concept Drift*. There are different approaches used for handling concept drift. For example Koren in his paper [**Koren10**], has designed model for example Timesvd++ which considers the temporal dynamics of the data to predict user preferences towards items based on past feedback (ratings) of users. The model is basically divided into two major components. The first component is known as **Baseline Predictors**. The Baseline Predictors are responsible for capturing the temporal dynamics of data. The second component is known as **User-Item Interaction**. This component keeps a track of user choices. For example a user who is interested in Shirts of "Lee" can change his interest to Shirts of a different brand for example "Peter England". In our thesis we have used this approach for future trend prediction of product. We will be discussing this approach in details in later section.

2.3 Work Related to Product Feature Modelling

Zhang et all in his paper [**Zhang0ZLLZM15**], presents personalized recommendation by making use of textual information to extract explicit product features in a specific product domain. With these extracted features the future trends of different products are estimated by time series analysis. We have learned this approach and developed FARIMA in our thesis. We will be discussing the working procedure of FARIMA in details in later section.

Chapter 3

Background and Evaluation Techniques of Recommender Systems

In this chapter we discuss about the background of Recommender systems and the different evaluation techniques used to measure the performances of the models.

3.1 Background of Recommender Systems

Recommender Systems are used to generate meaningful recommendations to a set of users for products that might interest them. For example suggestions for books or clothes on Amazon or movies on Netflix are a few popular Recommender systems. The design of these recommendation engines depends on the domain and the particular properties or characteristics of the data available. For example, customers of Amazon or FlipKart provide ratings in form of 1 to 5 scale about different fashion products. So, the data source records the quality of interactions between users and items. Additionally, the system has access to user-specific and item-specific profile attributes such as demographics and product descriptions or characteristics respectively. Recommender systems differ in the way in which they analyze these data sources to develop affinity between users and items. This information can be used later to identify well-matched pairs. They are some approaches of Recommendation which are as follows.

1. **Collaborative Filtering:** Collaborative Filtering recommends to users different products with which the users have never interacted. These recommended products are similar to those products which the users have already purchased or visited. This approach also recommends items to a user based on other users who are similar to the target user. Thus, this approach predicts the future user preferences based on their past feedback or purchase histories.

2. **Content Filtering:** Content Filtering approach determines the future user preferences towards different items by taking into account both user feedback and user/item properties (content information). This approach of generating recommendations is not as popular as Collaborative Filtering since gathering the content data about users and items are at times difficult.

3. **Hybrid Techniques:** Hybrid techniques attempt to combine both of these designs. The architecture of this type Recommender systems and their evaluation on real-world data sets is an active area of research today.

4. **Knowledge-based Recommender System:** Knowledge-based Recommender systems are particularly useful in case of items which are rarely purchased. Examples include items such as real estate, automobiles, tourism requests, financial services. In these cases, sufficient ratings may not be available for the recommendation process. In this case, since the items are purchased rarely with different options it is difficult to collect sufficient ratings for an item. This problem is compared with cold start items with few ratings. There are cases where the item domain tends to be complex in terms of its varied properties, and it is hard to associate sufficient ratings. These types of cases can be addressed with knowledge-based Recommender systems, in which ratings are not used for the purpose of recommendations. Rather, the recommendation process is performed on the basis of similarities between customer requirements and item descriptions. The process is facilitated with the use of knowledge bases, which contain data about rules and similarity functions to use during the retrieval process. In both collaborative and content-based systems, recommendations are decided entirely by either the user's past ratings, ratings of his peers, or a combination of the two. Knowledge-based systems are unique, they allow the users to explicitly specify what they want. The inputs to different types of recommender systems are as follows.

- **Collaborative Filtering:** This approach accepts user and community ratings as inputs.
- **Content Filtering:** This approach uses item properties and user ratings as inputs for generating recommendations.
- **Knowledge Based Filtering:** This method of recommendation accepts item and user profiles and domain knowledge.

There are several ways in which recommendation problem is formulated. The two primary ways are as follows.

1. **Prediction version of problem:** This approach is to predict the rating value for a user-item combination or interaction. The input data is divided into training set and test set for training the model and validating the model respectively. We can consider m users and n items, this corresponds to an incomplete m * n matrix, where the specified (or observed or know) values are used for training. The missing (or unobserved or unknown) values are predicted using the test set. This problem is also referred to as the Matrix Completion problem because we have an incompletely specified matrix of values, and the remaining or missing values are predicted by the learning algorithm.
2. **Ranking version of problem:** Practically it is not necessary to predict the ratings of users for specific items or products in order to make recommendations to users. Rather, a person or any online merchant may recommend the top-k items for a particular user, or determine the top-k users as target for a particular item. The determination of the top-k items is more common and easy than the finding of top-k users. This problem is also known as the Top-k Recommendation problem, and it is the ranking formulation of the recommendation problem. In this case the absolute values of the predicted ratings are not necessary.

The first approach is more general than compared to the second one, since the second approach can be derived from first one by determining the user/item combinations and then ranking the predictions. Some goals of Recommender models are as follows.

- **Relevance:** The most common and obvious objective of a Recommender system is to recommend or suggest items that are relevant to the user at hand. Users are more interested to consume items that they find interesting or attractive.
- **Novelty:** Recommender systems are truly helpful when the recommended item is something that the user has not seen in the past or has never interacted with. For example, popular movies of a preferred generation would rarely be interesting to the user.
- **Serendipity:** In this case the items recommended are somewhat unexpected or surprising. Serendipity is different from novelty where the recommendations are truly surprising to the user, rather than simply something they did not know about before or has interacted with. For example a user may only be consuming items of a specific type, although a latent interest in items of other types may exist which the user might themselves find surprising. For example, if a new Indian restaurant opens in a neighbourhood, then the recommendations of that particular restaurant to a user who normally eats Indian food is novel but not necessarily serendipitous. On the other hand, when the same user is recommended Ethiopian food, and it was unknown to the user that such food might be interesting to him or her, then the recommendation is serendipitous. Serendipity helps in beginning a new trend of interest in the user. Increasing serendipity often has long-term and strategic benefits to the online merchants since it increases the possibility of discovering entirely new areas of interest. But at times the algorithms that provide serendipitous recommendations often tend to recommend irrelevant items. So, in such cases the long term and strategic benefits over-weigh the short term disadvantages.
- **Increasing recommendation diversity:** Recommender systems suggest a list of top-k items for particular set of users. When all these recommended items are very similar to each other, it increases the risk that the user might not like any of these items. On the other hand, when the recommended list contains items of different types, there is a greater chance that the user might like at least one of these items. Thus, diversity has the benefit of ensuring that the user does not get bored by repeated recommendation of similar items.

The recommendation process also meets some soft goals both from user perspective and business perspective.

- **User Perspective:** When we consider from the user side, recommendations can help improve the user satisfaction with the Web site. For example, a user who repeatedly receives relevant recommendations from Amazon.com or FlipKart.com will feel more satisfied with his or her experience and is more interested to use the site again. This improves user loyalty and further increases the sales at the site.
- **Business Perspective:** At the merchant end, the recommendation process can analyze the user or customer needs and help customize the user experience.

This process finally provides the users an explanation of why recommending a particular item is useful. For example, in the case of Netflix, recommendations about new movies are provided along with previously watched movies. Also Amazon provides recommendations for new accessories, books or clothes along with the list of items or products that a user has purchased or viewed before.

There is a wide diversity in the types of products or items recommended by different Recommender systems. Some Recommender systems, such as Facebook, do not directly recommend products to the customers. Instead these types of Recommender systems recommend social connections, which have an indirect benefit to the site and this increases usability. Some examples of Recommender systems are as follows.

1. **Amazon.com Recommender System:** Amazon.com is one of the popular Recommender systems, especially in the commercial setting. Originally it was a book e-retailer and consequently, now Amazon sells virtually all categories of products such as books, CDs, software, electronics, clothes and several other accessories. The recommendations in Amazon are provided on the basis of explicitly provided ratings, buying patterns, and browsing styles. The ratings in Amazon are specified on a 5-point rating scale, the lowest rating being 1-star, and the highest rating being 5-star. The customer-specific data can be easily collected when users are logged in with an account authentication process supported by Amazon. The purchase or browsing pattern of a user can be viewed as a type of implicit rating, which is different from explicit rating specified by user. There are many commercial Recommender systems which gives the flexibility of providing both explicit and implicit feedback.

2. **Netflix Movie Recommender System:** Netflix is one of the most popular movie rating Recommender systems. Netflix provides users the facility to rate the movies or the television shows that they have watched on a 5-point rating scale. Additionally, the user actions in terms of watching various items are also stored by Netflix database system. These ratings and actions are then used by Netflix to provide recommendations and also an explanation of why a particular item is recommended. This type of a system explicitly provides recommendations based on specific items that the users have viewed in the past. Based on such information a user can decide whether to watch a particular movie or not, this improves the user experience, loyalty and trust.

3. **FaceBook Friend Recommendation:** The different Social networking sites recommend potential friends to users in order to increase the number of social connections at the site. For example Facebook is a social networking Web site. The Friend Recommender systems have a slightly different goal compared to a product Recommender system. The product recommendation directly increases the profit of the merchant by facilitating product sales but a Friend Recommendations increase the social connections which in turn improves the experience of a user at a social network. The recommendation of potential friends enables better growth and connectivity of the network. This problem is called "link prediction". These types of recommendations are based on structural relationships rather than data ratings.

4. **Google News Recommender System:** The Google News personalization system recommends news to users based on their history of clicks. The clicks

provided by specific users are identified by Gmail accounts. In this type of Recommender system news articles are treated as items. The act of a user clicking on a news article is treated as positive rating for that article. These ratings are visualized as unary ratings, in which a process is present for a user to express their affinity for an item, but there is no process for the users to show their dislike. Furthermore, the ratings are implicit, because they are inferred from user actions rather than being explicitly specified by the user. Some recommender systems are **Amazon, Netflix, Group Lens, YouTube and FaceBook**.

3.2 Types Of Ratings

The recommendation algorithms are influenced by the systems used for tracking ratings. The ratings are often specified on a scale that indicates the specific level of like or dislike of an item. It is possible for ratings to be continuous values, for example the Jester joke recommendation engine in which the ratings can take on any value between -10 and 10. Usually, the ratings are interval-based, where a discrete set of ordered numbers are used to quantify like or dislike. These types of ratings are called **interval-based** ratings. For example, a 5-point rating scale might be drawn from the set (1, 2, 3, 4, 5), in which 1 indicates strong dislike and 5 indicates strong affinity towards an item. The number of possible ratings might vary with different systems which are used. The use of 5-point, 7-point, and 10-point ratings is particularly common. The 5-star ratings system, is an example of interval based ratings. In 5 point rating scale we interpret each rating as user's level of interest . This interpretation might vary across different websites. For example, Amazon uses a 5-star ratings system in which the 4-star point corresponds to "really liked it," and the central 3-star point corresponds to "liked it." Thus, in Amazon there are 3 favourable and 2 non-favourable ratings. This is called **Unbalanced Rating scale**. In some cases, there may be an even number of possible ratings, and the neutral rating might be missing. This approach is referred to as a **forced choice** rating system.

We can also use ordered categorical values such as (Strongly Disagree,Disagree, Neutral, Agree, Strongly Agree) in order to express our likes and dislikes. In general, such ratings are referred to as **ordinal ratings**. In **binary ratings**, the user may represent his/her like or dislike for the item and nothing else. For example, the ratings may be 0, 1, or unspecified values. There is also concept of **unary ratings**, in which there is a way for a user to specify his liking for an item but no possibility to specify a dislike. These types of ratings are particularly used in the case of implicit feedback data sets. In these cases, customer preferences are derived from their actions and not their explicitly specified ratings. For example, the buying behavior of a customer can be converted to unary ratings such as when a user buys an item, it can be viewed as a preference for the item. But at the same time the act of not buying an item does not indicate dislike towards an item. There are many social networks, such as Facebook, use "like" buttons, which provide the ability to express liking for an item but there is no mechanism to specify dislike for an item. We present a picture of 5 point Amazon rating system.

3.2.1 Explicit and Implicit Feedback

We explain the concept of explicit ratings with the following Figure: **??**. In our example, there are 5 users and 6 different items. Higher ratings indicate more positive

FIGURE 3.1: Example of Amazon 5 point-interval ratings [7].

feedback . The missing entries correspond to unspecified preferences. We represent the missing entries with "0". Here we present a small toy example. We have considered the users and the items in form of a rating matrix. A ratings matrix is sometimes referred to as a utility matrix. Although utility matrices are set to be the same as the ratings matrices, it is possible for an application to explicitly transform the ratings to utility values based on domain-specific criteria. Collaborative filtering approach is then applied to the utility matrix rather than the ratings matrix. Actually, in practice, Collaborative filtering method work directly with the ratings matrix. For cases in which the ratings are unary, the matrix is known as positive preference utility matrix since it allows only the specification of positive preferences. The matrix in Figure: **??** has different insights. In the matrix User2 and User5 are considered similar users since they have both given same rating to the same product (Item1). Again, the same pair of users (User2 and User5) are different with respect to the product (Item4), since they have different ratings for that product. The User1, User2 and User5 can be considered similar users since all of them have provided positive preference for the same product (Item4).

The Figure: **??** provides an example of a Unary matrix, where there are 6 users (U1.....U6) and 6 items (D1....D6) and it is possible for the non-zero entries to be arbitrary positive values. For example, they could correspond to the quantities of items bought by the different users. In general, unary matrices are formed by user actions for example buying an item, and are therefore also called implicit feedback matrices. Unary ratings have a significant effect on the recommendation algorithm because no information is available about whether a user dislikes an item. In the case of unary matrices, it is often good to perform the analysis in a simple way by treating the missing entries as 0s in the initial phase. However, the final predicted value by the learning algorithm is much larger than 0 especially if the item matches user interests. In fact, if the missing entries are not substituted with 0s, significant over-fitting is possible. This over-fitting indicates there is not sufficient difference between the observed ratings.

In explicit feedback matrices, ratings correspond to (highly discriminated) preferences, whereas in implicit feedback matrices, ratings correspond to (less discriminated) confidences.

Now, we describe the different evaluation techniques used to measure model performances.

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0	3	0	3	0
User 2	4	0	0	2	0
User 3	0	0	3	0	0
User 4	3	0	4	0	3
User 5	4	3	0	4	0

FIGURE 3.2: Example of Utility Matrices.

	D1	D2	D3	D4	D5	D6
U1	1			1		1
U2		1			1	
U3	1	1		1		
U4			1			1
U5				1	1	
U6	1		1			

FIGURE 3.3: Example of Unary Matrices.

3.3 Different Evaluation Techniques

The performance of a Recommender system can be measured by different evaluation techniques. In this thesis we have used some evaluation techniques which are as follows. In the context of our work we have tried to use simple formulas taking concepts from reference documents.

1. The first evaluation technique that we have used is known as Area Under ROC Curve (AUC). This evaluation technique is used for recommending items. For example in case of user-item pair (u, i), the preference of a user u towards an item i is considered as a function of time. Thus, the recommended item ranking for user u is time dependent. So for a held-out triple (u, i, tui), our evaluation process consists of calculating how accurately a particular item denoted by i is ranked for specific user denoted by u at a particular time instance tui. Our input data set is split into training/validation/test sets by uniformly sampling for each user u from user set an item i from item set(associated with a time stamp tui) to be used for validation (Vu) and another for testing (Tu). The rest

of the data (Pu) is used for training. The AUC evaluation methodology is then applied on the test set to measure the accuracy of the model. In our thesis we have used AUC to evaluate the performance of TVBPRPlus model. The formula of AUC taken from reference paper**[HeM16]** is given below.

$$AUC = \frac{1}{U}\sum_{u}\sum_{(i,j)\in E(u} \delta((\hat{x}_{u,i}(t_{u,i})) > (\hat{x}_{u,j}(t_{u,j}))) \tag{3.1}$$

In the Eq: **??** the indicator function within δ returns 1 if and only if the function is true and the evaluation process goes in a similar way for each user-item pair. We understand from above equation that the predicted value for item(i) which is a known item is given more importance than the predicted value of an unknown item (j).

2. The second evaluation technique that we have used to verify the performance of our models is known as Root Mean Squared Error (RMSE). This technique is based on rating prediction of items. In order to compute RMSE we need Observed values and Predicted Values. We first calculate the difference between Observed and Predicted values then we square the result of difference. We then divide the net result by number of data instances and finally do a square root of the result. In most of Industrial applications the RMSE values ranges between 0 and 1. We discuss this technique in details in later section. We have used this Evaluation technique to measure the accuracy of all the models namely TVBPRPlus, Timesvd++ and FARIMA. The formula for RMSE in simplified form is given in Eq: **??** .

$$RMSE = \sqrt{\frac{\Big(PredictedValue(\hat{x}) - ActualValue(x)\Big)^2}{N_{testSet}}} \tag{3.2}$$

In the Eq: **??** we compute the square of difference between Predicted and Actual values and then we divide it by number of data instances (N) and finally do a square root of net result. This evaluation methodology puts more emphasis on large errors compared to the mean absolute error. The lower the RMSE value the better is the prediction quality.

3. The third technique that we have applied in our thesis is known as Akaike information criterion(AICc). With the help of this evaluation technique we are able to determine the best fit model. We have used this evaluation methodology in our third model FARIMA. In FARIMA process with the help of AICc we are able to find the best values of Fourier order (K) and Auto-Regressive order (p) and Moving Average order (q). We discuss this technique in details in later section.

3.3.1 AUC (Area under the ROC curve)

This evaluation technique is widely used to measure the accuracy of different models. In this evaluation method a ROC curve is drawn to validate the performances of various models. This ROC curve consists of different Threshold values and is drawn by taking true positive values against false positive values. For example we consider

a positive class (P+) and a negative class (N-). Based on the position of the item in the respective classes there are different types of classification possible. If we consider an item to be a part of positive class then this type of classification is termed as true positive. But in case if the item is not a member of the class then the classification is termed as false positive. Similarly, if the target item is a part of negative class then the classification is called true negative but if the item is not a member of the negative class then it is termed as false negative. We have evaluated the performance of TVBPRPlus using AUC technique. We provide an example of how AUC is applied on TVBPRPlus below.

- **Example of AUC:** In TVBPRPlus model we have divided our input data into three parts. The first part is the train set which we use to train our model. The size of this data set is larger compared to other two parts. The other two parts are namely validation set and test set. We have uniformly sampled all the users in our model such that for each user there is an item (from set of known items) and associated time in validation set and test set. For the triplet (u, i, t - where u is the user , i is the item and t is the associated time) the evaluation technique calculates how perfectly an item (i) can be ranked for a user(u) at a specific time(t). This evaluation technology gives higher rank or more importance to a positive data instance than compared to a negative data instance. The higher the AUC value the better is the prediction quality of the model. The formula of AUC taken from reference paper [**HeM16**] is given in Eq: **??**.

$$AUC(u) = \frac{1}{|I_{u+}|\frac{I}{|I_{u+}|}} \sum_{(i)\in I_{u+}} \sum_{(j)\in \frac{I}{|I_{u+}|}} \delta\Big(\hat{x}_{u,i}t(u,i) > \hat{x}_{u,j}t(u,j)\Big) \tag{3.3}$$

The Eq: **??** computes the AUC value of one user. In the Eq: **??** Iu+ represents the positive item set and I/Iu+ refers to the negative item set. By negative item set we mean those items which user has not observed. Within the delta function we have the comparison of the predicted values. The predicted value of positive item (i) is given more importance compared to the predicted value of negative item(j). After calculating the AUC for 1 user we can do the same for all users. This is achieved by taking the sum of all AUC values and dividing it by the number of users present. This is formulated in a simple way below Eq: **??**.

$$Average \;\; AUC = \frac{Sum(AUC)}{nUsers} \tag{3.4}$$

In the Eq: **??** we compute average AUC of all users (nUsers). We calculate the sum of all AUC value and then we divide it by total number of users.

3.3.2 RMSE (Root Mean Square Error)

The root mean square error (RMSE) also known as Root Mean Square Deviation has been used as a standard statistical metric to measure model performance in meteorology, air quality, and climate research studies. RMSE measures the model accuracy. It does this by measuring difference between predicted values and the actual values. Let us consider a model with input X. The model predicts value 10, but the actual value is 5. The difference between predicted value and the actual observation is the

error term. The error term is important because we usually want to minimize the error. Then our prediction will be very close to the actual value, which improves the accuracy of our model.

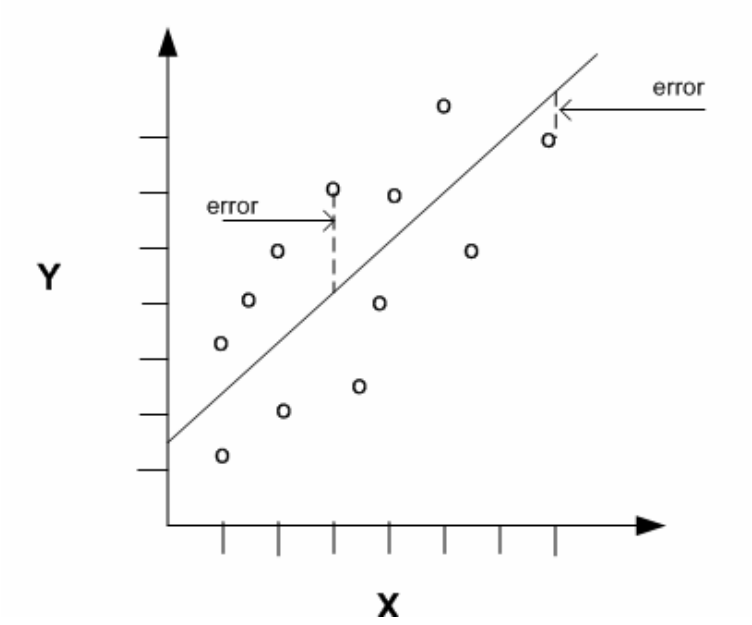

FIGURE 3.4: Graphical Plot of Error Term.

In the Figure: **??**, X axis represents the input data of the model (actual values) and the Y axis denotes the predicted value of the model. There are different ways we could apply to minimize this error term or minimize the absolute value of the error. The formula of RMSE is given in Eq: **??**.

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}} \tag{3.5}$$

In Eq: **??** we compute the error by differencing predicted value from actual value. Then the square of the error is divided by number of data instances in test set and then a square root of the net result is calculated.

In a good model, the RMSE value should be close for both the testing data set and the training data set. If the RMSE for our testing data is higher than the training data, there is a high chance that our model is overfit. In other words, our model performed worse during testing than training. In our work we have used RMSE to evaluate the performances of Timesvd++, TVBPRPlus and FARIMA models.

3.3.3 Akaike information criterion (AIC)

AIC helps to estimate the relative quality of many statistical models for a given data set. AIC helps to determine the best model among a set of models. So, AIC provides a means for model selection. When we use a statistical model to represent a process, some information or data gets lost in the process. AIC helps to estimate the amount of information lost by the model. It is observed that less the amount of information lost, the better is the quality of the model. AIC is formulated by Hirotugu Akaike and is widely used for statistical inference and is formulated in Eq: **??**.

$$AIC = -2Log(L) + 2(\phi + k + 1) \tag{3.6}$$

In the context of FARIMA model for given values of Fourier (k) , Auto-Reggressive (p) and Moving Average (q) orders the technique tries to maximize the log-likelihood while determining parameter estimates. In the Eq: **??** L is the likelihood of data. k=1 when ϕ is not 0 and k=0 when $\phi = 0$. Here ϕ is the sum of p and q . This evaluation technique is one of the best techniques for estimating order of parameters. When the sample data size is small, there is a substantial probability that AIC will select models that have too many parameters.Thus, AIC will overfit in such a situation. To address such potential of over fitting, AICc was developed. The AICc is given below in Eq: **??**.

$$AICc = AIC + \frac{2(\phi + K + 1)(\phi + K + 2)}{N - \phi - K - 2} \tag{3.7}$$

The Eq: **??** represents corrected AIC. Here N is the number of data instances. We obtain good models by minimizing the AICc. AICc is good for determining the best values of p term (Auto Regressive), q term (Moving Average) and Fourier order (denoted by K) in FARIMA model, but it is not appropriate for determining the value of d (differencing order). This is because the differencing changes the data on which the likelihood is computed, making the AIC values between models with different orders of differencing not comparable. So we need to use some other approach to choose d.

3.3.4 Application of AIC

In order to apply AIC we start with a set of candidate models. Then, we compute the AIC values of the corresponding models. There is always some amount of information loss in the process. We wish to select those models which minimize the information loss. Although we can not choose with certainty but we try our best to minimize the information loss. For example, let us consider there are three candidate models, whose AIC values are 100, 102, and 110. Then the second model is exp((102 - 100)/2) = 0.368 times as probable as the first model to minimize the information loss and the third model is exp((110 - 100)/2) = 0.007 times as probable as the first model to minimize the information loss. In this example, we would discard the third model from further consideration. We are then left with three options which are as follows. The quantity exp((AICmin - AICi)/2) is referred as the relative likelihood of model.

1. We gather more data, so that we can clearly distinguish between the first two models.
2. We simply conclude that the data is insufficient to support selecting one model from among the first two models.
3. We can also take a weighted average of the first two models, with weights proportional to 1 and 0.368, respectively.

Chapter 4

Prediction Models

In our thesis we have worked with three different prediction models which are TVBPRPlus, Timesvd++ and FARIMA. We have applied these models to predict the future trend of a specific product based on Amazon fashion dataset. The efficiency and accuracy of these models are evaluated using different evaluation methods such as RMSE, AUC and AICc. In this chapter we discuss the implemented models in details.

4.1 TVBPRPlus model

TVBPRPlus model is based on Collaborative Filtering. The model determines the users fashion aware personalized ranking based on the past feedback (user reviews) of users. The model is capable of capturing both the temporal and visual dynamics of data. The users are highly influenced by the visual styles of different items. These visual styles or features of various product vary over different fashion epochs. This model is able to uncover these visual features which has a direct impact on users choices. TVBPRPlus is visually aware, temporally evolving, interoperable, scalable and personalized. The model aims at accurately recommending an item (i) for a user (u) at time period (t) which will fit his purpose of need. For example if a user has purchased fashion products such as sunglasses and perfumes then the model will recommend the same user similar category of items such as a hat or a watch etc which will serve the needs of the user. TVBPRPlus model follows the basic formulation of matrix factorization, where the users and the items are presented in form of a matrix. A mapping is performed between the users and the items by observing the values in the matrix. This type of a matrix is known as **User-Item Interaction** matrix. For example if a cell corresponding to a user and an item contains 1 value or + sign (positive sign), then it is considered that the user has shown interest in the corresponding item. Such an item is called a positive item and if the value in a cell of the matrix is -1 or - sign (negative sign), then it is understood that the user has not preferred that particular item. Such an item is referred as negative item. The user-item matrix is divided into two sub matrixes. The values used in the matrix are assumed to be ratings given by users to the items. We present a picture of a user-item matrix from reference paper [**BPR**] in Figure: **??**. In this figure user specific pairwise preferences between a pair of items is created. The + sign indicates that the user prefers item i over item j and – sign indicates that user prefers j item over i. The question mark sign signifies that the user has not visited those items. Thus, the user-item interaction matrix helps to determine the user's preference towards different items.

We have used Latent Factor Model or matrix factorization to implement TVBPRPlus. According to the Latent Factor model the users and items are mapped to latent space.

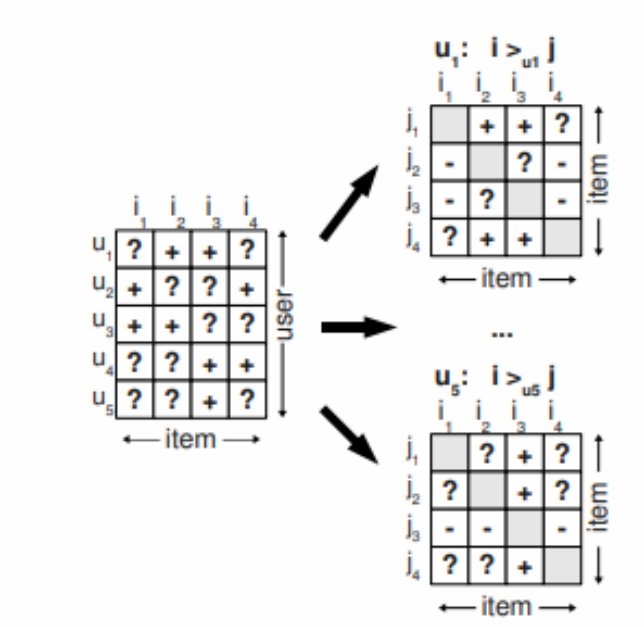

FIGURE 4.1: User-Item Interaction Matrix [**BPR**].

In the context of TVBPRPlus the formulas we have used are taken from reference paper [**HeM16**]. The preferences of the users towards items is predicted by using the formulation Eq: **??**.

$$\hat{x}_{u,i} = \alpha + \beta_u + \beta_i + < \gamma_u, \gamma_i > \tag{4.1}$$

The Eq: **??** involves the user/item biases and latent factors related to users and items. These latent factors include properties of item and user's personal interest towards those item properties. This prediction formula works well when there is enough data available but suffers in cold start situations where there is not enough ratings present for items. So, Eq: **??** is extended with other components of the model. TVBPRPlus captures the visual features of items and determines the user's interest towards various visual styles. So, we add visual components to the Eq: **??** in order to improve prediction accuracy. The formulation that involves the visual components is given in Eq: **??** .

$$\hat{x}_{u,i} = \alpha + \beta_u + \beta_i + < \gamma_u, \gamma_i > + < \theta_u, \theta_i > \tag{4.2}$$

In the Eq: **??** $\theta(u)$ represents the visual factors which characterizes the user (u) and $\theta(i)$ signifies the visual factors of the item(i).

It is very important to have explicit visual features of different items so that the model can estimate which visual style of an item is most predominant or mostly preferred by users during a certain fashion epoch. The explicit visual features of different Amazon products are extracted from their raw images with the help of Deep Convolutional Neural Network (CNN). We denote these visual features as fi. These visual features are mapped into a visual space of specific dimentionality denoted as K'. We represent the number of dimensions of these visual features as F. The value of F is set to 4096 by cross validation. We consider an Embedding matrix denoted as E

(K' * F) which linearly embeds the high dimensional features extracted from product images into a lower dimensional visual style space (K'). The visual features of items are formulated in Eq: **??** . It is denoted as θ_i.

$$\theta_i = E * f_i \tag{4.3}$$

In the Eq: **??** E (k' * F) represents the embedding matrix and fi denotes the explicit visual features extracted from the images of items using CNN. The embedding matrix E is able to uncover the visual factors of items and these visual factors highly influence user preferences.

The visual and as well as the Non-visual components of the model varies over time. The temporal dynamics related to visual and Non-visual factors are as follows.

1. **Time Dependent visual factors.**
2. **Temporally Changing visual bias.**
3. **Time Dependent Non- visual Factors.**

4.1.1 Time Dependent visual factors

In the fashion world new items with different visual styles always emerge in different fashion epochs. For example we consider the shoes of **"Puma"**. This shoe has a particular visual style but this visual may lose its significance when a new version of **"Puma"** shoes with new visual features evolves in market. So, we need to design the Embedding matrix (E) to be time dependent. The Embedding Matrix should be able to capture different types of fashion dynamics over time. This is called **Temporal Attractiveness Drift**.

- **Temporal Attractiveness Drift:** The visual features of different products gain or lose their importance in various time zones. Thus, the attractiveness of items change over time. In order to determine which visual feature is most attractive during a time period, it is necessary to capture this temporal drift in visual styles. So, the Embedding matrix is extended with temporal components. This time sensitive Embedding matrix is given in Eq: **??**.

$$E(t) = E + \Delta E(t) \tag{4.4}$$

In the Eq: **??** the stationary component of the model is captured by E and the time dependent part is handled by delta E. This Embedding matrix is capable of capturing different visual styles of items over several epochs.So, now the visual factors of items are extended with time dependent Embedding matrix. This is formulated in Eq: **??**

$$\theta_i(t) = E(t) * f_i \tag{4.5}$$

In the Eq: **??** the E(t) denotes time dependent embedding matrix and fi are the visual features extracted from images of items using CNN.

As new fashionable items or products evolve with time, the different visual dimensions are weighted differently by different users. For example users may pay less attention to a dimension describing bright colour if they are already tolerant to bright colours. So, a temporal weighing vector is considered to capture the users evolving emphasis on different visual dimensions. The formulation of $\theta(i)$ is extended with the weighing vector.

$$\theta_i(t) = E * f_i \odot \omega(t) \tag{4.6}$$

In the Eq: **??** $\odot$ refers to Hadamard product. So, now the visual factors related to items can be formulated in Eq: **??**.

$$\theta_i(t) = \underbrace{E * f_i \odot \omega(t)}_{\text{Base}} + \underbrace{\Delta E(t) f_i}_{\text{Deviation}} \tag{4.7}$$

The Eq: **??** is valid for global structures but to model or capture the personal interest of each user over time we introduce the concept of Temporal Personal Drift.

- **Temporal Personal Drift:** The personal interests of users vary over time. The interests of users are influenced by both outside evolving fashion trends and his/her personal taste. In this case we consider certain concepts of Timesvd++ model (discussed in details later). The time sensitive visual factors of users is formulated based on Timesvd++ model.

$$\theta_u(t) = \theta_u + sign(t - t_u) \cdot |t - t_u|^k \eta_u \tag{4.8}$$

The Eq:**??** explains the deviation of user u at time t from his or her mean feedback date tu.

4.1.2 Temporally evolving visual bias

In addition to the temporal visual factors we also use a time dependent visual bias. The visual bias helps to capture the changes in item appearance using low rank structures. So, we can use high rank structures to capture other variation for example per-user dynamics and per-dimension dynamics. This is computed as a product of <ß(t),fi>, ß(t) is a time dependent F dimensional vector. Visual Bias helps to improve the performance of the model and is also good for visualization. The formulation for the visual bias is given in Eq: **??**.

$$\beta(t) = \beta \odot b(t) + \Delta\beta(t) \tag{4.9}$$

4.1.3 Time Dependent Non- visual Factors

The Non-visual components also play a vital role in the prediction process of the model. In our context the Non-visual components are related to item profiles such as category of an item. There are two approaches to handle Non-visual factors. They are as follows.

- **Per item level dynamics:** There are various factors responsible for buying an item at different time periods. So, item properties are considered to be time dependent and denoted as ßi(t).
- **Per subcategory dynamics:** We consider a new parameter for addressing the category information of an item. This new parameter is denoted by ßci(t). This parameter helps to capture drifting user preferences towards a specific category of an item.

We finally integrate all the components to derive the final prediction rule. This is presented in a step wise way.

$$Temporal\ \ Biases = VisualBias + Non - VisualBias \tag{4.10}$$

$$Visual\ \ Bias = \Big(\beta(t), f_i\Big) \tag{4.11}$$

$$Non - Visual\ \ Bias = \Big(\beta_i(t) + \beta_c i(t)\Big) \tag{4.12}$$

$$User - Item\ \ Interaction = Non - Visual\ \ Interaction + Visual\ \ Interaction \tag{4.13}$$

$$Non - Visual\ \ Interaction = \Big(\gamma_u, \gamma_i\Big) \tag{4.14}$$

$$Visual\ \ Interaction = \Big(\theta_u(t), \theta_i(t)\Big) \tag{4.15}$$

$$Predictedvalue = \alpha + \beta_u + Temporal\ \ Biases + User - Item\ \ Interaction \tag{4.16}$$

The equations starting from Eq: **??** to Eq: **??** helps to recommend correctly an item for a certain user at a specific time.

4.1.4 Fashion Segmentation

Fashion tends to evolve in a non-linear or abrupt way. We divide our whole time-line into discrete parts or small time zones. These temporal parts are known as Epochs. We learn through our model which visual style or characteristic is most popular or predominant during a certain fashion epoch. In order to find the most visually prominent item feature during an epoch, we divide our time system into for example N epochs and assign certain parameter to each epoch. Then, we determine the preference of a user towards a certain visual style of an item during each epoch(ep).

4.1.5 Learning the model

Bayesian Personalized Ranking(BPR) is used for learning the model. BPR is a pairwise ranking optimization technique which is based on Stochastic Gradient Ascent.

The objective of our model is to rank items to users with which they have never interacted with. For this purpose we consider two sets of items - positive items (Pu) and negative items (I/Pu). Here I is the whole set of items. Positive items are those which the users have already visited and negative items are those with which users have never visited. For example we consider an item i which belongs to Pu and an item j which belongs to I/Pu. BPR determines the probability that a user prefers item i compared to item j with $\sigma(\hat{x}_{u,i} - \hat{x}_{u,i})$. Here, σ is used for comparative function and $\hat{x}_{u,i}$ is the predicted preference of user towards positive item (i) and $\hat{x}_{u,j}$ is the predicted preference of user towards negative item (j).

In order to learn the parameters we maximize log-likelihood function. The maximization is done by **Stochastic Gradient Ascent**. This is formulated in Eq: **??**.

$$\sum_{(u,i,j)\in D_s} log\sigma(\hat{x}_{u,i} - \hat{x}_{u,j}) - \frac{\lambda_\theta}{2}||\Theta||^2 \tag{4.17}$$

In case of 10 epochs the maximization function is formulated in Eq: **??**.

$$\hat{\Theta}\hat{\wedge} = argmax\ \ \Theta, \wedge \sum_{(u,i,j,t_{u,i})\in D_s} log\sigma(\hat{x}_{u,i}(t) - \hat{x}_{u,j}(t)) - \frac{\lambda_\theta}{2}||\Theta||^2 \tag{4.18}$$

The stochastic gradient ascent is applied on the train set of data for updating and learning the parameters. This learning process is repeated in an iterative manner until the convergence point is reached.

4.2 Timesvd++

The user preferences and item profiles are never static. They drift over time. New items continuously evolve in the economic market with newly added features. Thus, with the emergence of new products old products lose their significance. The users also easily get attracted towards new products. Thus, the preferences of users also change over time. A user who used to like an old type of **"Nike"** shoes can get more interested in a new version of the "Nike" shoes. There are many factors which cause user preferences to change over time. For example seasonal effects or specific holidays change user's shopping pattern. This can be explained with an example, a user who used to buy Sunscreen in summer will not buy the same in winter. Instead he/she will buy a moisturizer in winter. Thus, seasonal effects influences users shopping style. Similarly, on specific holidays most users buy more items of different categories compared to normal days. Sometimes change in family tradition influence user's preferences. Besides, these factors the personal interests of users also change over time. For example a user who used to like wearing T-shirts of **"Denim"** can now change his interest towards T-shirts of **"Peter England"**. The users also change his/her rating style. A user who used to give "3 star" rating for neutral preference may now give "2 star" rating for neutral choice. The users also differ from each other in rating patterns. For example UserA may give "3 star" as neutral choice and UserB may give "2 star" as neutral preference. Thus, the data related to users and items are dynamic in nature. This phenomenon is known as **Concept Drift**.

It is vital to consider the temporal dynamics of data when we design a Recommender system so that the current data is always provided to users. Thus, it is

necessary for us to build models which are temporal aware. One of such models is Timesvd++ which captures the drifting user interests and item properties. The model comprises of main two components namely, time dependent **Baseline predictors** which captures much of temporal dynamics in data and **Use-Item interaction** factors which are responsible for capturing the dynamic user behaviour and changing item profiles.

Timesvd++ is based on Collaborative Filtering (CF) which relies on past user behaviour such as past user transactions or user preferences. CF does not require domain knowledge and there is no need to explicitly create user/item profiles. When enough data is collected CF method is most preferred. In CF based technique we rely on past user purchase histories or preferences which helps us to uncover complex patterns which would otherwise be difficult to explore with only data attributes. This methodology is used by many popular commercial Recommender systems such as Amazon, Netflix. In order to give recommendations CF technique needs to compare different things such as items against users. CF supports two main approaches which are as follows.

1. **Neighbourhood approach.**
2. **latent factor model.**

We briefly discuss the different approaches of Collaborative Filtering.

4.2.1 Neighbourhood Model

This model mainly focuses on computing the relations between items and users. In an item-item approach the user preference towards an item is determined by the ratings of similar items given by the same user. This method is also known as Memory based Collaborative Filtering. These are one of the earliest Collaborative Filtering methods, in which the ratings of user-item combinations are predicted on the basis of their neighbourhoods. These neighbourhoods can be classified in one of two ways.

1. **User Based Collaborative Filtering:** In this case, the ratings provided by all the users are used in order to make recommendations for any target user for example User A. Thus, the main idea is to determine set of users, who are similar to the target User A. In this model we recommend ratings for the unobserved items or products of User A by computing the weighted averages of the ratings of the peer group. For example, if Alice and Bob have rated item like jeans in a similar way in the past, then one can use Alice's observed ratings on the item jeans (Lee or Denim) to predict Bob's unobserved ratings on this item. In general, we consider the k most similar users to Bob to make rating predictions for Bob. The Similarity functions are computed between the rows of the rating matrix (Rows signify users and columns signify products) to discover similar users.
2. **Item Based Collaborative Filtering:** In order to determine the rating predictions for target Item B by User A, we first need to determine a set S of items that are most similar to target Item B. The ratings in item set S, which are provided by the User A, are used to predict whether the user A will find item B interesting. For example Alice's ratings on similar fashion products for example sunscreen, lipstick etc and can be used to determine her rating on the item

eyeliner. Similarity functions are computed between the columns of the ratings matrix to discover similar items. The columns of the rating matrix signify the items.

The advantages of memory-based techniques or neighbourhood model are as follows.

1. These models are simple to implement.
2. The resulting recommendations achieved from these models are often easy to explain and understand.

But at the same time these models also suffer from some limitations such as.

1. The memory-based algorithms do not work very well with sparse rating matrices. For example, it might be difficult to identify enough number of similar users to Bob. In such cases, it is difficult to predict Bob's rating. Thus, these methods do not have full coverage of rating predictions. But the lack of coverage is not a big problem when we only consider the top K items.

4.2.2 Latent Factor Model

This model is also known as Model based Collaborative Filtering. In this method both users and items are mapped into latent factor space. This latent factor space characterizes products and users on factors automatically inferred from user feedback in order to explain the ratings. Some examples of this model include matrix factorization, decision trees, rule based methods, Bayesian methods. This model has a high level of coverage for sparse matrix. Although, the neighbourhood based models are preferred for their simplicity but they do not work well in all circumstances. The latent factor models perform better in all scenarios and have a good coverage even for sparse rating matrix. The latent factor models have gained popularity due to some popular Recommender systems like Netflix.com, Amazon.com and Flip Kart.com.

In our thesis we have implemented Timesvd++ model based on Factor model. Before moving into the details of Timesvd++ model we discuss about different approaches of Concept Drift.

4.2.3 Different Approaches of Concept Drift

The user or customer preferences changes with time as new products or services become available – this is a very good example of Concept Drift. There are circumstances where we face a more complex form of Concept Drift, where the interconnected preferences of users vary differently in many ways with different time instances. So, it is essential to keep track of multiple changing Concept Drifts. There are three approaches of handling concept drift which are as follows.

- **Instance selection approach:** This approach discards instances which are less relevant compared to the current instance. For example here we can consider the time window technique. In time window technique only recent instances are stored. But this approach suffers from a disadvantage, which is the approach gives same significance or value to all the instances in the same time window, disregarding other instances which might be important.

- **Instance weighting approach:** In this approach the instances are weighted based on the estimated relevance. Also in this approach a time decay function is used for under weighted instances which decay with time.
- **Ensemble Learning:** In this approach a group of predictors produce the final result. These predictors are weighted by their perceived relevance. For example the predictors which are more popular in the recent time span have higher values.

We follow some general rules or guidelines for capturing the drifting user preferences. Some of these rules are as follows.

- We must consider models which capture user behaviour or buying patterns throughout the time span not only the present user pattern. In this way we are able to extract data or information at each time point in the time line.
- We must adopt models which capture multiple changing Concept Drifts, since some drifts are user dependent and some are item dependent.
- The Concept Drifts which we collect for individual users and items must be combined in a single frame, so that there are interactions among all users and items to get a high level pattern.

4.2.4 Timesvd++ Implementation Based on Matrix Factorization

Matrix factorization method maps both users and items into a joint latent factor space of certain dimensionality which we denote by f. The user-item interactions are modeled as inner product in that space. The latent space tries to explain user ratings by characterizing users and items with factors relevant to the application domain (For example casual dress vs party dress for clothes). Each user is associated with a vector and each item is associated with another vector. The item vector measures the extent to which item i possesses these factors while the user vector measures the extent of interest of a user in these factors. Even in cases where independent implicit feedback is absent, one can capture the rating by accounting for which items users rate, regardless of their rating value. To this end, a second set of item factors is added, relating each item i to a factor vector denoted by Yi. The formulas that we have used in the context of Timesvd++ are taken from reference paper [**Koren10**]. The rating is predicted as an inner product of user and item factors.

$$\hat{r}_{u,i} = q_i^T p_u \tag{4.19}$$

In the Eq: **??** pu denotes the set of users (can also be considered as vector) and qi (can also be considered as vector) signifies the set of items. In order to learn the vectors pu and qi, we minimize the regularized squared error.

$$min \;\; q_*, p_* \sum_{(u,i,t)\in K} (r_{u,i} - q_i^T p_u)^2 + \lambda(||q_i||^2 + ||p_u||^2) \tag{4.20}$$

In the Eq: **??**, λ controls the extent of regularization. The minimization is performed by **Stochastic Gradient Descent**. Thus, the factor model captures the interaction between users and items in a well manner. The different parameters of the model are discussed below.

- **Baseline Predictors:** In Factor model most of the observed rating values is either due to users or items independently of their interaction. For example it is the nature of some users to give higher ratings to items and some items eventually receive higher ratings compared to other items. As described in the paper [**Koren10**] we use baseline predictors to capture those effects which do not involve user-item interaction. The baseline predictors capture the temporal dynamics in data, so it must be modelled accurately. The baseline predictors for an unknown rating are given in Eq: **??**.

$$b_{u,i} = \mu + b_u + b_i \tag{4.21}$$

In the Eq: **??**, μ denotes the average rating. The bu and bi denotes the user and item biases respectively. The baseline predictors must be integrated into the factor model, so we extend the Eq:**??** and the new Eq: **??** is formulated.

$$AUC = \frac{1}{U}\sum_{u}\sum_{(i,j)\in E(u}\delta((\hat{x}_{u,i}(t_{u,i})) > (\hat{x}_{u,j}(t_{u,j}))) \tag{4.22}$$

In order to achieve better accuracy we consider the implicit information about the items which are rated, regardless of their rating value. We consider a second set of item vector denoted by Yi. These new set of items helps to determine the user characteristics based on the items that they have rated. After incorporating this new set of items the formulation Eq: **??** looks as follows.

$$\hat{r}_{u,i} = \mu + b_u + b_i + q_i^T\left(p_u + |R(u)|^{-}\frac{1}{2}\sum_{(j)\in R(u)}\right) \tag{4.23}$$

In the Eq: **??** R(u) denotes the set of items rated by user. The bu and bi are the user and item biases. The user preferences are denoted by vector pu.The Baseline predictors are not fixed they vary over time. So we have added the temporal component to the Baseline Predictors.

- **Time changing Baseline Predictors:** Regarding baseline predictors there are two major temporal effects. The first one is related to items and the second one is related to users. The popularity of items drift with time. For example a particular type of women jacket is popular in winter season but it may lose its popularity in the summer season, a particular type of men shoes might become popular during some period of time but later it may lose its popularity. The users also change their ratings or preferences over time. For example a user who used to rate average items by 4 star rating, later changes his/her rating pattern by giving average rating as 3 star due to several reasons. So, the Baseline predictors are time dependent. The time sensitive equation of baseline predictors is formulated in Eq: **??**.

$$b_{u,i}(t) = \mu + b_u(t) + b_i(t) \tag{4.24}$$

In the Eq: **??** the user and item biases vary over time.The user effects change on daily basis. So, there are inconsistencies in user behaviour. Therefore, we need finer resolution to capture user biases. The item biases do not change on a daily basis. So, the item biases are easier to capture as they do not need fine resolution. To deal with item biases we consider splitting the item biases into time based bins. The choice of how to split the item biases into time based bins depends on the application. If we need enough rating per bin then larger bins are required and if we require finer resolution then smaller bins are needed. In our work we have used 30 bins spanning all days in the dataset. The item bias is split into stationary part and a time dependent part. The formulation for item bias is given in Eq: **??**

$$b_i(t) = b_i + b_{i,Bin(t)} \tag{4.25}$$

In the Eq: **??** a day t is associated with an integer Bin(t), where t can be any number from 1 to 30. The user biases are difficult to capture since they have short lived temporal effects. We use a liner function to capture the gradual drift in user behaviour. This is given by the following formulation in Eq: **??**.

$$dev_u(t) = sign(t - t_u) \cdot |t - t_u|^{\beta} \tag{4.26}$$

In the Eq: **??** the mean date of rating is denoted by tu for a user u and the user gives his/her rating on a particular day t. We express the time distance or difference of number of days between t and tu by | t - tu |. We consider the value of ß to be 0.4 by cross validation [**Koren10**]. Thus, the final equation for time changing user bias is given in the Eq: **??**.

$$b_u^{(1)}(t) = b_u + \alpha_u \cdot dev_u(t) \tag{4.27}$$

Apart from the gradual concept drift there are sudden drifts emerging as spikes from many applications. For example in any application we often find a particular user giving multiple ratings in a single day. This may be due to the mood of the user on that day. These effects are short lived and generally do not span more than a single day. In order to address such short lived effects we use a new parameter denoted by bu,t. This newly introduced parameter captures day specific variability. This parameter alone is not appropriate for capturing the user bias since it misses all the necessary data that span more than a single day. Thus, it serves as an additive component to the previously described user bias. This is formulated in Eq: **??**.

$$b_u^{(3)}(t) = b_u + \alpha_u \cdot dev_u(t) + b_{u,t} \tag{4.28}$$

Next, we combine the user and item biases which vary over time to derive the final equation for temporally varying baseline predictors. This is given in Eq. **??**.

$$b_{u,i}(t) = \mu + b_u + \alpha_u \cdot dev_u(t) + b_{u,t} + b_i + b_{i,Bin(t)} \tag{4.29}$$

Besides, temporal effects we can use the same methodology to capture periodic effects. For example some items are more popular in certain seasons such as the jackets and winter shoes are more popular in winter season and sunglasses and T-shirts are more popular in summer season. In order to capture these periodic effects we introduce a new parameter. This newly introduced parameter is modelled with item bias. This is presented in the Eq: **??**.

$$b_i(t) = b_i + b_{i,Bin(t)} + b_{i,period(t)} \tag{4.30}$$

In the Eq: **??** the periodic effects regarding the items is captured by the parameter $b_{i,period(t)}$.

The periodic effects are also observed in case of users. For example some users have different shopping patterns in the weekends or on specific holidays. This periodic effect is captured by using a new parameter in a similar way like in case of items. This is given by the following Eq: **??**.

$$b_u(t) = b_u + \alpha_u \cdot dev_u(t) + b_{u,t} + b_{u,period(t)} \tag{4.31}$$

In the Eq: **??** periodic effect regarding users is captured by the parameter $b_{u,period(t)}$.

There is another temporal effect which is related to baseline predictors, it is the changing scale of user ratings. For example different users have different rating scale and even a single user can also change his/her rating style. In order to address this type of a temporal effect we add a time dependent scaling feature to the baseline predictors denoted by cu(t). This is given by the Eq. **??** .

$$b_{u,i}(t) = \mu + b_u + \alpha_u \cdot dev_u(t) + b_{u,t} + \Big(b_i + b_{i,Bin(t)}\Big) \cdot c_u(t) \tag{4.32}$$

In the Eq. **??** cu(t) represents the day specific variability. This multiplicative factor helps to reduce the RMSE value, which further improves the accuracy of the model. The temporal dynamics also affect the user preferences or in other words they affect the interaction between users and items. The preferences of users vary with time. For example a user who used to like wearing sunglasses of "RayBan" can now change his/her preference and switch to sunglasses of "RainBow" brand. In order to model user preferences we use a vector represented by p(u) and it is taken as function of time. This is given in Eq: **??**.

$$p_{u,k}(t) = p_{u,k} + \alpha_{u,k} \cdot dev_u(t) + p_{u,k,t} \tag{4.33}$$

In the Eq: **??**, puk captures the stationary portion of this factor and puk,t captures the day specific variability and k= 1....f. Finally, we incorporate all the model parameters and deduce the final equation of Timesvd++. This is given in Eq: **??**.

$$\hat{r}_{u,i} = \mu + b_u(t) + b_i(t) + q_i^T \left(p_u(t) + |R(u)|^{-\frac{1}{2}} \sum_{(j) \in R(u)} y_j \right) \tag{4.34}$$

In the Eq: **??** all the model parameters drifts over time. We divide our input data set into train set and test set. We use the train set to train and learn the model. The learning is done by stochastic gradient descent and then we validate the model using the test set. We compute RMSE on the test set to measure the accuracy of the model. We validate the performance of our model using different experiments, for example we vary the dimensionality (f) of latent space. We take different f values like 10, 20, 50, 100 and 200. We observe that the RMSE value decreases with increasing dimensionality. The lower the RMSE the better is the model accuracy. There are other experiments performed on the model to test its efficiency and accuracy. We discuss these experiments in details in later section.

4.3 Fourier Assisted ARIMA or FARIMA

In the economic market the user preferences and item profiles change frequently with the emergence of new products in the market, this leads to dynamic modelling of users and items in the personalized Recommender systems. In the previous methods we focused on product level modelling, where user preferences are modelled based on their purchase histories (Collaborative Filtering). But it is observed that product level modelling suffers from lack of data since some users may rate few items compared to the huge set of items in the whole system. So, now we move from product level modelling to feature level modelling, where we extract the domain knowledge or explicit features in a specific product domain. The feature level modelling grants the ability to predict the user preferences through time series analysis. The expanded feature space of this methodology helps to make cold start recommendations for users with few purchasing records. The product level modelling approach discards the cold start users for example users with less than 20 reviews. But in real world applications this is not feasible since these cold start users constitute a major part of the system.

Our Recommender systems must be designed in such a way so that it behaves intelligently to recommend items or products that fit the current user needs. For example in summer users normally use sunscreens to protect their skin from strong UV radiation of the sun but in winter users buy moisturizing cream due to cold and dry weather. If the Recommender system always considers the purchasing histories for recommending products to users instead of dynamic modelling of user preferences then the system will recommend products like sunscreen in winter. An item also receives new reviews from users which make it profile drift dynamically over time. Thus, we must build our Recommender systems such that it will recommend items which fit the purpose of users.

In order to track new fashionable items and provide recommendations timely we analyze the time series on daily basis. In this context, we use the Auto-Regressive Integrated Moving Average (ARIMA) model. The ARIMA model is widely used for monthly, yearly, quarterly time series analysis, however, the year-long period (around m = 365) of daily time series makes the ARIMA models in-feasible in practice due to the huge computational cost. In order to address this problem we develop Fourier-assisted Auto-Regressive Integrated Moving Average (FARIMA) to capture the seasonal component of the time series and use ARIMA to handle the residuals.

We first provide a brief description of time series decomposition analysis and then gradually move to the working of FARIMA model.

4.3.1 Time Series Decomposition

Time series analysis and modelling have many business and social applications. It is extensively used to forecast company sales, product demand, stock market trends, agricultural production etc. A time series is a sequential set of data points which is measured typically over successive times. It is mathematically defined as a set of vectors x(t),t = 0,1,2,... where t represents the time elapsed. We consider the variable x(t) as a random variable. The measurements in a time series are always in a proper chronological order. The time series containing records of a single variable is termed as uni-variate. But if records of more than one variable are considered, it is termed as multivariate. A time series can be continuous or discrete. In a continuous time series observations are measured at every instance of time, whereas a discrete time series contains observations measured at discrete points of time. For example temperature readings, flow of a river, concentration of a chemical process etc. can be recorded as a continuous time series. On the other hand population of a particular city, production of a company, exchange rates between two different currencies may represent discrete time series.

The fundamental idea for time series analysis is to decompose the original time series (sales, stock market trends, etc.) into several independent components. Typically, business time series are divided into the following four components. The components are as follows.

1. **Trend:** The overall direction of the series like upwards, downwards etc. The general tendency of a time series is to increase, decrease or remain stagnate over a long period of time. This is termed as Trend. Thus, it can be said that trend is a long term movement in a time series. For example, series relating to population growth, number of houses in a city etc show upward trend, whereas downward trend can be observed in series relating to mortality rates, epidemics, etc.

2. **Seasonality:** The regular patterns which occur monthly,yearly, quarterly or weekly.The important factors causing seasonal variations include climate and weather conditions, customs, traditional habits, etc. For example sales of ice-cream increase in summer, sales of woolen cloths increase in winter.

3. **Cycle:** Long term business cycles which do not have any regular patterns. The cyclical variation in a time series describes the medium-term changes in the series, caused by circumstances, which repeat in cycles. The duration of a

cycle extends over longer period of time, usually two or more years. Most of the economic and financial time series show some kind of cyclical variation.

4. **Irregular Residual:** Random noise left after extraction of all components from time series and it does not have any pattern at all.

The combination or interference of all these components produces the final series. We decompose the time series since it is much easier to forecast the individual regular patterns produced through decomposition of time series than the actual series. This is similar to forecasting the individual sine waves (For example A, B, C, and D) instead of the final irregular pattern produced through the product of these four sine waves. We consider our time series data as a combination of trend, seasonality and reminder or residual. We decompose the time series data to understand the underlying patterns.

1. **Trend Removal:** We first try to remove the trends from data. One of the most commonly used procedures to remove trends is Moving Average. The Moving Average procedure used here is different from the Moving Average of ARIMA model. We use Moving Average to remove all trends or zigzag motion from data in order to produce a steady trend. The formulation for Moving Average procedure is given in Eq: **??**.

$$MovingAverage = \frac{\sum_{i=-m}^{m} Y_{t+i}}{2m} \tag{4.35}$$

Trends in the time series can also be removed by differencing the time series data which we have described in the later section.

2. **Seasonality Removal:** The time series data may contain seasonal variation or seasonality. Seasonality are cycles that repeat regularly over time. Thus, a cyclic structure in a time series may or may not be seasonal. If it consistently repeats at the same frequency, then it seasonal otherwise it is not seasonal and is simply referred as a cycle. There are several types of seasonality such as daily, weekly, monthly, quarterly, yearly. The simplest approach to identify whether there is any seasonality in our time series data is to plot and review the data. If the plotted data reveals periodicity or repeats at constant rate, then we are sure that the data contains seasonality. The understanding of seasonality component in a time series helps to improve the model performance. Once we are able to identify the seasonality then we can remove it from time series. The process of removing seasonal component is called Seasonal Adjustment or De-seasonalizing. A time series where the seasonal component is removed is referred as seasonal stationary. There are some sophisticated methods to study and extract seasonality from time series data. We discuss one of the such methods.

    - **Seasonal Adjustment with Differencing:** The seasonal adjustment is done with the help of differencing. For example, if there is a seasonal component at the level of one week, then we can remove it by subtracting it from the value of last week.

Seasonality plays a very important role in the time series estimation and prediction. So, we model the seasonal component of time series for each feature such as sunscreen or moisturizer with Fourier series, where we consider the time period m =365. The seasonality (S(t)) of a time series is given by the Eq: **??** taken from reference paper [**Zhang0ZLLZM15**].

$$S(t) \approx \hat{S}(t) = a + Asin\left(\frac{2\pi t}{m} + \varphi\right) \tag{4.36}$$

In the eq: **??**, for the seasonal series S(t) of each feature f, we compute the percentage of energy that a simple one-order Fourier can consume out of total energy. This is given by the Eq. **??** taken from reference paper [**Zhang0ZLLZM15**].

$$P_f = \sum_{t=1}^{n} \frac{|\hat{S}_f(t)|^2}{|S_f(t)|^2} \tag{4.37}$$

In the Eq: **??**, Pf refers to the Feature percent and Sf(t) refers to the seasonal component. According to Parseval's theorem [**Zhang0ZLLZM15**], a full-order Fourier trans-formation from time domain to frequency domain will preserve the energy, like, pf = 1, while pf < 1 for an under ordered Fourier due to the loss of some information.

3. **Irregular Remainder or Residual:** After removing all the necessary information or data from time series we are simply left with the residual, this is referred as white noise. The residual has no specific pattern, it is simply trend less and irregular in nature. For example, we can consider the screen of a television without any signal. The screen displays some curves without any specific pattern.

   Let us consider a multiplicative time series composed of trend, seasonality and irregular residual. Trend is removed from the time series using the following Eq: **??**.

$$Seasonality_t \times Remainder_t = \frac{Y_t}{Trend_t} \tag{4.38}$$

After trend is removed from time series only irregular remainder is left which has no specific pattern. This is given in Eq: **??**.

$$Remainder_t = \frac{Y_t}{Trend_t \times Seasonality_t} \tag{4.39}$$

Now, we movie to the description of Fourier Assisted ARIMA (FARIMA) model. The FARIMA model is based on the ARIMA process. So, we first discuss about ARIMA model then move to working of FARIMA.

4.3.2 Autoregressive Integrated Moving Average (ARIMA)

ARIMA stands for Auto-Regressive Integrated Moving Average models. ARIMA is a forecasting methodology that estimates the future values of a series based entirely on its own inertia. Its main application is in the area of short term forecasting requiring at least 40 historical data points. It works best when the data exhibits a stable or consistent pattern over time with a minimum amount of outliers. Sometimes called Box-Jenkins method after the name of original authors, ARIMA really performs very well when the data is reasonably long and the correlation between past observations is stable. Exponential smoothing and ARIMA methodology are the two most widely used approaches to time series forecasting. While exponential smoothing models are based on a description of the trend and seasonality in the data ,ARIMA models aim to describe the auto correlations in the data. For example the working of ARIMA model can be compared with the making of sugarcane juice. Sugar cane juice is prepared by crushing a long piece of sugar cane through the juicer with two large cylindrical rollers. However, it is difficult to extract all the juice from a tough sugar cane in one step hence the process is repeated multiple times. In the first go, a fresh sugar cane is passed through the juicer and then the residual of the sugar cane that still contains juice is again passed through the juicer many times till there is no more juice left in the residual. The ARIMA model also works in a similar manner. We can consider our time series data as a sugar cane and ARIMA models as sugar cane juicer. The idea with ARIMA models is that the final residual should look like white noise (exhibiting no specific pattern) otherwise; there is juice or information available in the data to extract.

ARIMA is a combination of 3 parts i.e. AR (Autoregressive), I (Integrated), and MA (Moving Average). A convenient notation for ARIMA model is ARIMA (p,d,q). Here p, d, and q are the levels for each of the AR, I, and MA parts. Each of these three parts is an effort to make the final residuals display a white noise pattern (or no specific pattern at all). In each step of ARIMA modelling, time series data is passed through these 3 parts.

In order to understand the concept of ARIMA, we need to first understand the concept of stationarity and the technique of differencing the time series in order to make it stable and consistent over time.

4.3.3 Condition of Stationarity

The first step of applying ARIMA methodology is to check for stationarity of time series data. "Stationarity" implies that the time series remains at a fairly constant level or stable (consistent) over time. A stationary time series is one whose properties do not depend on the time. If a trend exists, as in most economic or business applications, then our data is not stationary. The data should also show a constant variance in its fluctuations over time. This is easily seen with a series that is heavily seasonal and growing at a faster rate. In such a case, the ups and downs in the seasonality will become more dramatic over time. In general, a stationary time series will have no predictable patterns in the long-term. Time plots will show the series to be roughly horizontal (although some cyclic behaviour is possible), with constant variance. Without these stationarity conditions being met, many of the future calculations associated with the process cannot be computed.

4.3.4 Condition of Differencing

If a graphical plot of the data indicates non stationarity, then we should "difference" the series. Differencing is an excellent way of transforming a non-stationary series to a stationary one. In order to make the time series data stable we subtract time series with its lagged series to extract trends from the data. In the first pass of ARIMA model, we extract trend(s) from the original time series data. Differencing is one of the most commonly used mechanisms for extraction of trends. Here, the original series is subtracted from its lagged series e.g. November's sales values are subtracted from October's values to produce trend-less residual series. The formulae for different orders of differencing are given in Eq: **??** and Eq: **??** .

$$Nodifferencing = Y'_t = Y_t \tag{4.40}$$

$$FirstOrderDifferencing = Y'_t = Y_t - Y_{t-1} \tag{4.41}$$

In the Eq: **??** the time series is already stationary, so no differenceing is needed. In the Eq: **??** the current series Y_t is subtracted from its lagged series Y_{t-1}. In most of the cases after first order differencing the trends are removed from the series. If there are still trends present in the series then the series is again differenced. This is called second order differencing. It is very important to know that over differencing of series can also increase trends in the series.

4.3.5 Auto-correlations

Autocorrelations are numerical values that indicate how a data series is related to itself over time. In other words, it measures how strongly data values at a specified number of periods apart are correlated to each other over time. The number of periods apart is usually called the "lag". For example, an autocorrelation at lag 1 measures how values 1 period apart are correlated to one another throughout the series. An autocorrelation at lag 2 measures how the data two periods apart are correlated throughout the series. Autocorrelations may range from +1 to -1. A value close to +1 indicates a high positive correlation while a value close to -1 implies a high negative correlation. These measures are most often evaluated through graphical plots called "correlagrams". A correlagram plots the auto- correlation values for a given series at different lags. This is referred to as the autocorrelation function and is very important in the ARIMA method.

In order to determine a proper model for a given time series data, it is necessary for us to carry out the Auto-Correlation Function (ACF) and Partial Auto-Correlation Function (PACF) analysis. These statistical measures reflect how the observations in a time series are related to each other. For modeling and forecasting purpose it is often useful to plot the ACF and PACF against consecutive time lags. These plots help in determining the order of AR and MA terms. For, a given time series [x(t),t = 0,1,2, ...], the Auto-covariance at lag k is given by the Eq: **??**.

$$Y_k = Cov\Big(xt, x(t+k)\Big) = E\big[(xt-\mu)(x(t+k)-\mu)\big] \tag{4.42}$$

In Eq: **??**, μ is the mean of time series. The Auto-correlation coefficient is given by the Eq: **??**.

$$P(k) = \frac{Y(k)}{Y(0)} \tag{4.43}$$

The value of auto-correlation coefficient ranges from -1 to +1. There is another measure, known as the Partial Auto-correlation Function (PACF) is used to measure the correlation between an observation k period ago and the current observation. The ACF plots helps to determine the order of MA process and the sample PACF plot helps in identifying the maximum order of an AR process.

4.3.6 Auto-Regressive Models (AR)

In Auto-Regressive models we extract the influence of the previous period's values on the current period. After the time series data is made stationary through the integrated (I) pass or Differencing, the AR part of the ARIMA model gets activated. As the name Auto-Regression suggests, here we try to extract the influence of the values of previous periods on the current period e.g. the influence of the September and October's sales value on the November's sales. This is done through developing a regression model with the time-lagged period values as independent or predictor variables. An AR model with only 1 parameter may be written as the Eq: **??**.

$$X(t) = A(1) * X(t-1) + E(t) \tag{4.44}$$

In the Eq: **??**, X(t) is the time series data, A(1) is the Auto-Reggressive parameter of order 1, X(t-1) is the time series data lagged 1 period and E(t) is the random error of the model. Form the Eq: **??** we understand that any given value X(t) can be explained by some function of its previous value denoted by X(t-1), plus some unexplainable random error denoted by E (t). If required the series could be related to more than just one past value. This is described in the Eq: **??**.

$$X(t) = A(1) * X(t-1) + A(2) * X(t-2) + E(t) \tag{4.45}$$

The Eq: **??** now indicates that the current value of the series is a combination of the two immediately preceding values, X(t-1) and X(t-2), and some random error E(t). Our model is now an Auto-Regressive model of order 2.

4.3.7 Moving Average Models (MA)

In Moving Average model we extract the influence of the previous period's error terms on the current period's error. A second type of Box-Jenkins model is called a Moving Average model. Although these models look very similar to the AR model, the concept behind them is quite different. Moving average parameters relate what happens in period t only to the random errors that occurred in past time periods, i.e. E(t-1), E(t-2), etc. A moving average model with one MA order may be written as the Eq: **??**.

$$X(t) = \mu + B(1) * E(t-1) + E(t) \tag{4.46}$$

In the Eq: **??** the symbol μ represents mean of the series. The term B(1) is called an MA of order 1. The above model simply explains that any given value of X(t) is directly related only to the random error in the previous period and to the current error. As in the case of Auto-Regressive models, the Moving Average models can be extended to higher order structures covering different combinations and moving average lengths. Conceptually a Moving Average model is a linear regression of the current observation of the time series against the random shocks of one or more prior observations. Fitting an MA model to a time series is more complicated than fitting an AR model because in the former one the random error terms are not foreseable.

ARIMA methodology allows models to be built that incorporate both AR and MA parameters together. By combining the AR and MA models we obtain a much powerful forecasting tool that simulate the time series data and produce a more accurate prediction result. An ARIMA model is usually stated as ARIMA (p, d, q). This represents the order of the Auto-Regressive components (p), the number of differencing operators (d), and the highest order of the Moving Average term (q). For example, ARIMA (2, 1, 1) means that you have a second order Auto-Regressive model with a first order moving average component whose series has been differenced once to induce stationarity.

Thus, we understand that ARIMA methodology is applied for stationary time series data. However in practical cases there are many time series such as those related to economic and business which show non-stationary behavior. Time series, with trend and seasonal patterns, are also non-stationary in nature. So, in order to handle non-stationary time series data we convert it to a stationary series by applying **finite differencing** of the data points. As described earlier in finite differencing we substract the current value of time series data from its previous lagged value. Then, we apply AR and MA process in order to get future predictions of data points.

4.3.8 White Noise

In every step we extract information from time series data such as in the first step we perform integration or differencing of the series data to remove trends from data and make the series stable. Next, in second step we perform Auto Regression to find the impact or influence of past observations on the current data. In the third step we find the influence of previous error on the current error. After extracting all the necessary information from the data by different steps we are left with the residuals which we refer as White Noise. This residual is not having any pattern or trend; if it does then we still have information to extract.

A crucial step in an appropriate model selection is the determination of optimal model parameters. One option we have is the sample ACF and PACF plots, calculated from the training data should match with the corresponding theoretical or actual values There are other widely, used measures for model identification are Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC). In our thesis work we have used AICc (Based on AIC) for optimal model selection. We discuss this in details in the below section.

4.3.9 Estimation and Order Selection

The main challenge in classical Box-Jenkins ARIMA model is to determine how many AR and MA parameters to include. In order to determine the best fit model or the best AR and MA parameters, we use Akaike's Information Criterion (AIC). The formula for AIC is discussed before in Chapter 3. It is important to note that these information criteria tend not to be good guides to selecting the appropriate order of differencing (d) of a model, but only for selecting the values of p and q. This is because the differencing changes the data on which the likelihood is computed, making the AIC values between models with different orders of differencing not comparable. So we need to use some other approach to choose d, and then we can use the AICc to select p and q.

4.4 Fourier Assisted ARIMA Model or FARIMA

In order to predict a future data instance in ARIMA model requires data or information from at least one period ago to capture the seasonal and cyclic effects in data fluctuations. The ARIMA model works fine for monthly, yearly, quarterly data. But the ARIMA model is in-feasible due to the huge computational cost for estimation and prediction when the time period is of 365 days, for daily recommendation in a time series. In order to address this problem of daily time series we enhance the ARIMA model with Fourier terms. We adopt the Fourier series to model the seasonal component of time series and use the ARIMA process to model the dynamic error. The ARIMA model enhanced with Fourier series is given by the Eq: **??** and eq: **??**.

$$\hat{X}(t) \approx \underbrace{a + \sum_{k=1}^{k} \left[\alpha_k sin\Big(\frac{2\pi kt}{m}\Big) + \beta_k cos\Big(\frac{2\pi kt}{m}\Big) \right]}_{\text{F(t)}} + \underbrace{ARIMA(p,d,q)(t)}_{\text{E(t)}} \tag{4.47}$$

$$\hat{X}(t) = F(t) + E(t) \tag{4.48}$$

In the Eq: **??** and Eq: **??** X(t) is the fitted time series, F(t) is the K-order Fourier and E(t) is the dynamic error modelled by ARIMA process. We represent the enhanced ARIMA model with Fourier in Figure: **??** which we have implemented in our graphical tool based on Amazon fashion product (Sunscreen).

We describe our FARIMA process in form of an Algorithm: **??**. The inputs to the algorithm are the original time series data, the possible orders of Fourier series and ARIMA components. The algorithm determines the best fit model, which means the best values of p, q and K by minimizing the AICc. The value of d is obtained by differencing the time series. We select those values of p and q which corresponds to the minimum value of AICc. In the final output we receive the fitted or predicted time series data which is a summation of F(t) (the Fourier component) and E(t) the

(ARIMA component).

Algorithm 1: FARIMA Algorithm

Input : Time series X(t), maximum Fourier order(K), ARIMA order(p,d,q), Period n =731
Output: Predicted Series X′(t)

fourierMinAICc← 10000;F′(t);
for $i \leftarrow 1$ **to** K **do**
  $(F(t), AICc) \leftarrow fourierTransform(K, X(t), n)$
  if $AICc < fourierMinAICc$ **then**
    $F'(t) \leftarrow F(t)$
    $fourierMinAICc \leftarrow AIC$
  end
end
arimaMinAICc← 10000;E′(t);
for $j \leftarrow 1$ **to** p **do**
  for $l \leftarrow 1$ **to** q **do**
    $(E(t), AICc) \leftarrow ARIMAProcess(p, d, q, [X(t) - F'(t)], n)$
    if $AICc < arimaMinAICc$ **then**
      $E'(t) \leftarrow E(t)$
      $arimaMinAICc \leftarrow AIC$
    end
  end
end
return $X'(t) \leftarrow (F'(t) + E'(t))$

Thus, Fourier Assisted ARIMA or FARIMA helps us to leverage time series analysis for dynamic daily aware recommendation and helps to overcome the data insufficiency problem of Product Level modelling by extracting product features from user text reviews. The FARIMA approach makes time series analysis on daily resolution computationally feasible and also used for the purpose of time series prediction.

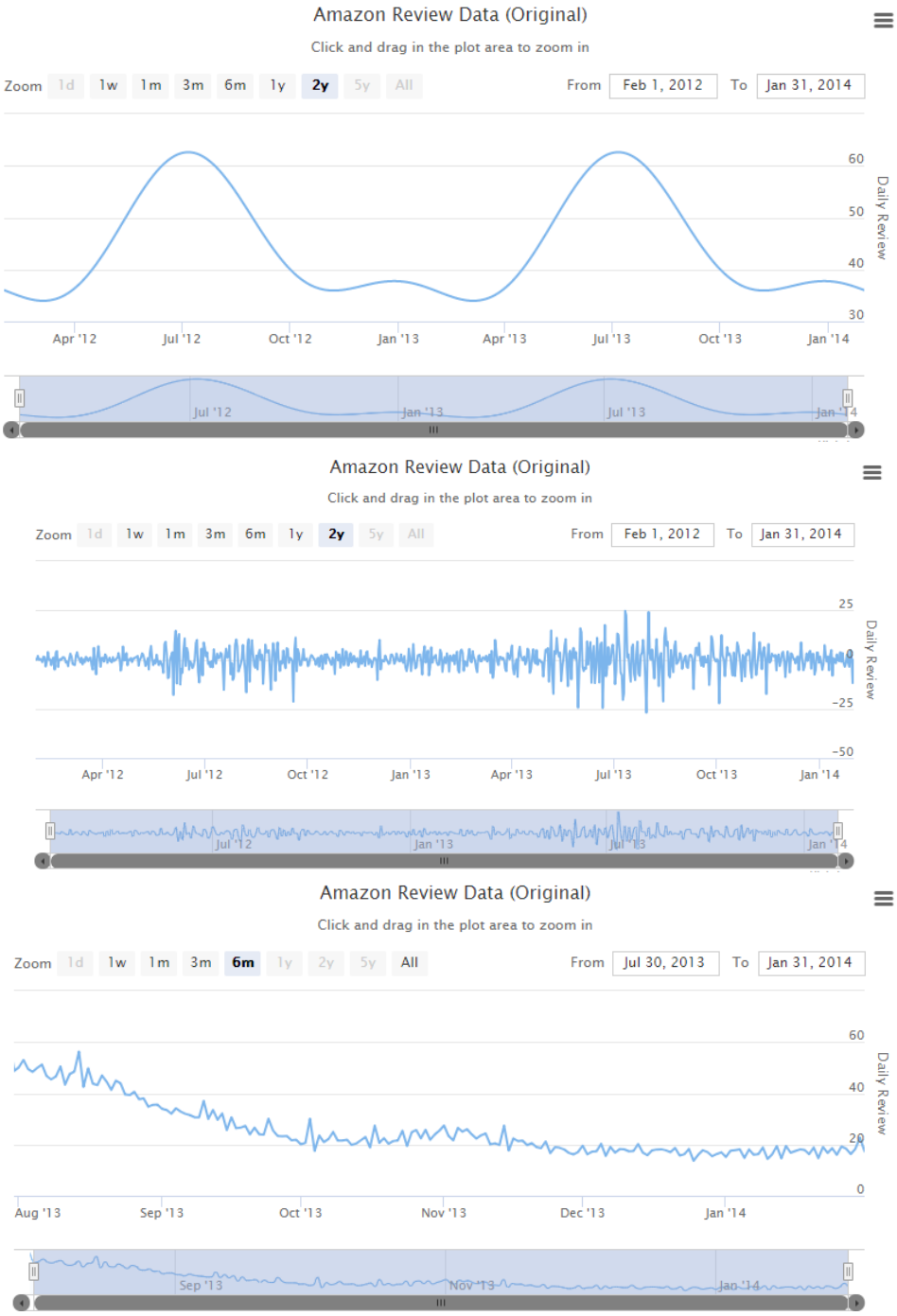

FIGURE 4.2: Graphical Plot Fourier Series, ARIMA Series and Predicted Series.

Chapter 5

Experiments

In our thesis we have evaluated the performance of our implemented models by running different experiments on them. We check the efficiency and accuracy of our models in different settings by tuning various parameters. These experiments evaluate the power of our implemented models and also help us to understand when or under which circumstances a specific model performs better/worse compared to other models. In this chapter we discuss several experiments and our observations/conclusions from these experiments. We begin by first describing the dataset design we have chosen to run the experiments, then we compare all the three models (TVBPRPlus, Timesvd++ and FARIMA) in terms of their efficiency and accuracy under different settings.

The details of the machine where we have performed our experiments are as follows.

1. **Processor:** Intel (R) Core(TM) i7-3770 CPU @ 3.40 GHz
2. **Operating System:** Unbuntu 14.04
3. **Ram:** 16Gb

5.1 Dataset Design

In this set of experiments we use the Amazon dataset. The data consists of the category such as clothing, shoes, jewelery, beauty cream and accessories. The data is extracted from Complete.json.gz file which has latest inclusion of data from Amazon and has 142.8 million of records. One dataset is created by extracting time series data from the above mentioned file from Amazon. Time series data is must for FARIMA algorithm. From this dataset, another 2 datasets are created which are also time series data. Table**??** provides the details about the datasets.

We first compare all the three models in terms of efficiency and Accuracy and then move to some experiments specific to individual model.

Dataset	Users	Actions	Actions/user	Timestamp (Years)
D1	15520	25458	1.64	Feb 2012-Jan2014
D2	10399	17106	1.64	Feb 2013-Jan2014
D3	4726	7420	1.57	Aug 2013-Jan 2014

TABLE 5.1: Dataset Description.

5.2 Analyzing the Performance of the Models

In this section we compare the efficiency and accuracy of our models by tuning various input parameters. This helps us to understand under which conditions a certain model performs its best and when the performance of the model degrades. We are also able to learn which model shows best performance among others under specific settings.

We compare our models by tuning these different parameters. We discuss our evaluation process in details in below sections. We use Dataset (D1) to perform the following experiments.

5.2.1 Comparing the Efficiency and Accuracy of the Models

We first compare Timesvd++ and TVBPRPlus in terms of accuracy under different epochs.The results of our experiments are plotted below in Figure **??**.

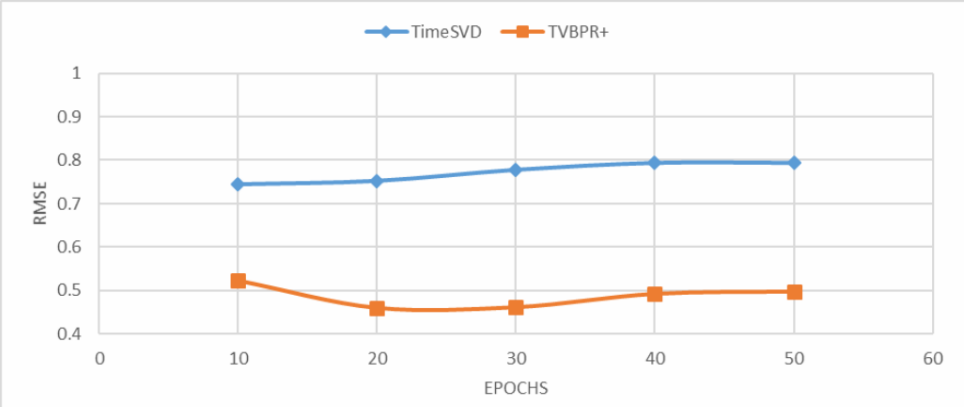

FIGURE 5.1: Comparing the accuracy of Timesvd and TVBPRPlus under different epochs.

We observe from Figure **??** that TVBPRPlus performs better in epoch ranging between 20 and 30. After epoch 30 as the epochs increase the accuracy of the models slowly decreases. In case of Timesvd there with increasing epochs the RMSE increases. Thus, with increasing epochs the performance Timesvd decreases. We compare the accuracy of Timesvd and TVBPRPlus under different latent factors The results are given in Figure **??**.

We observe from Figure **??** that the TVBPRplus model performs better when the latent factors are between 30 and 50. As the latent factors are increased above 50 the accuracy of the model slowly decreases. In case of Timesvd as the latent factors are increased the RMSE decreases, thus the performance of Timesvd Increases. We compare the accuracy of Timesvd and TVBPRPlus under different Learn Rates (L). The results are given below in Figure **??**. We observe from Figure **??** that both the models Timesvd and TVBPRPlus performs better when the learn rate is 0.005. We compare accuracy of Timesvd, TVBPRPlus and FARIMA under different dataset sizes. We provide the results in below Figure **??**.

We observe from Figure **??** that when the dataset size is increased the performance of TVBPRPlus and FARIMA increases. In case of Timesvd the there is slight change in performance (performance increases) with increased dataset size. These models show better performance with more data. We compare efficiency of Timesvd, TVBPRPlus and FARIMA under different dataset sizes. We provide the results in below Table: **??**.

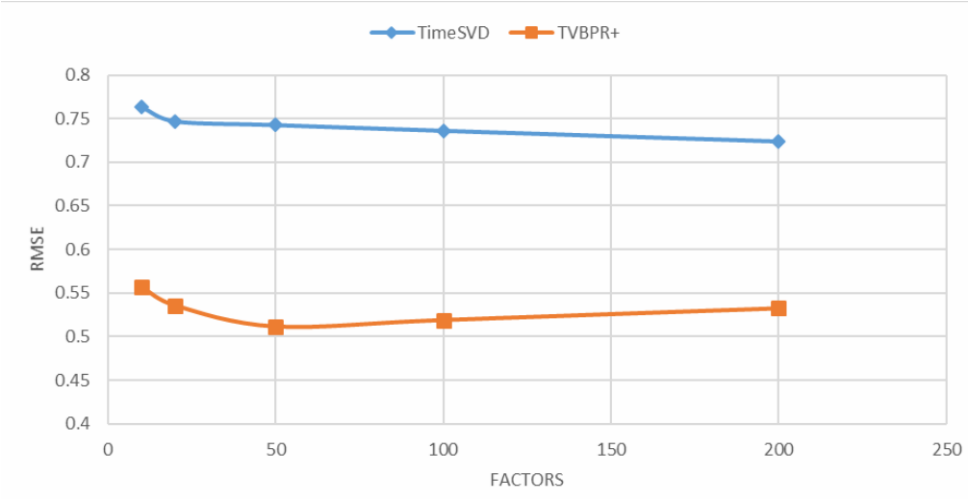

FIGURE 5.2: Comparing the accuracy of Timesvd and TVBPRPlus under different latent factors.

FIGURE 5.3: Comparing the accuracy of Timesvd and TVBPRPlus under different learn rates.

We observe from Table: **??** that as the dataset size is increased the runtime of the models increases. TVBPRPlus model takes more time to run compared to Timesvd and FARIMA under same dataset size, since in TVBPRPlus there are visual factors involved which takes time to update.

We now perform some specific experiments on certain individual models. We discuss them in details in the below section.

5.2.2 Analyzing Accuracy of Timesvd++

In this subsection we have evaluated the Timesvd++ model by tuning some hyper-parameters. We test the accuracy of Timesvd under different Beta values. The results are given below plot in Figure **??**.

The beta value is used in calculation of user biases in the Timesvd model. We observe from Figure **??** that when the beta value between 0.3 and 0.4 the Timesvd model performs better compared to other beta values. We test the accuracy of Timesvd under different General Gamma values. The results are given in below Figure **??**.

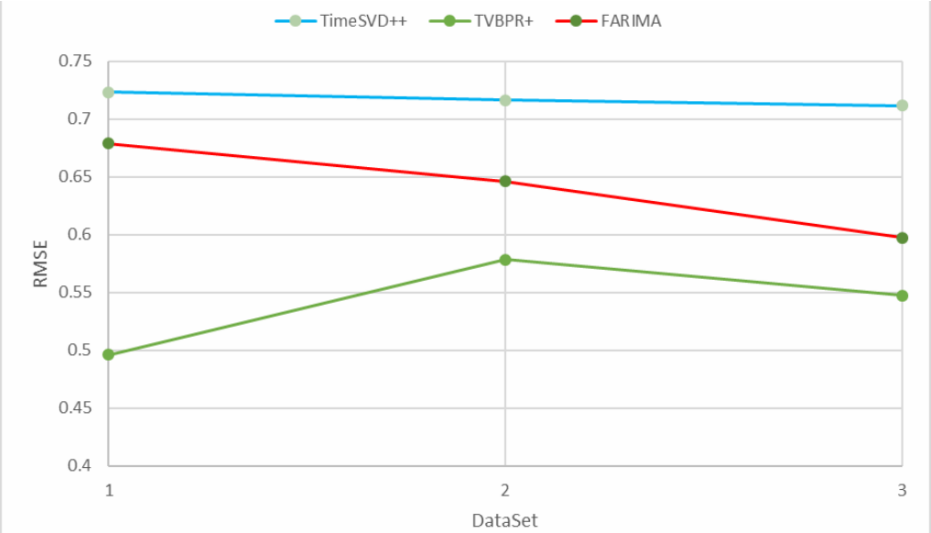

FIGURE 5.4: Comparing the accuracy of Timesvd, TVBPRPlus and FARIMA under different dataset sizes.

Dataset	Timesvd (Efficiency secs)	TVBPRPlus (Efficiency secs)	FARIMA (Efficiency secs)
D1	12	774	12
D2	8	437	10
D3	6	229	6

TABLE 5.2: Comparing the Efficiency of Timesvd, TVBPRPlus and FARIMA under different dataset sizes.

The Gamma values are used for learning and updating different model parameters. We observe from Figure **??** that as the Gamma values are increased the accuracy of the Timesvd model increases.

5.2.3 Analyzing the Accuracy of TVBPRPlus

In this subsection we perform some specific experiments on TVBPRPlus. We test the accuracy of TVBPRPlus under different Visual Factors (K').The results are given in below in Figure **??**. We observe from Figure **??** that as the visual factors are varied the performance of TVBPRPlus also varies. TVBPRPlus shows better performance when the visual factors are between 20 and 30. We test the accuracy of TVBPRPlus under Bias Regularizer. We provide our results in below Figure **??**.

We observe from Figure **??** that as the value of Bias Regularizer is increased the performance of TVBPRPlus increases.

Thus, the experiments help us to learn how the models perform in terms of accuracy and efficiency under different circumstances. We are able to analyze when a certain model performs best and when the model performance degrades. By comparing the models under same hyper-parameters we are able to understand which model performs better compared to others under certain settings. We also learn the best parameters to be used for a specific model.

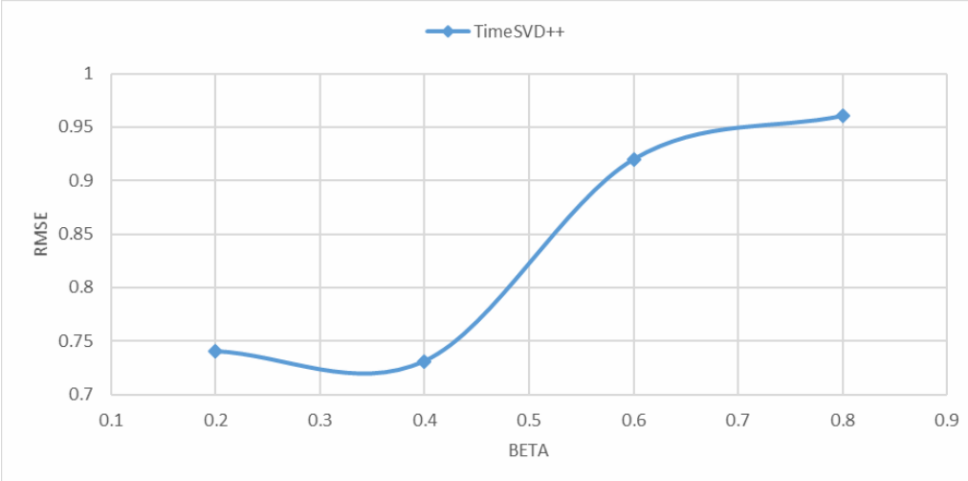

FIGURE 5.5: Comparing the accuracy of Timesvd under different beta values.

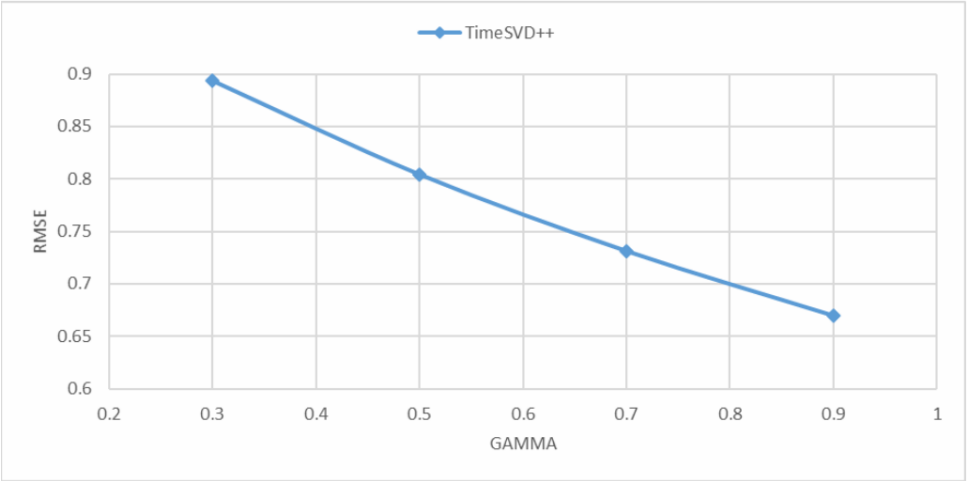

FIGURE 5.6: Comparing the accuracy of Timesvd under different Gamma Values.

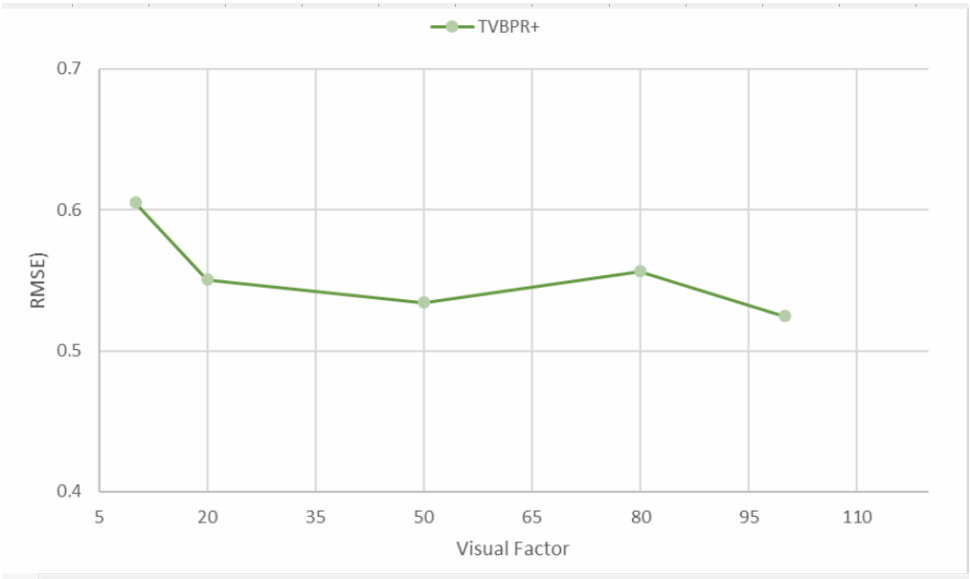

FIGURE 5.7: Comparing the accuracy of TVBPRPlus under different Visual Factors.

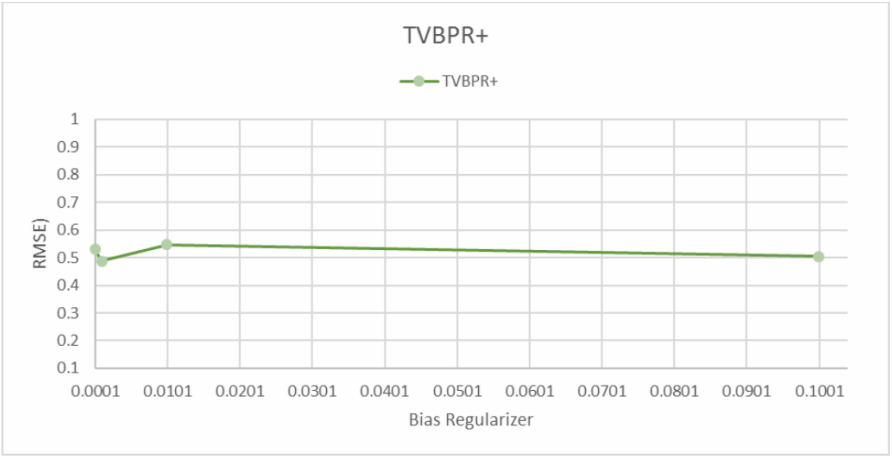

FIGURE 5.8: Comparing the accuracy of TVBPRPlus under different Bias Regularizer.

Chapter 6

Trend Prediction through Graphical Tool

6.1 Features of the Tool

In our thesis we have developed a graphical tool which helps to visualize the results of prediction by using fashion dataset of Amazon. We have created time series data of different items (for example Sunscreen, Sunglass, Jacket, Gift Product and Moisturizer) by extracting the number of reviews for the products. We forecast the future trend of the products by predicting a part of the time series data. We visualize this trend prediction through our implemented graphical tool. Some features of our tool are as follows.

1. With the help of our tool we are able to browse and visualize the fashion data.
2. The tool helps us to visualize the prediction results of our implemented techniques namely TVBPRPlus, Timesvd++ and FARIMA for different fashion products.
3. We have also used our tool to visualize the taxonomy of some selected products from Amazon dataset for example Sunscreen, Sunglasses, Moisturizer, Jacket and Gift Product.
4. With the help of our tool we are able to zoom and visualize data (Both Time series data and Predicted Series) for different options such as 1 week, 1 month, 3 months, 6 months, 1 year and 2 years.
5. We have also used the tool-tip option in the graphical plots. This helps users to observe the values of Original time series and predicted series.
6. In the graphical plots of different techniques we have used different colors to show time series data (Blue), part of time series to be predicted (green dotted form) and predicted series (red). This is helpful for the users who are visualizing the predicted results to differentiate between original series and predicted series.

6.2 Technologies Used

We have used Html, CSS, Bootstrap, Javascript and Jquery to develop our graphical tool. We now discuss the details of our implemented tool in the following sections.

6.3 Description of our Implemented tool

In our thesis work we have created a well designed graphical tool to visualize the prediction results of different implemented techniques (Timesvd++,TVBPRPlus and FARIMA). In this tool we have created a product taxonomy based on Amazon fashion data which includes different types of items such as Sunglasses, Sunscreen etc. The users has the option to select different products to visualize the product trend. We now describe the different parts of our tool staring from Home to Trend Prediction page.

- **Description of Home Page:** Our Home page is well designed and we have given it a good attractive look for the users. We can navigate to different pages such as Categories page and Help File page from the Home page.

- **Description of Categories Page:** In our Categories Page we provide a taxonomy of different Amazon fashion products in form of side panel. Users can easily navigate to respective page by clicking the product links to visualize the predicted results of the product for three different techniques namely TVBPRPlus, Timesvd++ and FARIMA. We provide a picture of this page below in Figure **??**.

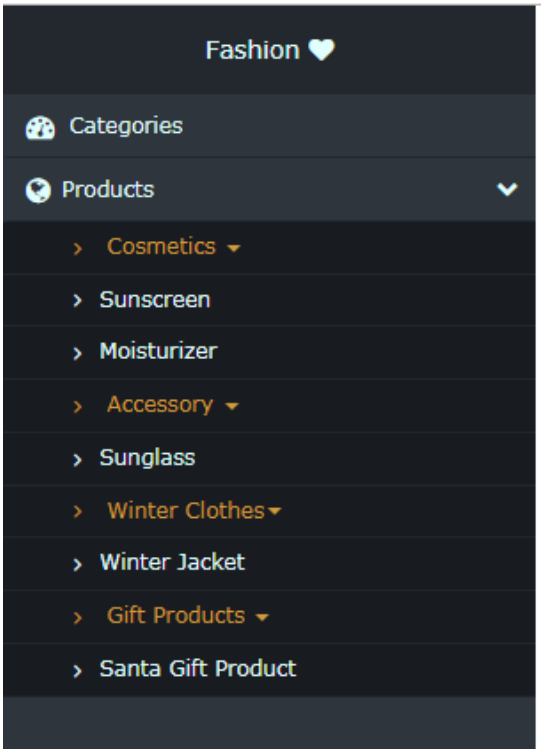

FIGURE 6.1: The Categories Page.

- **Description of Trend Prediction Page:** In the context of the report in this page we show the predicted results of a specific product among other products for example Sunscreen. We show the fashion trend of Sunscreen product for three different proposed techniques namely TVBPRPlus, Timesvd++ and FARIMA. This page consists of three different options or tabs for three implemented techniques such as, TVBPRPlus, Timesvd++ and FARIMA. Users by clicking on the

respective technique or tab can visualize the Time series data and the Predicted series of the respective methodology. For the purpose of easy understanding we have marked our Time series data in Blue color and the part or range of Time series which we are going to predict in future in dotted green color. We show our predicted results in red color. In the graphical plot we have datetime along the x-axis and reviews of the product on the Y-axis. We also show the item (Sunscreen) in the same page for the purpose of ease of use. By clicking the item on this page will show the data series, for example if we click on Sunscreen Time series (TS) option will show our time series data for Sunscreen in blue and if we click on Sunscreen Predicted option will show the predicted result in red. We also provide a scroll bar with the graphical plot to scroll and visualize the data history. We provide a picture of this page in Figure **??**, Figure **??** and Figure **??** which shows the time series and predicted results of Sunscreen product under different techniques.

FIGURE 6.2: The Trend Prediction Page for Sunscreen when TVBPRPlus Technique applied.

FIGURE 6.3: The Trend Prediction Page for Sunscreen when Timesvd++ Technique applied.

We can also zoom and visualize the data in details for 1 week, 1 month, 3 months , 6 months, 1 year and 2 years for all the implemented techniques. We provide a picture of this page in Figure **??**.

In this way we can zoom and visualize the data in details.

In the trend Prediction page we also have a collapsible side panel for more information. In this side panel we can click on different techniques such as TVBPRPlus,

FIGURE 6.4: The Trend Prediction Page for Sunscreen when FARIMA Technique applied.

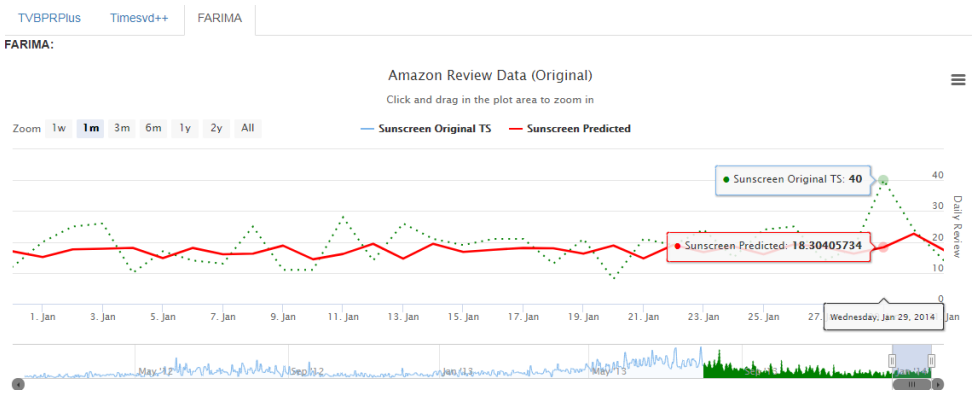

FIGURE 6.5: The Trend Prediction Page of FARIMA showing data of 1 month by Zooming for Sunscreen Product.

Timesvd++ and FARIMA for more detail information. For example by clicking on FARIMA from side panel we are navigated to a page which describes FARIMA process in details. In this page the users can click on the different options from select menu such as Predicted data , Fourier Data (FCT) and ARIMA data (ECT) in the navigation bar at the top to visualize the respective data series. This process is maintained in a similar way for TVBPRPlus and Timesvd ++. We provide a picture of this page in Figure **??**.

In the Figure **??** the users by clicking on FARIMA link from the side menu navigates to a new page showing FARIMA details. The picture is given below in Figure **??**.

In Figure **??** the users can choose different options from the top Menu bar to visualize the different components of FARIMA process. We provide pictures of these pages in Figure **??** and Figure **??**.

Similarly if a certain user wants to know more about the techniques such as TVBPRPlus and Timesvd++, then he/she can click on the collapsible side menu for respective options in Figure **??**. We provide pictures of these pages in Figure **??** and Figure **??**.

we have designed our specific item (Sunscreen) to be clickable. For example when we click on Sunscreen Time Series option we can visualize the time series data for the product in blue color and when we click on Sunscreen predicted option we can visualize the predicted series in red color. This pattern is same for all the techniques. We give pictures in Figure **??** and Figure**??**.

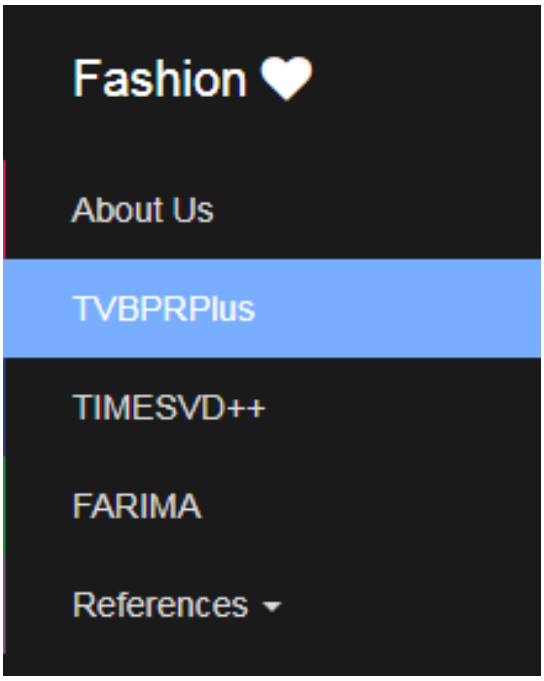

FIGURE 6.6: The collapsible side panel for extra information.

In the Figure **??** the Sunscreen Time series option is selected in blue color.

In the Figure **??** the Sunscreen Predicted series option is selected in red color.

In our tool we also show tool-tip option where the users can observe the values in both original time series and predicted series. We provide a picture of this in Figure **??**.

In our tool we have also incorporated the feature to download , save as an image (JPEG or PNG form) and view data-table of the graphical plots of different techniques. We provide an image of this in Figure **??**.

Thus, the graphical tool helps us to visualize the future trend prediction of a product. With the help of the tool we can browse and visualize the data history and also zoom to view data in details. We can also select products from related taxonomy to see their related taxonomies. Finally we can download or save in different forms the predicted results.

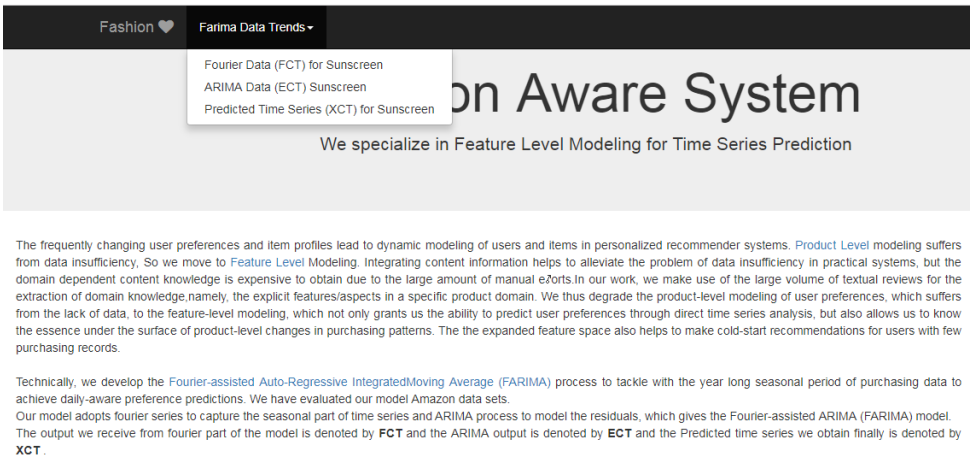

FIGURE 6.7: The FARIMA details page showing extra information.

FIGURE 6.8: The FARIMA page showing Fourier Series of Sunscreen product.

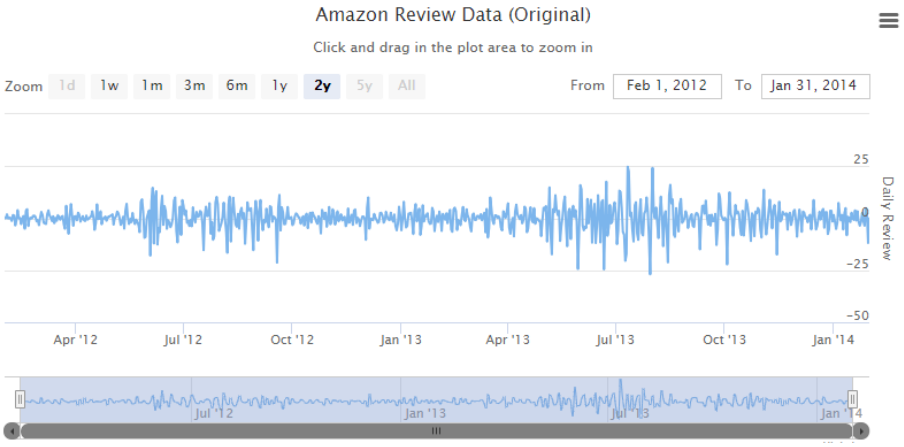

FIGURE 6.9: The FARIMA page showing ARIMA Series of Sunscreen product.

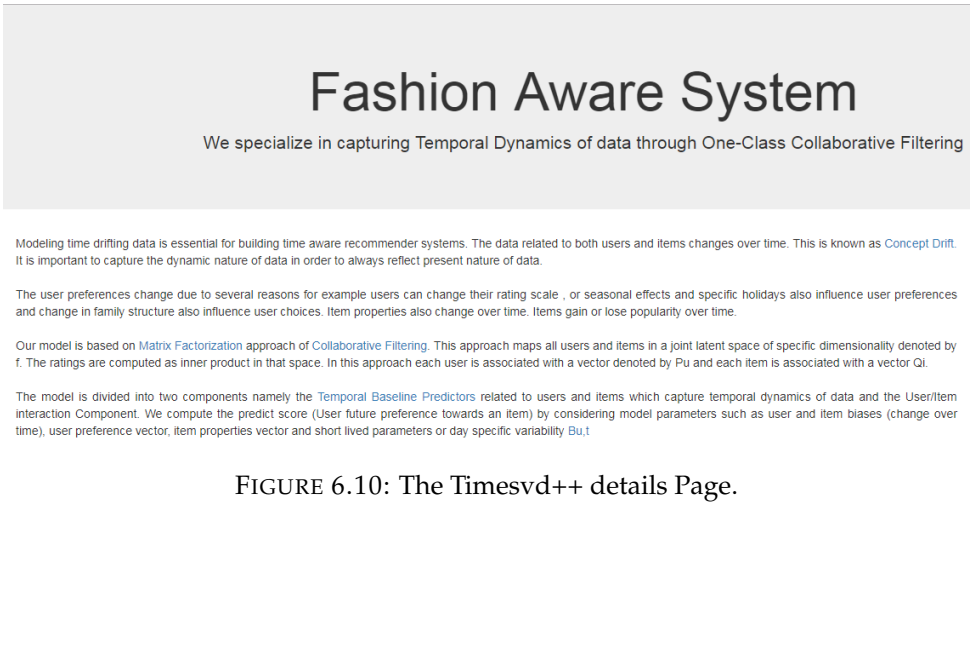

FIGURE 6.10: The Timesvd++ details Page.

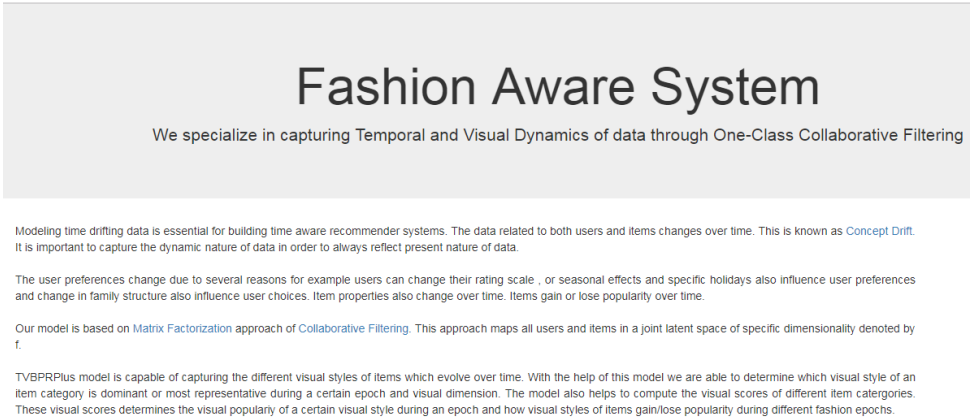

FIGURE 6.11: The TVBPRPlus details Page.

FIGURE 6.12: The Sunscreen Time Series when Original TS option selected.

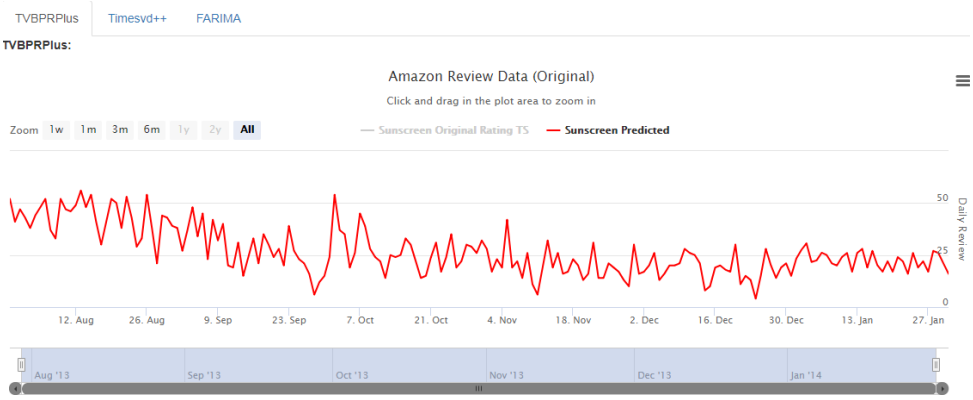

FIGURE 6.13: The Sunscreen Predicted Series when Predicted option selected.

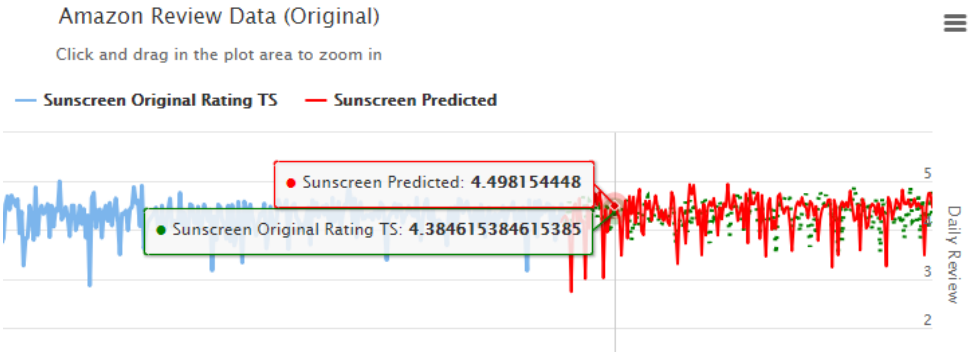

FIGURE 6.14: The Tool-tip option.

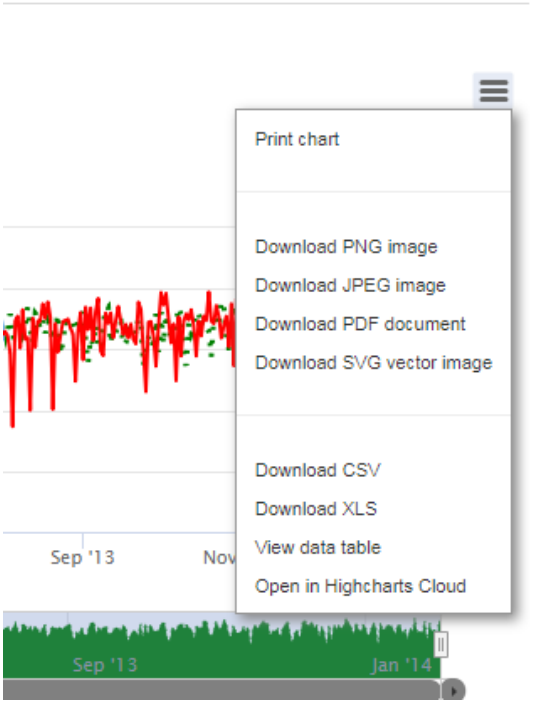

FIGURE 6.15: The Different options applied to download or save data.

Chapter 7

Discussion

In this chapter, we discuss about the different challenges we faced in our work and the solutions that we have adopted.

In our thesis we have worked with visual and temporal aware prediction models. Capturing the temporal dynamics of data is one of the greatest challenges of time aware systems. Users easily get attracted towards new items. This causes user preferences to change over time. The item profiles also change over time for example some items become popular during certain fashion epochs. Thus, data related to both users and items is dynamic in nature. Moreover, it is really challenging for us to capture these changes of data within a single model or framework, where we have interconnected users and items to each other to identify communal patters of behaviour. We could have used multiple separate models instead of mapping all users and items in one single model, but this reduces prediction accuracy since much of the data is lost in the process. In order to overcome this challenge, the solution we adopted was to capture temporal dynamics along the whole time period within a single framework. We have used Matrix Factorization method, which maps all users and items in a latent space of certain dimentionality (f) and ratings are modelled as inner products of users and items interactions in the latent space. This solution proved to be very useful in improving the quality of prediction.

Building visually aware Recommender system which is scalable, interoparable and temporally evolving is another challenge we face in our thesis. Users are mostly attracted towards the visual appearance or style of products. Thus, visual dimensions or visual styles play a key role in predicting user's future interests. Different visual styles of items become visually dominant during certain fashion epoch. Capturing these temporally evolving visual dimensions was really challenging for us. So, we have considered K' dimensional **visual decision factors** which encode visual compatibility between users and items. The visual styles of items gradually evolve over time. This presents more challenge in modelling the visual dimensions of items, since an item may be favoured during specific time period and disliked during others. In order to handle such a challenge we have considered the visual decision factors as function of time. The solutions that we have adopted are very useful in capturing the temporally evolving visual factors and improved our prediction accuracy.

In our thesis we have also worked with Feature Level Modelling, where we have extracted explicit features of a specific product from textual reviews. We have implemented Fourier Assisted ARIMA (FARIMA) model, which grants us the ability to predict user preferences through time series analysis. We developed FARIMA for daily user preference prediction. So, it was a challenge for us to ensure that there

is no discontinuity in time series data. In order to tackle this challenge we collected all the dates (time) and the corresponding number of reviews for each data for a particular feature and sorted our data (Both dates and reviews) to make sure that no data is missing. Thus, we prepared a continuous time series (TS) of 2 years for a particular product feature. It is very important and challenging to make the time series stationary over time for future predictions. So, we have differentiated our time series data and plotted it to ensure that all the trends are removed and the time series is stationary over time. Estimating the correct order of parameters such as correct Fourier order or AR/MA orders was also a challenging job. In order to select the best fit model we have used the evaluation technique according to the paper [**Zhang0ZLLZM15**] known as Akaike's Information Criterion (AICc). The solutions we adopted helped us to estimate the best fit model and determine the future trend of the product.

Apart from the above challenges we have also faced some technical challenges. In our thesis we have implemented our prediction models in Java language. In the first two models such as TVBPRPlus and Timesvd++, we have considered Java Arrays to represent certain model parameters in our code. But while running the code to generate predicted results we found there was efficiency and memory problems. So, later we replaced those parameters with Java ArrayList to solve the problem. Regarding the FARIMA model we also faced some technical problems such as it could be much easier for us to implement FARIMA in languages like R instead of Java, since in R language we already have in-built packages for ARIMA process. But in our case we tried to understand the underlying concepts of FARIMA in details. Then we have implemented the different components of the model programatically without the use of in-built packages. In our graphical tool we have visualized the future trend of a specific product. This can be easily achieved for FARIMA model. But for user centric models like TVBPRPlus (ranking based) and Timesvd++ (rating based) this was challenging. So, we have adopted our models in such a way so that we can compare all the models on a common ground and visualize the trend prediction of a specific product. For example in case of Timesvd++ we have considered the number of product reviews to determine its future trend. In case of TVBPRPlus we have considered the number of reviews of a certain product known to the user to predict the future trend of the product or similar products that can be recommended to the user.

Chapter 8

Conclusion

Fashion is a fascinating domain that have gained a lot of attention due to the emergence of new products in the market. The main focus of our thesis was to implement the three prediction models namely TVBPRPlus,Timesvd++ and FARIMA based on Amazon fashion data and design a graphical tool that visualizes the prediction results of these techniques.In our thesis work we have developed time and visually aware Recommender systems such as TVBPRPlus and Timesvd++ based on user feedback and product images (Product Level Modelling). Our models are capable of capturing evolving visual dimensions of products and as well as associated temporal dynamics of data. We have applied these models to determine the future trend of a certain item (Sunscreen product). We have also developed Recommender systems based on Feature Level modelling (Fourier Assisted ARIMA or FARIMA), where we have extracted explicit features of a specific product. FARIMA helps us to achieve daily aware preference prediction through time series analysis. We have designed a graphical tool to visualize the predicted results of these different models.

In order to see the power of the models we have evaluated them using different evaluation techniques. We have compared the efficiency and accuracy of these models by changing the different input parameters. We have learned which model performs better than others in certain settings and what should be the ideal values of the model parameters to achieve best performance from a specific model.

Chapter 9

Future Scope

In our work we have used feature level modelling in FARIMA model. We have extracted explicit features from user reviews for a particular product domain. The feature extraction is done by Keyword search from user reviews. In future we plan to perform feature extraction through an automatic tool. The feature extraction tool will perform data pre-processing, tokenization and data parsing to extract features from textual reviews. In case of TVBPRPlus model we have determined visually similar items among top 50 items and found the future visual popularity of one specific item based on their visual scores in the experimental section. We plan to include this concept as a part of our graphical tool which will show the similar products in the visual space.

[HeM16] [Koren10] [VBPR] [Zhang0ZLLZM15] [Al-HalahSG17] [Fashionista] [MelvilleS17] [BPR] [KorenBV09] [TSForecast]