GOLDSMITHS, UNIVERSITY OF LONDON

Computing Department

BSc Computer Science Degree

Films and series Recommender System using Naive Bayes classifier and Logistic regression

by Valentin Abrutin

Abstract

Recommender Systems are currently one of the most popular field in machine learning. They are used in massive amount of situations mostly related to e-commerce websites and services. This paper will concentrate on film and series Recommender Systems algorithms. It will cover strengths and weaknesses of basic approaches such as Naive Bayes classifier and Logistic regression, and compare them to each other with discussion and evaluation of the future of the field. Second part of the report will show implementation of these algorithms on a real world data.

Chapter 1

Introduction

1.1 Overview

Recommender Systems (RSs) are a subfield of information filtering systems that provide suggestions for items that are most likely of interest to a particular user.[1, p. 1] These suggestions can be involved in many aspects of human life, mostly related to e-commerce and entertainment industry. For example, what product to purchase, what music to listen, what news are you interested in or in this paper particular case – what film or series to watch.

Since RSs usually are focused on one specific subject for recommendations, they create unique set of parameters, methods and design features that, when combined with context and knowledge of that particular subject, provide the most accurate and effective suggestions. [1, p. 1]

The research of RSs is relatively new compared to other classical information system tools and techniques such as for example search engines. RSs emerged as an independent research area in the mid-1990s. [1, p. 3] However, as mentioned in abstract paragraph, nowadays interest in this field has dramatically increased as it plays enormous role on websites and web application of such tech giants as Facebook, Google, Amazon and Netflix. In addition, there are special conference, workshops and books dedicated specifically to this field of study. One of the famous example will be Association of Computing Machinery's (ACM) Conference Series on Recommender Systems (RecSys), but there are many other that are closely connected to RSs, such as Intelligent User Interfaces (IUI), Special Interest Group on Information Retrieval (SIGIR), Knowledge Discovery and Data Mining (KDD) and so on.

1.2 Motivation

I am genuinely interested in machine learning field and all corresponding to it areas and sub-fields. We face RSs and algorithms every day in almost any application or website. It is hard to think of the world without any RSs, they help and advise us on a daily basis. Therefore, I have passion to dig deep into that area of ML to investigate key aspects and find out what challenges and perspectives does it have. And of course I would love to try build my own movie and series recommender system because nowadays streaming services are very popular what leads to a crucial role of RSs in them. Poor recommendations lead to bad user experience and eventually to loss of clients. I want to find out how Naive Bayes classifier and Logistic regression algorithms will perform in this task.

1.3 Aims and objectives

To sum up previous paragraphs aim of this report is to investigate RSs research field. Then after that study, evaluate and create Naive Bayes classifier and Logistic regression algorithms for films and series recommender system. Therefore, there are the following objectives:

- Research and evaluation into the history of recommender systems
- Identification of key approaches to recommender systems
- Research on Naive Bayes classifier and Logistic regression algorithms
- Finding films and series data
- Implementation of RSs based on Naive Bayes classifier and Logistic regression algorithms using that data
- Evaluation and discussion of practical work in this project and further steps

1.4 Sections Overview

Chapter 2 – background research on the history and general approaches of RSs. The study of algorithms that will be used in the future for implementation.

Chapter 3 – specifications and resources overview. What data and techniques will be used, and what needs to be found and studied.

Chapter 4 – design, dataset and implementation description. Detailed explanation of practical work made.

Chapter 5 – testing and evaluating both algorithms, discussion based on results. Thoughts on future work.

Chapter 6 – Conclusion and evaluation of the entire project.

Chapter 2

Background Research

2.1 History of Recommender Systems

Recommender Systems started their journey in 1990s when they established many of the basic principles that are currently in use. In 1992, Goldberg et al. proposed Tapestry system which can be named as the first RS based on collaborative filtering approach. Later, in 1994, a news recommendation system called GroupLens was created. The main idea of the project was to automate the rule-based collaborative filtering process of the Tapestry system.[2]–[4]

In 1996 was found, Net Perceptions, the first company that focused on offering the marketing recommender engine. One of their customer was Amazon. Since then, the academic studies and e-commerce experience became a locomotive of RSs.[2]

In 1997, the GroupLens research lab launched the MovieLens project where they have trained the first RS on MovieLens dataset. GroupLens released several MovieLens datasets through 2019 which became very popular for RSs studies. Interestingly, the film industry, as well as its datasets, has become crucial in the research work of MS. MovieLens is not the only one example. The early RSs was mostly build on a nearest-neighbour approach. Field was dominated be collaborative filtering technologies and studies around that. However, Netflix Prize (2006 to 2009) created huge boost in study of machine learning algorithms for rating prediction. It made huge impact on the matrix factorization models research. And again it was a movie dataset with a task to create RS for most accurate movie rating prediction.[2], [4]

Most recent studies are related to deep neural network based RSs which has been rapidly researched since 2016. At present, we can say with confidence that the main limitations of the original matrix-completion problem abstraction have been identified and investigated. RS research is a very rapidly developing field, making it difficult to predict its future. However, it can be said with certainty that classical RSs have a space to develop further. On the other hand new methods, such as the neural science, causal inference and knowledge-enhanced can be a way of resolving current problems.[2], [4]

2.2 The use and importance of Recommender Systems

As mentioned previously Recommender Systems are used in a wide variety of applications, mostly related to e-commerce and media content like movies, books, video games and music. However, it is not the end for use cases, RSs also can be implemented for location-based application, for example they are used by websites for vacation travels like Airbnb or Booking.com. Another popular example will be projects for health and fitness where RSs can recommend different variations of diets or workout plans based on user data.

Such a wide range of use cases actually has solid foundation of importance of RSs. We can highlight several major benefits[1]:

Increase the number of items sold – obviously it is a primary object for RSs. It is so even for non-commercial ones. RS complete this object as they are most likely to meet user needs with their recommendations. Accurate evaluation of this parameter is challenging task and may vary depending on which items RS is build for, however work in that directions shows the positive impact of RSs on sales.[5], [6]

Sales diversity – that benefit goes very closely with previous point as it helps user to find items that are not "mainstream". For example, without RS, a streaming service for films and series shows cannot risk wasting recommendation slots on niche products that are less likely to satisfy user needs. Interestingly that benefit can also vary from different use cases and RS types, so evaluation is again depends on many parameters, but still shows positive influence overall.[7]

Understand user needs – one of the most important features for RS owners. RS is not only a good instrument to provide useful information to the user, it also can provide same information about user predictions to owner. This information can be used to create more popular media content or to create promotions targeted at a more appropriate group of users, for example.

Increase the user satisfaction – this benefit works only if RS not only good at recommendations themselves, but also provide well designed human-computer interaction. Combination of these will increase chances that user will use system more often and therefore more likely accept recommendation that may lead to increasing effectiveness of rest benefits mentioned here.

Increase user fidelity – this is a continuous process where user give to RS information based on his previous interactions with the system and website or application and as a result RS compute new recommendations from that. This process helps treat old users as valued customers who received more and more accurate recommendations.

2.3 General approaches to Recommender Systems

Today's one of the most popular method to classify RSs approaches was made by R. Burke[1, pp. 12–16], [8]. R. Burke distinguishes between six different classes of recommendation approaches:

Content-Based – This type of system learns to recommend items that are similar to the ones that the user liked in the past. To find out what items are similar, system is comparing them based on associated features. For example, in films and series RS, user has highly rated film made by Quentin Tarantino, therefore system will suggest other films from same director.

There are two main content-based recommendation groups of techniques. First is a classical way that implies matching the attributes of the user profile against the attributes of the items. In this case, attributes are represented as keywords that are extracted from item description.

On the other hand, another group of techniques called “semantic indexing”, represent the item and user profiles using concepts instead of keywords. They may rely on external factors such as integration of data from ontologies and encyclopaedic knowledge or internal factors, which based on hypothesis that the meaning of words depends on distributional hypothesis, which states that “Words that occur in the same contexts tend to have similar meanings”, so algorithm should analyse usage of the word in large corpora of textual documents.[9]

Collaborative Filtering – This was an original approach to RSs and it is still the most popular approach nowadays. It is so, because of its simplicity as it produces suggestions to user based on items that other users with similar tastes liked in the past. The similarity in the taste of two users is calculated based on the similarity in the rating history of the users. In other words collaborative filtering can be also named “people-to-people correlation”. Therefore such approach uses Neighbourhood-based methods.

However, there are more advanced techniques for collaborative filtering which address the issue of sparsity and limited coverage. Solution can be found with latent factor models, such as matrix factorization (e.g., Singular Value Decomposition, SVD).[10]

In addition, since 2010, a variety of deep learning solutions have been developed for RS. Deep learning can save time as it requires less feature engineering work and it is also can work with unstructured raw data (e.g., image, text) which are very common in RSs.

Community-Based – This type of systems recommends items based on the preferences of the users friends. Such systems are very close to previous, however, they rely on observation that people are more likely to listen to the recommendations of their friends rather than similar but unfamiliar persons. They take data from user’s friends ratings and based on that provide recommendations.

Demographic – these systems use demographic profile of the user. The logic behind such approach is that different segments of society would have different interests, and therefore they need different recommendations. For example, recommendation can be made regarding user age or marital status. However, there is not much in-depth research in this particular area of RSs.

Knowledge-Based – are systems that recommend items based on specific domain knowledge about how certain item features meet users’ needs and preferences and, ultimately, how the item is useful for the user.

Hybrid Recommender Systems – as it states in the name, such systems simply combine several techniques described above. Idea is to use advantages of one approach to cover disadvantages of another one. In that case, deep learning methods start to shine as they have great tools to easily build such systems.

2.4 Naive Bayes classifier

Naive Bayes algorithm is one of the most popular and simple classification machine learning algorithms. It is a supervised learning algorithm that classify the data based on Bayes theorem. Bayes theorem (Figure 1.1) is a very simple, but effective prediction tool. Basically, it calculates the probability of all outcomes and then selects the one with the highest probability. Important note, Naive Bayes classifier assumes that the effect of a particular feature in a class is **independent** of other features. For example, it means that if film will be recommended or not depending on it's genre, rating, cast, popularity and length, these features will not be considered interconnected even if they are so. Due to such assumption, Naive Bayes algorithm is easy scalable and works fast on large datasets. [11], [12]

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)}$$

Figure 1.1: Bayes' theorem *[13, Eq. 1.12]*

Here, P(Y|X) – is the probability of Y being true given that X is true. This is known as the posterior probability of Y. Same works with P(X|Y) where it will be posterior probability of X.[13]

P(Y) and P(X) – are simply the probability of Y and X respectively. They are not dependent on any data input as previous parts of theorem. This is known as the prior probability.[13]

Naive Bayes is actually a family of algorithms based on this theorem that can be used not only for RSs, but are very popular for text classification and real-time predictions due to its high speed nature.

Advantages:

- Algorithm itself is very simple, but effective
- As mentioned previously, it is very fast algorithm compared to more complex ones
- It has a low propensity to overfit, so with small dataset it can achieve better result
- It works well with high-dimensional data such as text classification, email spam detection
- Less sensitive to missing data
- Due to its simplicity, it is easy to explain the result

Disadvantages:

- In real-life problems it is actually almost impossible to find data that has entirely independent set of features.
- The model is unable to make predictions if it finds item of class which was not observed in the training data set. This problem called as Zero Frequency. It can be solved with smoothing techniques such as, for example, Laplace estimation.

2.5 Logistic regression

Logistic regression is a linear classification model. The result of prediction in that model will be a binary variable. This means that we predict variable that has only two states yes/no or as it will be in the code – 0 or 1. Logistic regression like Naive Bayes can produce prediction for multiple class problems, but in our case we look only for two classes[14], [16]. Algorithm is based on following formula:

$$p(\mathcal{C}_1|\boldsymbol{\phi}) = y(\boldsymbol{\phi}) = \sigma\left(\mathbf{w}^{\mathrm{T}}\boldsymbol{\phi}\right)$$

Figure 2.1: Logistic regression formula [11, Eq. 4.87]

Where $P(C_2|\phi) = 1 - P(C_1|\phi)$. So, the probability of class C_1 is a result of a logistic sigmoid acting on a linear function of the feature vector ϕ.[13, p. 205]

$$\sigma(a) = \frac{1}{1 + \exp(-a)}$$

Figure 2.2: Sigmoid function formula [13, Eq. 4.59]

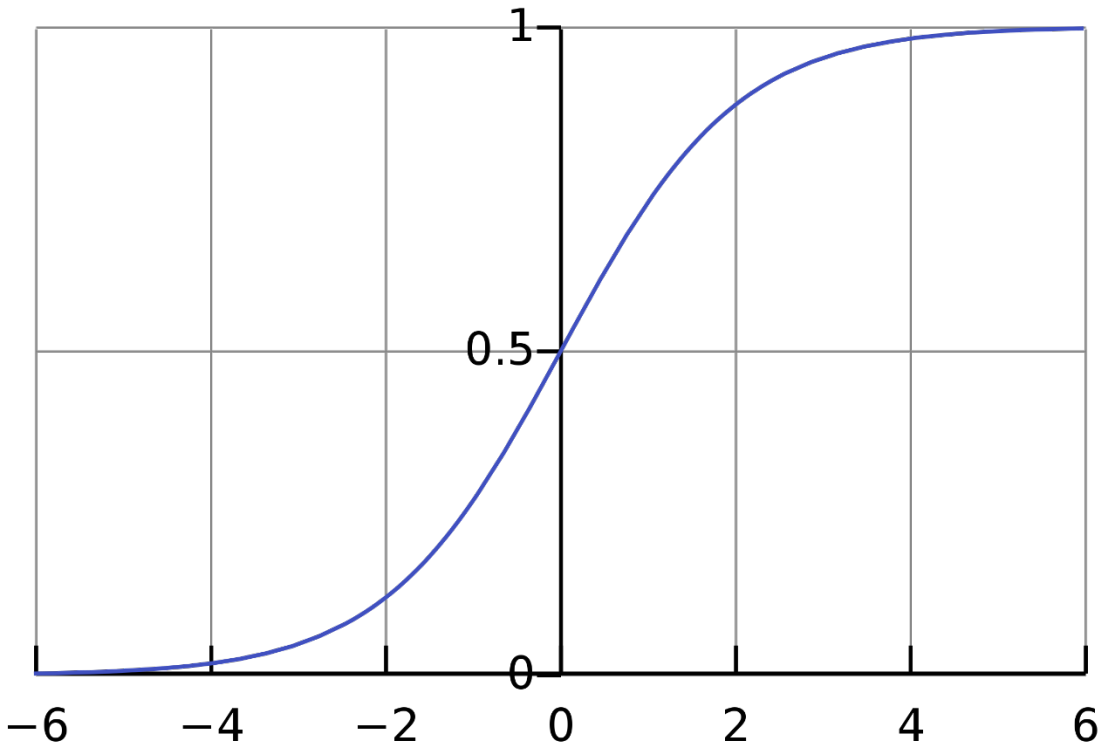

Figure 2.3: Sigmoid function plot [15]

Above you can see sigmoid function and its plot. Sigmoid function is an activation function for logistic regression that gives us a prediction on weighted sum of input features and maps it to limited interval. As we can see, the sigmoid function always takes as maximum and minimum two values 1 and 0 (Y-axis) [14], [16].

In order to train logistic regression model we now need to determine its parameters using maximum likelihood method. This process is described in “Pattern recognition and machine learning”, C. M. Bishop[13, p. 205]. We can use the derivative of the logistic sigmoid function, which can be expressed in terms of the sigmoid function itself:

$$\frac{d\sigma}{da} = \sigma(1 - \sigma).$$

Figure 2.4: Sigmoid function plot [13, Eq. 4.88]

Now we can retrieve our likelihood function. For a data set $\{\phi_n, t_n\}$, where $t_n \in \{0, 1\}$ and $\phi_n = \phi(x_n)$, with n = 1,...,N the function will be:

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} \{1 - y_n\}^{1-t_n}$$

Figure 2.5: Sigmoid function plot [13, Eq. 4.89]

where $t = (t_1,...,t_N)^T$ and $y_n = p(C_1|\phi_n)$. Now we can define an error function by taking the negative logarithm of the likelihood. Error function is a function that shows us how much our prediction differs from real result, in our case it is 1 or 0.

$$E(\mathbf{w}) = -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$

Figure 2.6: Sigmoid function plot [13, Eq. 4.90]

where $y_n = \sigma(a_n)$ and $a_n = w^T\phi_n$. Taking the gradient of the error function with respect to w, we now obtain the gradient of the log likelihood. The purpose of gradient is to optimise the error function by finding the minimum of a differentiable function. Here it is:

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\boldsymbol{\phi}_n$$

Figure 2.7: Sigmoid function plot [13, Eq. 4.91]

Finally, now we know how logistic regression can be trained using gradient and error functions and updating each of the weight vectors.

Advantages:

- Logistic regression like Naive Bayes algorithm is easy to implement and it shows good efficiency
- It can show importance of each feature based on prediction. Moreover, it shows direction of association, which can be positive of negative.
- It is also easy to interpret using previous advantage

Disadvantages:

- The main limitation is the assumption of linearity between the dependent variable and the independent variables which is usually not the case for real-world data.
- Logistic Regression can only be used to predict discrete functions
- The number of labels or observations should be bigger then number of features or it will lead to overfitting

Chapter 3

Resources

3.1 Specifications

As it follows from chosen algorithms, both variations of recommenders system will be build with machine learning, and data mining technics. Both algorithms will be written in python language. It is purely software project, so no physical materials will be needed. Both algorithms will be developed, trained and tested on the same dataset. It will contain rating for set of films and series from different users. Both algorithms will implement collaborative-filtering approach due to dataset nature.

3.2 Resource details

Dataset – https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset

Research on Logistic regression and Naive Bayes classifier algorithms and how to implement them in python – "Pattern recognition and machine learning", C. M. Bishop [13], https://developer.ibm.com/articles/implementing-logistic-regression-from-scratch-in-python/ and https://machinelearningmastery.com/naive-bayes-classifier-scratch-python/

Bibliography

[1] F. Ricci, L. Rokach, and B. Shapira, 'Recommender Systems: Techniques, Applications, and Challenges', in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. New York, NY: Springer US, 2022, pp. 1–35. doi: 10.1007/978-1-0716-2197-4_1.

[2] Z. Dong, Z. Wang, J. Xu, R. Tang, and J. Wen, 'A Brief History of Recommender Systems'. arXiv, Sep. 05, 2022. doi: 10.48550/arXiv.2209.01860.

[3] P. Resnick and H. R. Varian, 'Recommender systems', *Commun. ACM*, vol. 40, no. 3, pp. 56–58, Mar. 1997, doi: 10.1145/245108.245121.

[4] D. Jannach, P. Pu, F. Ricci, and M. Zanker, 'Recommender Systems: Past, Present, Future', *AI Mag.*, vol. 42, no. 3, Art. no. 3, Nov. 2021, doi: 10.1609/aimag.v42i3.18139.

[5] B. Pathak, R. Garfinkel, R. D. Gopal, R. Venkatesan, and F. Yin, 'Empirical Analysis of the Impact of Recommender Systems on Sales', *J. Manag. Inf. Syst.*, vol. 27, no. 2, pp. 159–188, Oct. 2010, doi: 10.2753/MIS0742-1222270205.

[6] D. Jannach and M. Zanker, 'Value and Impact of Recommender Systems', in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. New York, NY: Springer US, 2022, pp. 519–546. doi: 10.1007/978-1-0716-2197-4_14.

[7] 'Blockbuster Culture's Next Rise or Fall: The Impact of Recommender Systems on Sales Diversity'. https://pubsonline.informs.org/doi/epdf/10.1287/mnsc.1080.0974 (accessed Nov. 11, 2022).

[8] R. Burke, 'Hybrid Web Recommender Systems', in *The Adaptive Web: Methods and Strategies of Web Personalization*, P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds. Berlin, Heidelberg: Springer, 2007, pp. 377–408. doi: 10.1007/978-3-540-72079-9_12.

[9] C. Musto, M. de Gemmis, P. Lops, F. Narducci, and G. Semeraro, 'Semantics and Content-Based Recommendations', in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. New York, NY: Springer US, 2022, pp. 251–298. doi: 10.1007/978-1-0716-2197-4_7.

[10] Y. Koren, S. Rendle, and R. Bell, 'Advances in Collaborative Filtering', in *Recommender Systems Handbook*, F. Ricci, L. Rokach, and B. Shapira, Eds. New York, NY: Springer US, 2022, pp. 91–142. doi: 10.1007/978-1-0716-2197-4_3.

[11] ‘Naive Bayes Classifier Tutorial: with Python Scikit-learn’. https://www.datacamp.com/tutorial/naive-bayes-scikit-learn (accessed Nov. 14, 2022).
[12] A. Pujara, ‘Naïve Bayes Algorithm With Python’, *Analytics Vidhya*, Jul. 03, 2020. https://medium.com/analytics-vidhya/na%C3%AFve-bayes-algorithm-with-python-7b3aef57fb59 (accessed Nov. 14, 2022).
[13] C. M. Bishop, *Pattern recognition and machine learning*. New York: Springer, 2006.
[14] C. H. P. 15 February 2022, ‘Implementing logistic regression from scratch in Python’, *IBM Developer*. https://developer.ibm.com/articles/implementing-logistic-regression-from-scratch-in-python/ (accessed Nov. 13, 2022).
[15] ‘Sigmoid function’, *Wikipedia*. Oct. 29, 2022. Accessed: Nov. 14, 2022. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Sigmoid_function&oldid=1118893984
[16] J. Thorn, ‘Logistic Regression Explained’, *Medium*, Sep. 26, 2021. https://towardsdatascience.com/logistic-regression-explained-9ee73cede081 (accessed Nov. 10, 2022).