Πανεπιστήμιο Πειραιώς

Τμήμα Ψηφιακών Συστημάτων

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΨΗΦΙΑΚΑ ΣΥΣΤΗΜΑΤΑ ΚΑΙ ΥΠΗΡΕΣΙΕΣ

Κατεύθυνση: Προηγμένα Πληροφοριακά Συστήματα

Recommender systems using sentiment analysis

Μαρία-Ελένη Κουτρίδη, ΑΜ: ΜΕ1743

Μεταπτυχιακή Διπλωματική Εργασία

Επιβλέπων: Χαλκίδη Μαρία, Επίκουρη Καθηγήτρια

Ιούλιος 2019

Index

Introduction – Περίληψη

Τα τελευταία χρόνια έχει αναπτυχθεί πληθώρα συστημάτων συστάσεων για να καλυφθούν οι επιχειρησιακές ανάγκες. Οι επιχειρήσεις τείνουν να χρησιμοποιούν πολύτιμα δεδομένα όπως π.χ. τις προτιμήσεις των χρηστών και τις ομοιότητες μεταξύ αντικειμένων με σκοπό να συστήσουν στους χρήστες προϊόντα αντίστοιχα με τις προτιμήσεις αυτές. Αυτή η διαδικασία είναι ευρέως γνωστή και χρησιμοποιείται σε διάφορες περιπτώσεις.

Η ανάλυση συναισθήματος (γνωστή και ως εξόρυξη γνώμης) είναι ένα πεδίο της επιστήμης της Ανάλυσης Φυσικής Γλώσσας που οικοδομεί συστήματα τα οποία επιδιώκουν να αναγνωρίσουν και να εξάγουν γνώμες μέσα από κείμενα. Συνήθως, πέρα από την αναγνώριση της γνώμης, τα συστήματα αυτά αναγνωρίζουν ιδιότητες της έκφρασης όπως π.χ. την πόλωση μιας άποψης (αν ο εκφραστής της είναι θετικά ή αρνητικά διακείμενος), το θέμα (θέμα μιας συζήτησης) και την αναγνώριση έκφρασης μιας άποψης (την ύπαρξη ξεκάθαρα υποκειμενικής και όχι αντικειμενικής γνώμης). Σήμερα, η ανάλυση συναισθήματος χρησιμοποιείται στην αυτοματοποίηση ανθρώπινων διαδικασιών όπως αυτές που προαναφέραμε. Ακόμα μια χρήση της ανάλυσης συναισθήματος είναι η αξιοποίηση της αυτοματοποίησης στην προσπάθεια εξαγωγής των απόψεων αυτών από διάφορες ιστοσελίδες, forums και μέσα κοινωνικής δικτύωσης. Με τη βοήθεια της ανάλυσης συναισθήματος, αυτή η αδόμητη πληροφορία μπορεί να μετατραπεί αυτόματα σε χρήσιμα δεδομένα για εμπορική χρήση, κριτικές προϊόντων, ανατροφοδότηση των καταναλωτών σχετικά με αυτά τα προϊόντα και εξυπηρέτηση πελατών.

Για την εκπόνηση αυτής της διπλωματικής εργασίας, δημιουργήθηκε ένα σύστημα συστάσεων ενισχυμένο με ανάλυση συναισθήματος. Αναλυτικότερα, σχεδιάστηκε αρχικά ένα μοντέλο ανάλυσης συναισθήματος με σκοπό να μπορεί να αντιστοιχήσει κριτικές κινηματογραφικών ταινιών με ένα σύστημα βαθμολόγησης με αστέρια, από ένα έως πέντε. Στη συνέχεια, δημιουργήθηκε ένα σύστημα συστάσεων με συνεργατικό φιλτράρισμα με βάση το αντικείμενο. Τέλος, έγινε συνδυασμός αυτών των δύο έτσι ώστε να μελετηθεί αν η μελέτη συναισθήματος μπορούσε να βελτιώσει ή ακόμα και να αντικαταστήσει την κλασσική βαθμολογία του χρήστη στα συστήματα συστάσεων. Στην διπλωματική εργασία αυτή επίσης μελετώνται πολλά συστήματα συστάσεων και καθορίζονται τα πλεονεκτήματα και μειονεκτήματα του καθενός. Επιπλέον, γίνονται μετρήσεις τόσο πάνω στο μοντέλο ανάλυσης συναισθήματος όσο και στο σύστημα συστάσεων, τα αποτελέσματα των οποίων αναλύονται.

Το αντικείμενο της διπλωματικής εργασία αυτής είναι να ερευνήσει τον βαθμό στον οποίο η ανάλυση συναισθήματος μπορεί να βελτιώσει τα συστήματα προτάσεων. Η υλοποίηση συστημάτων συστάσεων ενισχυμένων με ανάλυση συναισθήματος αποσκοπεί στην εξερεύνηση της συμβολής και της σημασίας της εξόρυξης απόψεων στα συστήματα συστάσεων με μετρήσεις και παραδείγματα. Τέλος, η εργασία αυτή εξετάζει κατά πόσον η εξόρυξη απόψεων από τους χρήστες τους καθοδηγεί εν τέλει να λαμβάνουν χρήσιμες πληροφορίες για μελλοντικές συστάσεις.

Summary

In recent years, a variety of recommender systems have been developed in order to meet business needs. Businesses aim in using valuable information such as user preferences and item similarities to recommend clients more and more relevant products. This process is well-known by now and it is used in a range of occasions.

Sentiment analysis (known as opinion mining) is a field within Natural Language Processing that builds systems that try to identify and extract opinions within text. Usually, besides identifying the opinion, these systems extract attributes of the expression e.g.: Polarity (if the speaker expresses a positive/negative opinion), Subject (the subject of a conversation) and Opinion holder (the person or entity that expresses the opinion). Currently, sentiment analysis is used in order to automate some human procedures like the above. Another use is taking advantage of this automation, and try to educe the expressing opinions that exist in many sites, forums and social media. With the help of sentiment analysis, this unstructured information could be automatically transformed into useful information for commercial application, product reviews, product feedback, and customer service.

For the purposes of this thesis, a recommendation system enhanced with sentiment analysis is built. Firstly, a sentiment analysis model was designed in order to be able to assign to a movie review a star rated from one to five. Secondly, an item-item collaborative filtering recommendation system was developed. Then the two of them were combined in order to study if the sentiment analysis could enhance or even replace the rating of the user in recommender systems. This thesis also investigates many types of recommendation systems and states the pros and cons of each one. Also, the sentiment analysis model as well as the recommendation system are measured and the measurements are analyzed.

The objective of this thesis is to investigate the degree of the enhancement that sentiment analysis offers to recommendation systems. This implementation of recommender systems boosted with sentiment analysis tries to explore the meaning and contribution

with measurements and examples of opinion mining to recommendation systems. It also investigates whether extracting opinions from users steers the same users to get valuable information about future recommendations.

Chapter 1. Introduction to recommender systems

Recommender systems or recommendation systems are information systems that aim to predict the user's preferences, given their ratings about items, and suggest to them other items such as what items to buy, or what music to listen to, or what online news to read. The suggestions could be personalized or not, depending the recommendation system implemented. The recommender systems are divided on three categories, collaborative filtering, content-based filtering and hybrid systems that combine the first two.

1.1. Content-based filtering

Content-based filtering systems recommend items that are similar to the ones that the user liked in the past. The similarity is calculated on the characteristics of the items that the user rated higher using tf-idf calculation or other means like word embeddings (analyzed further in chapter 3.2).

$$tfidf(t,d,D) = tf(t,d) * idf(t,D)$$

Equation 1: Term Frequency and Inverse Document Frequency equation

Term Frequency (TF) and Inverse Document Frequency (IDF) (as shown in equation 1) are used to determine the relative importance of an item (that could be a document, an article, a movie etc.) [9].

Term Frequency - tf(t,d) - is the frequency of a word t in a document d. Inverse Document Frequency – idf(t,D) - calculates the weight of the more meaningful terms in a document, giving less weight to the words commonly used. Idf is calculated by the logarithmic scale of the total numbers of documents in the corpus divided by the number of documents where the term appears.

Content-based filtering has a number of advantages: Results are more relevant as users in general tend to choose items from the category of their preference. Users can start using the system more quickly after just a few ratings. New items can be recommended immediately as well, given that they already have tags before they are inserted in the system.

Disadvantages of this method are: lack of diversity, because it's important for a recommendation system to suggest items with diversity, and items that the user wasn't expecting. Other disadvantages are scalability and that attributes may be incorrectly applied. This is an aftereffect of the fact that experts should be hired to ensure the correct tagging of the items.

In content-based recommendation systems, in order to recommend items to a user, the system tries to group the common categories of the items that the user has already rated. With this procedure, the taste of the user is defined by categories or groups of categories. This technique is limited by the features that are explicitly associated with the objects that these systems recommend. So, in order to have a sufficient set of categories, the content should be in a form that can be parsed automatically by a computer or they should be assigned manually. The latter is almost impossible due to the limitation there is of resources. On the other hand, the forms that can be assigned (e.g. text) can work well with this system. [19]

1.2. Collaborative filtering

Collaborative filtering uses the preferences of other users with similar taste of the target user, in order to suggest items that the target user hasn't rated. Collaborative filtering is divided on two categories as well, memory-based collaborative filtering and model-based [14].

Memory-based recommendation systems

Memory-based recommendation systems are the systems that load all the data to the memory and make predictions based on such in-line memory database. It is quite simple, but there is a problem with huge data [28]. Memory based recommendation systems can be implemented with two ways: user-based and item-based.

User-based collaborative filtering (memory-based) is recommending items by finding similar users to the current user (K-nearest neighbors) (Figure 1). The neighborhood-based algorithm calculates the similarity between two users or items (user-user or item-item collaborative filtering) and produces a prediction for the user by taking the weighted average of all the ratings. Similarity can be calculated with Pearson correlation or cosine similarity.

Item-based collaborative filtering, on the other hand, is recommending items by finding the most similar items based on all the ratings of the users between those items. This technique was developed to offset the problems created by the user-based collaborative filtering. Collaborative filtering has problems like data sparsity, known as the cold start problem: a new user would have to review a number of items in order for the system to give a proper recommendation to this user. Another problem is scalability that is caused by large numbers of users and items. Those are some of the problems of collaborative filtering among others.

Collaborative filtering systems try to predict the utility of items for a user based on the items previously rated by other users that are near the user (KNN algorithm) or by other items that the user has liked in the past. The limitations of this technique are the new user problem, the new item problem and sparsity. The new user problem is a problem when a new user enters the system and the system doesn't know his reaction to any item, so it's not reliable to make any recommendation. The new item problem is when a new item enters the system, so until the new item is rated by a substantial number of users, the recommender system would not be able to recommend it. Sparsity is a problem when the number of ratings for an item is very small compared to the number of ratings that need to be predicted [19].

Advantages are the ease of the implementation and that compared to content-based, is more accurate. This technique has a problem with sparsity, because the percentage of people rating items or rating enough items is really low, and scalability because more users in the system may considered better for finding better results, but there is a high cost. Another problem is the known cold-start problem: new users will find it hard to have accurate recommendations. Last but not least of the problems is the new item problem, which will lack ratings to actually make a solid recommendation about this item.

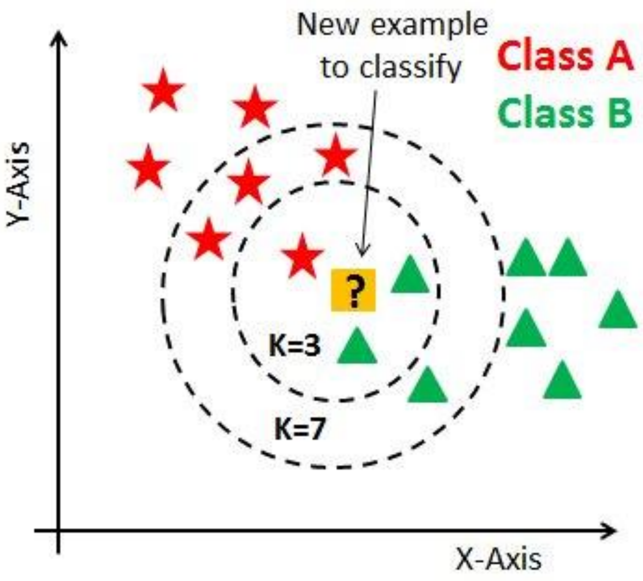


Figure 1: Knn algorithm example

Pearson Correlation

The Pearson (Product-Moment) correlation r is a metric in order to decide how similar is x and y items.[10] Pearson correlation is a measure between two variables x and y (as shown in equation 2).

$$simil(x,y) = \frac{\sum_{i\in I_{xy}}(r_{x,i} - \bar{r}_x)(r_{y,i} - \bar{r}_y)}{\sqrt{\sum_{i\in I_{xy}}(r_{x,i} - \bar{r}_x)^2}\sqrt{\sum_{i\in I_{xy}}(r_{y,i} - \bar{r}_y)^2}}$$

Equation 2: Pearson correlation equation

The value of this correlation is between -1 and 1 where 1 describes an absolute positive linear correlation, 0 is no linear correlation and -1 is an absolute negative linear correlation. Therefore, calculating the correlation between two ratings of two movies, we can assume the relationship of those two movies, and so we recommend the first 10 movies with the highest correlation score.

Cosine Similarity

Cosine similarity is a measure of similarity between vectors x and y, that measures the cosine of the angle between them [11]. The formula of the similarity calculated between x and y is shown in equation 3. Like Pearson correlation this similarity ranges between -1 and 1 and the semantics of this range is exactly the same. For the purposes of this paper, cosine similarity was used.

$$simil(x,y) = \cos(\vec{x},\vec{y}) = \frac{\vec{x}\cdot\vec{y}}{\|\vec{x}\| \times \|\vec{y}\|} = \frac{\sum_{i\in I_{xy}} r_{x,i}\bar{r}_{y,i}}{\sqrt{\sum_{i\in I_x} r_{x,i}^2}\sqrt{\sum_{i\in I_y} r_{y,i}^2}}$$

Equation 3: cosine similarity equation

Model-based recommendation systems

Model-based recommendation systems tries to fit the data into a model, using data mining and machine learning algorithms, and recommends by applying reference mechanism into this model. They respond to the user's request instantly [28]. There are many model-based collaborative filtering algorithms, like Bayesian Networks, clustering models etc not analyzed further here. An honorable mention is a collaborative based filtering method that belongs to this category, which is matrix factorization.

Matrix factorization is the collaborative based filtering method used in recommender systems. Matrix factorization algorithms work to represent users and items in a lower dimensional space (two lower dimensionality rectangular matrices). It is a group of filtering algorithms became widely known when the Netflix prize challenge ended in 2009. This method is where the matrix m*n is decomposed into m*d and d*n (as shown in figure 2). It is used for calculation of complex matrix operation.

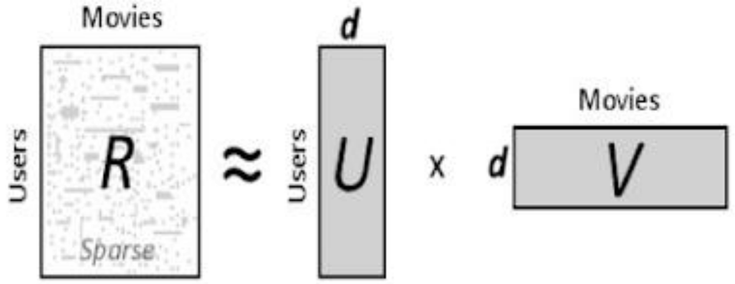


Figure 2: Matrix factorization

Matrix decomposition can be classified into three types

- LU decomposition: Decomposition of matrix into L and U matrix where L is lower triangular matrix and U is upper triangular matrix, generally used for finding the coefficient of linear regression. This decomposition failed if matrix can't have decomposed easily
- QR matrix decomposition: Decomposition of matrix into Q and R where Q is square matrix and R is upper triangular matrix (not necessary square). Used for eigen system analysis
- Cholesky Decomposition: This is the mostly used decomposition in machine learning. Used for calculating linear least square for linear regression

1.3. Hybrid recommendation systems

Hybrid methods combine collaborative and content-based which helps to avoid certain limitations of content-based and collaborative filtering. The ways of this combination are as follows:

1.implementing collaborative and content-based methods separately and combining their predictions.
This implementation assumes that two different systems are already implemented. The two answers of the two systems are combined into one final recommendation using either a linear combination of ratings or a voting scheme.

2. incorporating some content-based characteristics into a collaborative approach, in this implementation, the content-based profiles are kept for each user and enhance the collaborative-filtering technique.

3. incorporating some collaborative characteristics into a content-based approach, and the most popular approach when implementing such system, is to use some dimensionality reduction technique on content-based profiles.
4. constructing a general unifying model that incorporates both content-based and collaborative characteristics.

In [20] proposes using content-based and collaborative filtering features in a single rule-based classifier. Others [21][22] propose a unified probabilistic method for mixing collaborative and content-based recommendations.

In [18] they are trying to combine collaborative filtering systems with content-based filtering systems in an attempt to eliminate the weaknesses found in each approach. In content-based recommendation one tries to recommend items similar to those a given user has liked in the past, whereas in collaborative recommendation one identifies users whose tastes are similar to those of the given user and recommends items they have liked. In the system introduced, there are found two more advantages: First, two scaling problems common to all Web services are addressed - an increasing number of users and an increasing number of documents. Second, the system automatically identifies emergent communities of interest in the user population, enabling enhanced group awareness and communications. So, rather than recommend items because they are similar to items a user has liked in the past, they recommend items other similar users have liked, and rather than computing the item similarity, they compute the user similarity.

Chapter 2. Sentiment analysis approach

Sentiment analysis or else opinion mining, is a computational and mathematical study of people's opinions, trends, emotions and attitudes towards an item. This item can be an article, an item to purchase, music, movies or any other type of preference and they involve a review. In this chapter we analyze the sentiment analysis algorithms and how we use them in this implementation and approach. There are many algorithms used to get sentiment analysis, some of which are linear regression, SVM (Support Vector Machines) and Naïve Bayes classifiers. Below in figure 3 there is a complete diagram of all the sentiment analysis algorithms and how are they categorized.

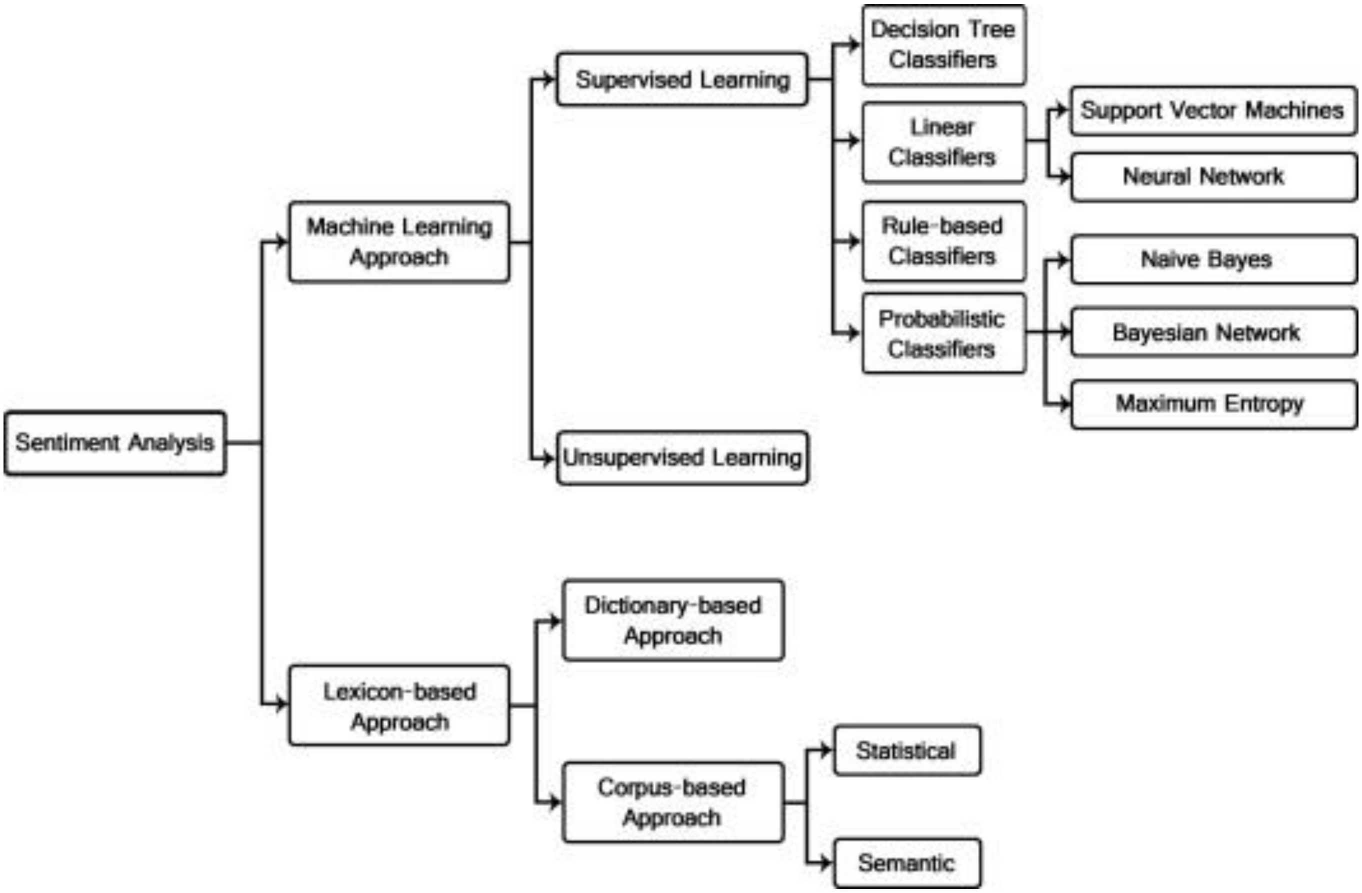


Figure 3: Sentiment analysis algorithm categorization

Firstly, there is a classification between machine learning approaches and lexicon-based. The first need training of algorithms and deliver pretty good results, while the second require a lexicon. Lexicon-based approaches involve dictionaries of words annotated with semantic polarity and sentiment strength. They deliver good results with high precision but low recall but require a large dictionary which is not always possible for all languages [24].

Machine learning approach is categorized in supervised and unsupervised algorithms. The difference between the two is that supervised algorithms have at their disposal the output values of the samples, while on the other hand, unsupervised algorithms don't. Unsupervised algorithms try to infer to the natural structure present within a set of data points. For the purposes of this thesis, we are going to try to cover only the basics [25].

There are many implementations and experiments throughout the research field. In [12], a system is presented for scalable and real-time sentiment analysis for Twitter data which is designed to extract raw text from tweets and process it through a supervised learning method in real time. The importance of this system in terms of classification accuracy, scalability and performance is researched. This research paper concluded that part-of-speech tags, emoticons and prior polarity are of the most significance int the sentiment analysis of Twitter data, and also that in the online process the feedback may play some part to the classification accuracy.

In [13], two deep learning systems are presented that competed at SemEval-201 7 Task 4 "Sentiment Analysis in Twitter". Long Short-Term Memory (LSTM) networks enhanced with two kinds of attention mechanisms are introduced. Text postprocessing step is added performing spell correction, word normalization and segmentation. These models achieved excellent results in the classification tasks, but mixed results in the quantification tasks of the competition. This paper concludes that they would like to explore more quantification techniques in the future and that they would be interested in designing models operating on the character-level.

In [15] combines recommendation system and sentiment analysis in order to generate the most accurate recommendations for users, working with the Algerian language. The experimental results suggested very high precision and recall. The results analysis evaluation provides interesting findings on the impact of integrating sentiment analysis into a recommendation technique based on collaborative filtering. The findings are so encouraging that a future work in this direction is promised. In [14] in chapter 5.1 (Considering feature opinions) there are sub-categories of studies that aim to determine a product's quality using the feature opinions extracted from reviews. In [16] they develop a product transforming the review text into a two-component form: product quality, which refers to the reviewer's evaluation of product features; and opinion quality, which indicated the reviewer's expertise with the reviewed product. They classify the reviews into three categories: good, bad and quality. Then they label each review with the features of the product reviewed. Finally, the overall assessment of the product is obtained by summing up all of its features' overall quality scores. In [17] they build a product profile with the help of feature opinions extracted from product reviews combined with a product's technical specifications. This model indicated the value of a product for the average user, which during the recommendation process it is unique for every user.

2.1. Linear regression

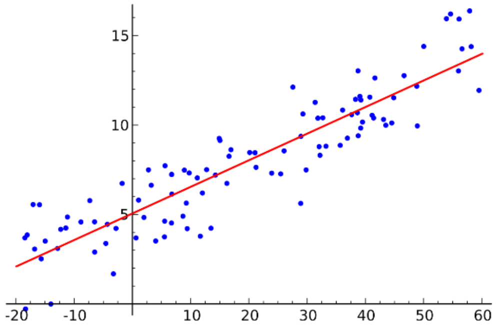


Figure 4: Linear regression example

Linear regression is the most commonly used type of predictive analysis and we also use it in this implementation. It can be defined as Y' = A + B *X (Figure 4 is an example). In this example Y is the predicted value (criterion variable), A is the intercept (estimated by regression), B is the coefficient (estimated by regression) and X is the predictor (present in the data) [8]. Thus, defining the function a line/hyperplane described by the given data, explains the relationship between the variables and can predict with accuracy the independent variant.

2.2. Support Vector Machines (SVM)

When we need to classify some points in multi-dimensional space, then SVM algorithm could be appropriate. The objective of SVM algorithm is to find a hyperplane that distinctly classifies the data points in multi-dimensional space. The dimension of the hyperplane depends upon the number of features.

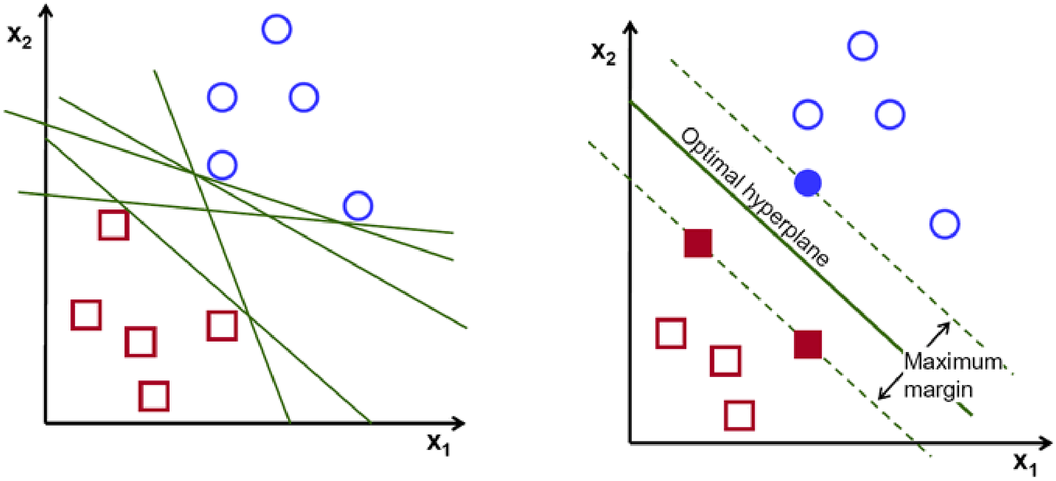


Figure 5: Finding the optimal hyperplane through all the possible hyperplanes

There can be found many hyperplanes correctly classifying the two groups. After many repetitions of the algorithm, the algorithm stops when it finds the one with the maximum margin from the nearest points of each class [26]. As shown in figure 5, there are many possible hyperplanes but only the one with the maximum margin (that is the maximum distance between the data points of the two classes) is chosen.

2.3. Naive Bayes Classifier

Naïve Bayes is a subset of Bayesian decision theory. It's called naive because the formulation makes some naïve assumptions. All naive Bayes classifiers assume that the value of a specific feature is independent of the value of any other feature, given the class variable.

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

Equation 4: Bayes theorem

Naïve Bayes delivers better results when it comes to classifying texts. P(A|B) is "Probability of A given B" (the probability of A given that B happens), P(A) is Probability of A, P(B|A) is "Probability of B given A" (the probability of B given that A happens) and finally P(B) is Probability of B happening [27] (equation 4).

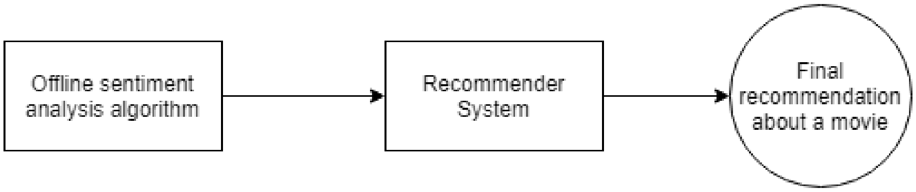
For two events, A and B, Bayes' theorem allows you to figure out p(A|B) (the probability that event A happened, given that test B was positive) from p(B|A) (the probability that test B happened, given that event A happened). Types of Naive Bayes Classifier:

- Multinomial Naive Bayes
- Bernoulli Naive Bayes
- Gaussian Naive Bayes

All three algorithms (SVM, Naïve Bayes Classifier and Linear regression) were tried for this implementation, but with the dataset provided, the linear regression had better results.

Chapter 3. Recommendation system using sentiment analysis: Our approach

The system is quite simple (figure 6) and is designed with and offline sentiment analysis system that feeds the recommender system as follows:



Offline sentiment analysis algorithm
Recommender System
Final recommendation about a movie

Figure 6: System overview

The sentiment analysis system runs the algorithm offline, which means the results are already in the dataset when the recommender system runs. The input is a review of the system processed through the algorithm analyzed further in chapter 3.2. The recommender system then chooses between the average rating between the user rating and sentiment analysis rating, the actual rating of the user or the sentiment analysis rating alone. This system, outputs the possible rating which the user would have given the movie, and a set of movies throughout the system, that the user would like.

3.1. Recommendation system

The recommendation system (figure 7) has as input the actual rating of the user or the result from sentiment analysis or the combination (or average) between the user and the result from sentiment.

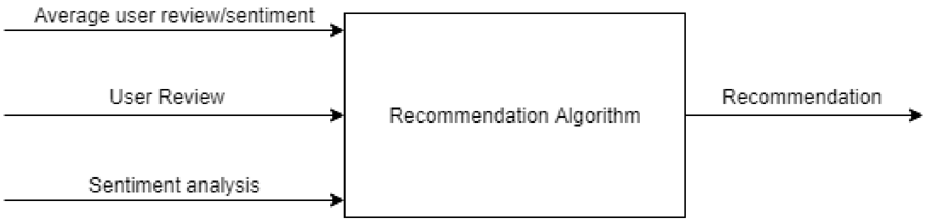
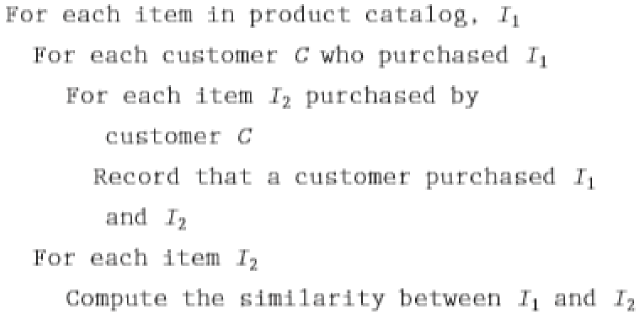


Figure 7: Recommendation system

Firstly, in order to test the system, the dataset was separated into training and test set by 70% and 30% respectively. Before calculating the similarity matrix, a normalization for ratings was implemented due to different average rating of each user. For example, the average rating of one user lies at a scale of 2; another one's lies at a scale of 4. Which means that the first user's rating is similar as the second user's rating of 4. The normalization method is to calculate the average rating of each user and subtract the average value from the actual ratings as adjusted ratings. The ratings that are fed to the computation, are the first time of the measurements the actual ratings of the users, the second time the average between the sentiment analysis prediction and the actual rating of the user, and the third time solely the sentiment analysis prediction.

The item-item similarity matrix (matrix factorization), was constructed with the help of cosine similarity which helps calculate the similarity value between two items (see cosine similarity formula in equation 3). Two distinct movies are selected, then the ratings from the users that rated one of the two movies, are eliminated, then for each of the two movies, all the ratings of the specific movies from all the remaining users are put into a vector, each user representing a dimension of the vector, then the similarity is calculated and stored in the matrix. Finally, in order to fill out the whole matrix, the Amazon algorithm (figure 8) [23] is implemented to iterate each pair of two distinct movies. To find the most similar match for an item, the Amazon algorithm builds an item-item similarity matrix by finding items that customers tend to purchase together. Instead of building an item-item matrix by iterating through all item pairs, this algorithm calculates the similarity between an item and all the related items if the products have common customers.



```
For each item in product catalog, $I_1$
  For each customer $C$ who purchased $I_1$
    For each item $I_2$ purchased by
       customer $C$
      Record that a customer purchased $I_1$
       and $I_2$
  For each item $I_2$
    Compute the similarity between $I_1$ and $I_2$
```

Figure 8: The Amazon algorithm

In order to predict ratings for unrated movies for a given user (or specifically for one movie), the prediction function (equation 5) need to be iteratively applied on all of the unrated movies for the given user.

$$r(i;u) = \mu_i + \frac{\sum_{j\in I_u}(r_{uj} - \mu_j)w_{ij}}{\sum_{j\in I_u}|w_{ij}|}$$

Equation 5: Prediction equation

r(i;u) means the predicted rating of movie I for user u, μ_i and μ_j represent the average ratings for movie i and movie j across all users, respectively, I_u is the set of movies rated by user u, and w_{ij} indicates the similarity value between movie I and movie j.

After the last step, the similarity matrix is completed with all the ratings of unrated movies by the given user. Lastly, we get the top-ranked movies which constitute the recommended movies that are likely to attract the user.

3.2. Sentiment analysis

The sentiment analysis system is an offline system (figure 9). It is trained with linear regression (chapter 2.1). For the training of the sentiment analysis model scikit-learn library is used, which is an implementation of linear regression and other supervised/unsupervised algorithms.

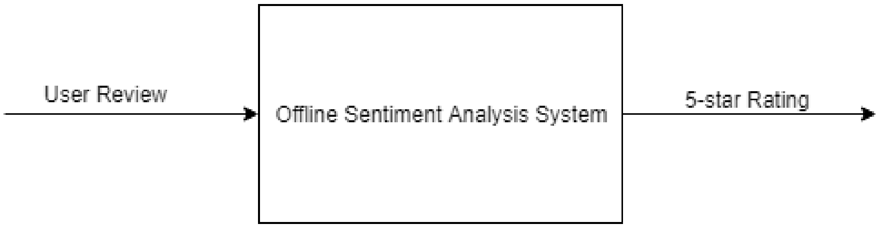


Figure 9: Offline sentiment analysis system

Scikit-learn is an open source machine learning library for Python used to implement various functions in this thesis. It provides many unsupervised and supervised learning algorithms. The functionality that this library provides include regression (Linear and

Logistic), Classification (KNN), Clustering (K-means and K-means++), model selection and preprocessing (Min-Max normalization).

For this approach train/test data were separated in 70% train data and 30% test data which is the optimum as tested. For the training of the model the data to 5 classes are separated according to the five-star rating according to the reviews given by users, and used the doc2vec model to transform the reviews into multi-dimensional space depictions.

Word embeddings

In this approach of sentiment analysis, word embeddings and specifically the implementation of doc2vec model is used.

Word embeddings are implementations of neural networks in the field of natural language processing (NLP) where words or phrases from the vocabulary are represented as a vector(s) into a low dimensional continuous space [7].

word2vec [3] (used for this implementation) is a two-layer neural net, the input is a word and the output are a multidimensional vector that represents the inputted word into space. word2vec representation is created using either Continuous Bag-of-Words model (CBOW) or the Skip-Gram model. The CBOW model is used on a context to predict one word, in contrary the skip gram model does the opposite: it uses one word to predict the context. doc2vec [4] is the equivalent but for documents, regardless their length.

For the implementation of the word embeddings gensim library for Python to train the model is used. During the reading of all the data there was a mild preprocessing. Finally, the model was trained utilizing Continuous Bag of Words (CBOW) architecture model and saved for further usage.

Chapter 4. Experimental evaluation

4.1. Dataset

For this analysis a dataset found on the Internet is used. It is provided from assistant professor in UC San Diego Julian McAuley crawled from Amazon and distributed online. This dataset includes reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, and image features), and links (also viewed/also bought graphs) about movies and TV series. [1][2] A subset of this massive dataset was used in order to be more maintainable.

Below are presented the evaluation metrics that are used to determine the precision of the system implemented.

The coefficient of determination

R^2 metric, otherwise as known the coefficient of determination, measures the proportion of total variation about the mean Y explained by the regression. The coefficient of determination, R^2, is similar to the correlation coefficient R. The correlation coefficient formula (equation 6) will tell how strong a linear relationship is between two variables. R^2 is the square of the correlation coefficient. Coefficient of determination (R^2) can take values as high as 1 (100%) or when all the values are different i.e. $0 \leq R2 \leq 1$ [5].

$$r = \frac{n(\Sigma xy) - (\Sigma x)(\Sigma y)}{\sqrt{[n\Sigma x^2 - (\Sigma x)^2][n\Sigma y^2 - (\Sigma y)^2]}}$$

Equation 6: correlation coefficient equation

r^2 metric was found **0.55** which is a very good score considering that we are trying to predict human interactions.

The Mean Absolute Error (MAE)

Mean absolute error is the average sum of the difference between the predicted value and the actual value where all differences have equal weight (equation 7).

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

Equation 7: Mean absolute error equation

This metric was found **0.78** which means that the built system predicts 0.78 more or less than the actual value on average.

Root mean squared error

Root mean squared error is a quadratic score that also measures the average difference between predicted and actual value (equation 8). The difference between root mean squared error and mean absolute error is the square root of the average of calculated differences which actually gives a relatively high weight to large errors.

$$RMSE = \sqrt{\frac{\sum_{I=1}^{N}(Predicted_i - Actual_i)^2}{N}}$$

Equation 8: root mean squared error equation

This metric was found **0.90** which is close to mean square error. The root mean squared error being close to mean square error means that in this model not many large errors were made.

For the testing of the recommendation system results we used precision and recall metrics.

Precision

Precision (otherwise positive predictive value) is the percentage of the results that are relevant (true positive).

Recall

Recall (otherwise sensitivity) is the percentage of total relevant results correctly classified.

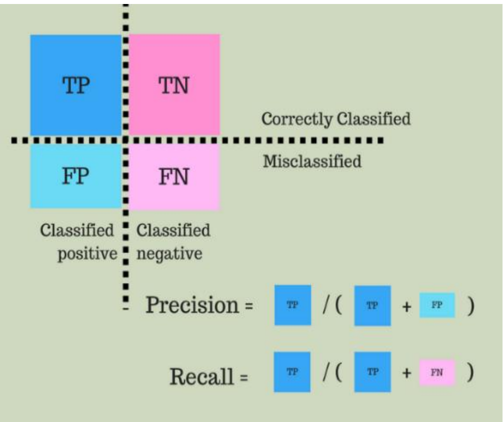


Figure 10: Precision recall explained

The precision and recall were measured in smaller datasets of 5000, 20000, 60000 and 100000 reviews and only for users with ratings. In order to calculate the precision and recall, the similarity matrix was calculated. For every two movies the similarity was calculated and the missing values were filled out by the prediction model analyzed in chapter 3.1.

The experiments were taken for a movie with many reviews **0767803434 (Air Force One)**, and a user **ANCOMAI0I7LVG (Andrew Ellington)** that is a user with many submitted reviews in his history.

4.2. Results

The following chart is the experiment results (**precision and recall**) matrix for every dataset:

Experiment results	Actual ratings		Average ratings/sentiment		Sentiment	
	Precision	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
5000 reviews	0.85	0.99	0.85	0.99	0.90	0.99
20000 reviews	0.88	0.99	0.88	0.99	0.88	0.99
60000 reviews	0.89	0.99	0.89	0.99	0.89	0.99
100000 reviews	0.90	0.99	0.91	0.99	0.89	0.99

Figure 11:Precision and recall measurements for every dataset

In figure 11 we observe that the precision with more reviews is better but the recall is the same. Precision is the amount of "noise" that the algorithm generates, so that means that the not related data are the same, as the reviews increase. Recall is how precise are the results, and this is a logical conclusion as the data increase. There is no difference when the actual ratings or the combination between ratings and sentiment (column average ratings/sentiment) or sentiment solely are used.

In figure 12 is presented the execution **time** of the program for every dataset (in **seconds**):

Experiment time	Actual ratings	Average ratings/sentiment	Sentiment
5000 reviews	12.09s	13.13s	12.63s
20000 reviews	46.15s	39.9s	37s
60000 reviews	242.57s (~4 minutes)	250.09s (~4 minutes)	249. 71s (~4 minutes)
100000 reviews	647 s (~10 minutes)	637 s (~10 minutes)	641s (~10 minutes)

Figure 12: execution time for every dataset

In the above figure (12) we observe that the execution time of the program increases as the reviews increase. It is very reasonable, because in the algorithm implemented there are many iterations of the items and that increases the time.

The figure 13 is the **mean absolute error** the program run for every dataset:

Mean absolute error	Actual ratings	Average ratings/sentiment	Sentiment
5000 reviews	0.84	0.86	0.85
20000 reviews	0.83	0.84	0.84
60000 reviews	0.82	0.82	0.82
100000 reviews	0.80	0.81	0.80

Figure 13: mean absolute error for every dataset

In figure 13 we see that the mean absolute error is better as the results increase. That is a logical assumption as in general that happens when the algorithm has more known points.

The figure 14 is the **root mean squared error** the program run for every dataset:

Root mean squared error	Actual ratings	Average ratings/sentiment	Sentiment
5000 reviews	1.45	1.02	1.47
20000 reviews	1.32	1.03	1.30
60000 reviews	1.24	0.96	1.25
100000 reviews	1.22	0.90	1.22

Figure 14: mean squared error for every dataset

In figure 14 we observe that the root mean squared error is better for average rating/sentiment versus the actual rating used. Although the mean absolute error in figure 14 is the same, +/-0.01 or +/-0.02 is not measurable in mean absolute error, root mean squared error in the contrary shows that the average is better. Sentiment score used alone is the same as the actual rating.

4.3. Discussion

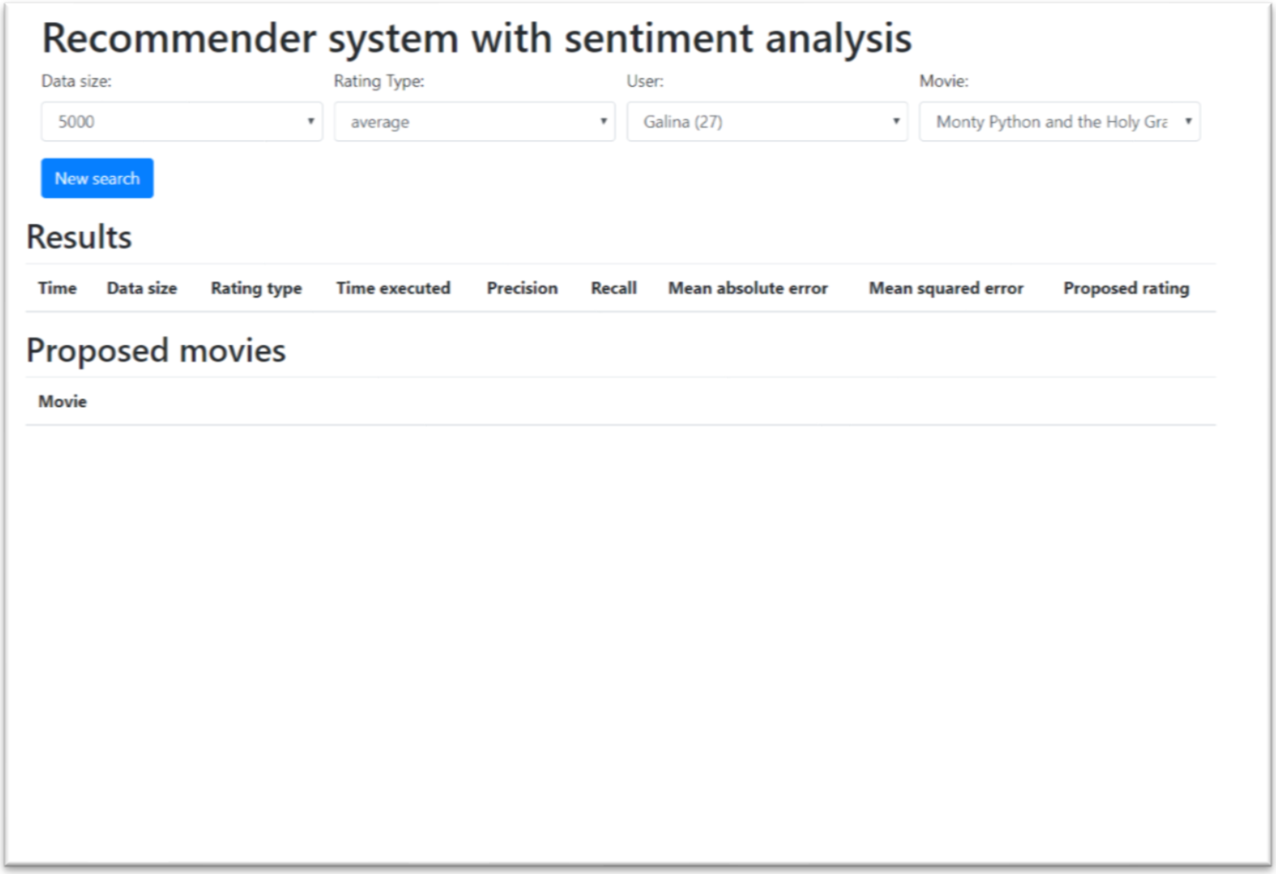
The precision and recall in all three experiments are exactly the same (figure 11). This means that the model is very accurate, with the outcome being above 90%, and that the sentiment model can replace entirely the ratings of the users. The sentiment results combined with actual ratings do not enhance the system as the results are the same in all three categories.

The higher recall compared to precision is not peculiar, actually precision is the amount of "noise" that the algorithm generates while recall is the exactness of the result. So, this essentially means that the algorithms generate similar amount of noise to the amount of correctly generated data labels.

The extra time spent for more reviews added is exponential and adds little value to precision and recall. Although, the actual prediction of the movie was better when more reviews were added, as shown from mean absolute error. The mean squared error in the contrary, showed that the average is indeed better than the actual rating, and than the sentiment score alone in figure 14. This is because, as mentioned above, mean squared error is not so tolerant about errors. The predictions about the actual rating and the sentiment score are the same, so this reinforces the hypothesis we made earlier that the actual rating can be replaced totally from the sentiment analysis prediction score.

4.4. The interface of the implementation

Below we can see some use-cases and examples of the application. The basic interface is consisted by an options section and a result section.



Recommender system with sentiment analysis

Data size: 5000

Rating Type: average

User: Galina (27)

Movie: Monty Python and the Holy Gra

Results

Time	Data size	Rating type	Time executed	Precision	Recall	Mean absolute error	Mean squared error	Proposed rating

Proposed movies

Movie

Figure 15: Overview of the interface

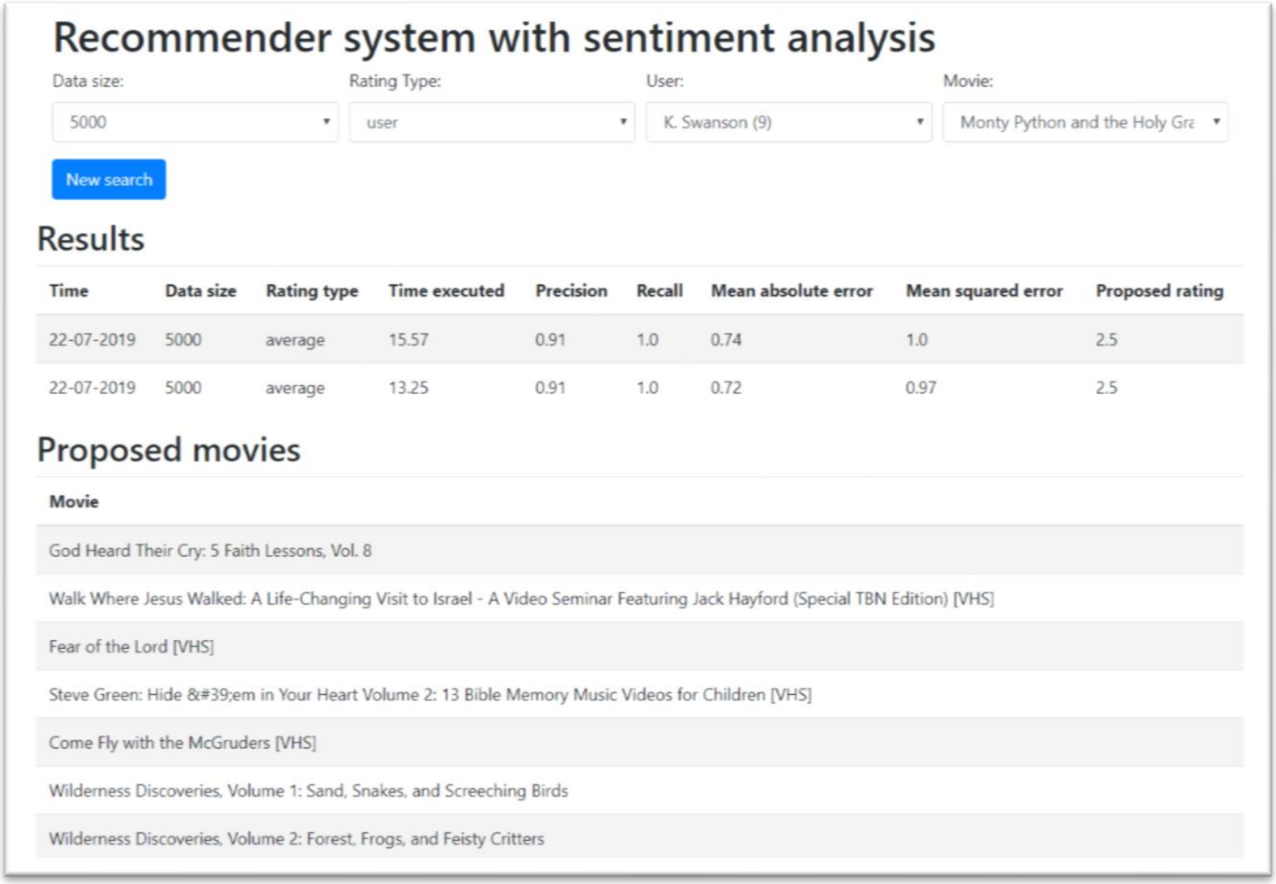
Data size refers to the size of our dataset. The more data we have, the more precise the model is.

Rating type gives us the ability to chose if we want to use the rating of the user, the rating of the sentiment analysis, or a combination of both of them.

Users are the system users.

Movies are some indicative examples of movies that vary in the number of user-ratings they have received.

In the next picture, we can see the result after a new search.



Recommender system with sentiment analysis

Data size: 5000

Rating Type: user

User: K. Swanson (9)

Movie: Monty Python and the Holy Gra

Results

Time	Data size	Rating type	Time executed	Precision	Recall	Mean absolute error	Mean squared error	Proposed rating
22-07-2019	5000	average	15.57	0.91	1.0	0.74	1.0	2.5
22-07-2019	5000	average	13.25	0.91	1.0	0.72	0.97	2.5

Proposed movies

Movie
God Heard Their Cry: 5 Faith Lessons, Vol. 8
Walk Where Jesus Walked: A Life-Changing Visit to Israel - A Video Seminar Featuring Jack Hayford (Special TBN Edition) [VHS]
Fear of the Lord [VHS]
Steve Green: Hide &#39;em in Your Heart Volume 2: 13 Bible Memory Music Videos for Children [VHS]
Come Fly with the McGruders [VHS]
Wilderness Discoveries, Volume 1: Sand, Snakes, and Screeching Birds
Wilderness Discoveries, Volume 2: Forest, Frogs, and Feisty Critters

Figure 16: Interface with results

In the results section we can see data concerning the search.

Data size is the size of the data set chosen for the search.

Rating type is the type of rating chosen for the search.

Time executed is the execution duration of the script.

Precision, Recall, Mean absolute error and **Mean squared error** are system metrics about the accuracy of the process.

Proposed rating is the rating that indicates how much the user would like the chosen movie.

Finally, the system, based on the user rating about this movie, recommends the top ten movies that probably the user would appreciate the most (shown in figure 16).

Chapter 5. Conclusion – Further work

In this thesis we aim to investigate whether a sentiment analysis model could enhance or even replace the recommender algorithm. This could provide better results for the users. First an implementation of the sentiment analysis was built (chapter 2.1), with the help of linear regression. Then, a recommender system was built (chapter 3.1), this implementation was then integrated to the recommender system.

The precision of the built model is very accurate, as shown from the above measurements (chapter 4.2). The sentiment analysis model is so accurate that it can replace the ratings of the users. The sentiment prediction for this data and experiment, cannot enhance the user's ratings further as shown in figure 11, so it is pointless to use it in that form.

It would be meaningful and a great use to the research community, on the other hand to replace the ratings with the predictions of the sentiment analysis model. This finding would be interesting if it was applied to social media, and other applications with opinion mining with a vast use.

Another interesting approach and investigation would be to try understanding through sentiment analysis if an opinion (a review) is objective or subjective. If a review expresses objective criteria then it could be used to identify an item or further analyze it. A subjective opinion could also be at use, to identify the sentiment of the user review.

A complete user interface system to display this work is left for future work. Also, it would be interesting if the reviews could be categorized or enhanced by using the location of the item or even the user to see the results.

Chapter 6. Bibliography

[1] R. He and J. McAuley, “Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering”, 2016

[2] J. McAuley, C. Targett, J. Shi and A. van den Hengel, “Image-based recommendations on styles and substitutes”, 2015

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado and Jeffrey Dean, “Distributed Representations of Words and Phrases and their Compositionality”, 2013

[4] Tomas Mikolov, “Distributed Representations of Sentences and Documents”, 2014

[5] Statistics How To. Coefficient of Determination (R Squared): Definition, Calculation Statistics How To. Available at: https://www.statisticshowto.datasciencecentral.com/probability-and-statistics/coefficient-of-determination-r-squared, 2019.

[6] Basic Statistics and Data Analysis. Coefficient of Determination: A model Selection Criteria. Available at: http://itfeature.com/correlation-and-regression-analysis/coefficient-of-determination, 2019.

[7] Yitan Li, Linli Xu, Fei Tian, Liang Jiang, Xiaowei Zhong and Enhong Chen, “Word Embedding Revisited: A New Representation Learning and Explicit Matrix Factorization Perspective”, 2015

[8] Spss-tutorials.com. Simple Linear Regression - Quick Introduction. Available at: https://www.spss-tutorials.com/simple-linear-regression, 2019.

[9] Rajaraman, A.; Ullman, J.D. (2011). "Data Mining". Mining of Massive Datasets. pp. 1–17

[10] A. G. Asuero, A. Sayago, and A. G. González (2006), “The Correlation Coefficient: An Overview”

[11] Faisal Rahutomo, Teruaki Kitasuka, and Masayoshi Aritsugi, “Semantic Cosine Similarity”

[12] Maria Karanasou, Anneta Ampla, Christos Doulkeridis and Maria Halkidi, “Scalable and Real-time Sentiment Analysis of Twitter Data”

[13] Christos Baziotis, Nikos Pelekis, Christos Doulkeridis, "DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis"

[14] Li Chen, Guanliang Chen, Feng Wang, "Recommender systems based on user reviews: the state of the art"

[15] Amel Ziani, Nabiha Azizi, Didier Schwab, Monther Aldwairi, Nassira Chekkai, et al. Recommender System Through Sentiment Analysis. 2nd International Conference on Automatic Control, Telecommunications and Signals, Dec 2017, Annaba, Algeria.

[16] Aciar, S., Zhang, D., Simoff, S., Debenhanm, J.: Informed recommender: basing recommendations on consumer product reviews. IEEE Intell. Syste. 22(3), 39-47 (2007)

[17] Yates, A., Joseph, J., Popescu, A.M., Cohn, A.D., Sillick, N.: Shopsmart: Product recommendations through technical specifications and user reviews. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, Napa Valley, California, USA, ACM, CIKM'08, pp 1501-1502 (2008)

[18] Balabanovic, M., Shoham, Y.: Fab: content-based, collaborative recommendation. Commun. ACM 40(3), 66-72 (1997)

[19] Gediminas Adomavicius, and Alexander Tuzhilin "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions"

[20] C. Basu, H. Hirsh, and W. Cohen, "Recommendation as Classification: Using Social and Content-Based Information in Recommendation," Recommender Systems. Papers from 1998 Workshop, Technical Report WS-98-08, AAAI Press 1998.

[21] A. Popescul, L.H. Ungar, D.M. Pennock, and S. Lawrence, "Probabilistic Models for Unified Collaborative and ContentBased Recommendation in Sparse-Data Environments," Proc. 17th Conf. Uncertainty in Artificial Intelligence, 2001.

[22] A.I. Schein, A. Popescul, L.H. Ungar, and D.M. Pennock, "Methods and Metrics for Cold-Start Recommendations," Proc. 25th Ann. Int'l ACM SIGIR Conf., 2002

[23] Greg Linden, Brent Smith, and Jeremy York "Amazon.com Recommendations Item-to-Item Collaborative Filtering" February 2003

[24] Kolchyna, Olga & Souza, Thársis & Treleaven, Philip & Aste, Tomaso. (2015). "Twitter Sentiment Analysis: Lexicon Method, Machine Learning Method and Their Combination."

[25] R. Sathya, Annamma Abraham, (2013), "Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification"

[26] Vladimir N. Vapnik (1998), "Statistical Learning Theory"

[27] Kaviani, Pouria & Dhotre, Sunita. (2017). Short Survey on Naive Bayes Algorithm. International Journal of Advance Research in Computer Science and Management. 04.

[28] Thi Do, Minh-Phung & Van Nguyen, Dung & of Loc Nguyen, Academic Network. (2010). Model-based approach for Collaborative Filtering.