A combination of classification based methods for recommending tweets

SUMIT SIDANA, TAKEAWAY

CCS Concepts: • **Information systems** → **Recommender systems**; *Learning to rank*; • **Computing methodologies** → *Learning from implicit feedback*.

ACM Reference Format:
Sumit Sidana, Takeaway. 2020. A combination of classification based methods for recommending tweets. In *Proceedings of the Recommender Systems Challenge 2020 (RecSysChallenge '20), September 26, 2020, Virtual Event, Brazil*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3415959.3415993

1 ABSTRACT

A 3 month long RecSys 2020 challenge[1] was organized by Twitter[1]. Various kinds of implicit feedback with varying levels of sparsity were considered. In subsequent sections, I describe my approach to address this task, which helped me land on the 4th place on the final leaderboard with the final score of 20. My team name was learner_recsys and link to my code is present on GitHub[2]. This is detailed in Table 1. I used a combination of classification-based approaches, content-based recommendations, and non-personalized baselines.

2 INTRODUCTION

Twitter has become a major source of information for analyzing all aspects of daily life. One can only imagine how significant it may be to show the right set of tweets to the right set of users, so that they find the information conveyed useful. Additionally, showing the right set of tweets, which user are going to engage with, ensures that users come back to the platform, thus increasing trust in the system. To this end, Twitter conducted a 3 month long RecSys 2020 challenge. For this challenge, Twitter released around 160 Million public tweets obtained by sub-sampling for 2 weeks. The dataset contained user features, tweet features and engagement features. Characteristics of the data provided in the competition are described in Table 2. The task in this challenge was to determine the probability that engaging_user_id is going to *engage* with the content of engaged_user_id. This engagement was measured by different implicit feedback signals namely $reply$, $retweet$, $retweet\_with\_comment$ and $like$. Dataset also comprised of 100 million pseudo negatives, which were randomly sampled. These positive and negative engagements, naturally give rise to a binary classification problem. Given the metric (described in section 3), binary classification based methods proved to be most promising for this task as I will describe in the subsequent sections. Methods used in this paper are the hybrids between classification based approaches [4] and collaborative filtering based approaches [9] and this is the reason that they perform better than pure classification or collaborative filtering based methods.

[1](http://www.recsyschallenge.com/2020/)
[2]https://github.com/sumitsidana/recsys_challenge_2020

Manuscript submitted to ACM

Table 1. Details of my scores on the final leaderboard

team name	learner_recsys	
overall score	20	
Engagement	PRAUC	RCE
reply	0.1161	10.42
retweet	0.4529	22.50
retweet_with_comment	0.5037	-0.04
like	0.7221	18.28
Position on Leaderboard	4	
code	https://github.com/sumitsidana/recsys_challenge_2020	

Table 2. Overall dataset statistics. The number of tweets in each set in original data. These numbers denote the original numbers and do not reflect the tweets that were being removed to preserve user privacy

# Tweets in training set	148,075,238
# Tweets in validation set	15,127,684
# Tweets in test set	12,434,838
# engaging_users in train	25,496,088
# engaging_users in validation	7,662,966
# engaging_users in test	6,561,475
# new engaging_users in validation	2,030,122
# new engaging_users in test	1,765,440
# Languages	35

3 VALIDATION PROCEDURE AND METRICS

I first sorted the dataset on the basis of timestamp and then divided the training dataset into 90% training and the rest of the 10% as validation set. I divided the training dataset in a way that all the 7 days are also included in the validation set. This is because, the test set also contained engagement from all the 7 days. In this way I did my best to mimic the given validation set so that I can monitor the performance of my models locally and tune parameters of my models. After dividing training data further into train set and validation set, I was left with the following number of tweets in train and validation set as shown in the Table 3. It should be noted that all the scores on *rce* and *prauc* from here on-wards are the results, I obtained on the validation set I created out of training set. This is because the validation set, I created out of training set was of the same distribution as the given validation set in the competition. Barring a few exceptions, the performance on this validation set, which I created also showed the performance on the leaderboard.

3.1 Metrics

Two metrics were considered in the challenge. Relative cross entropy (RCE) and and area under precision recall curve (PR-AUC) for each *engagement*.

Table 3. I divide the given training data into train-set and validation-set in order to monitor the performance of models locally and tune parameters.

# Tweets in given training data	148,075,238
# Tweets in train set	133,267,714
# Tweets in validation set	14,807,524

Table 4. Priors for various actions.

# Action	Prior
Reply	0.028
Retweet	0.113
Retweet with Comment	0.008
Like	0.439

3.2 Priors Baseline

I first set my *priors* baseline as follows. I compute the probability of the respective implicit feedback (reply, retweet, retweet with comment and like) using the training data. This is the formula I use:

$$\frac{\#\ tweets\ with\ positive\ engagement}{Total\ \#\ of\ tweets} \tag{1}$$

This gave me the following priors for various actions as shown in the Table 4: The goal from here was to be able to

Table 5. Results with Content Based Recommendations using BERT

Action	PRAUC	RCE
Reply	0.100	4.1

outperform this simple baseline using advance recommendation models.

Table 6

Features used for reply with factorization machines
"tweet_type", "Language", "enaged_with_user_id", "engaged_with_user_is_verified", "engaging_user_id", "enaging_user_is_verified", "engagee_follows_engager", "engaged_with_user_follower_count", "engaged_with_user_following_count", "enaging_user_follower_count", "enaging_user_following_count"

4 CONTENT BASED RECOMMENDATIONS USING BERT EMBEDDING

I applied basic content based recommendation techniques using BERT [5] embedding. I took the BERT embeddings of given BERT tokens and treated each tweet as a 760 dimensional BERT embedding vector. From this vector, I created user profile vector by averaging out vectors for all the tweets, which user had positive engagement with. For predicting score of a given user, tweet pair, I took the dot product of user vector and tweet vector. This approach did not perform very well for *reply* engagement as shown in the Table 5. This could have been owing to the fact the I did not do proper scaling of the scores. But, I moved on to classification and collaborative filtering based approaches after content based recommendations did not show promise.

Model	PRAUC	RCE
FM	0.421	18.47
FFM (fields of Table 6)	0.46	23.62
FFM(fields of Table 6 + fasttext embeddings)	**0.49**	**25.7**

Table 7. Results with Field-aware Factorization Machines for $retweet$ feedback. Best results are shown in the bold

Model	PRAUC	RCE
FM	0.687	14.529
FFM (fields of Table 6)	0.724	21.188
FFM (fields of Table 6 + fasttext embeddings)	**0.736**	**22.333**

Table 8. Results with Field-aware Factorization Machines for $like$ feedback

5 CLASSIFICATION BASED MODELS

In this task, we were given positive samples (tweets, which user engaged with) and negative samples (these were sampled randomly). For test set, we were again asked to predict the probability of positive engagement. This paradigm naturally gives rise to binary classification problem. Hence, I started applying the techniques applied to such recommendation problems with binary labels.

5.1 Factorization Machines

FM [9] can be seen as a hybrid solution between classification approaches (such as Support Vector Machines [4]) and factorization approaches (such as matrix factorization [7]). FM break the independence of interaction parameters by factorizing them. The prediction function of FM is given by:

$$f(x) := \underbrace{\boxed{w_0}}_{\text{bias}} + \underbrace{\boxed{\sum_{i=1}^{n} w_i x_i}}_{\text{Linear Regression}} + \underbrace{\boxed{\sum_{i=1}^{n} \sum_{j>i} \sum_{k} v_{ik} v_{jk}}}_{\text{factorization}} \underbrace{\boxed{\langle x_i, x_j \rangle}}_{\text{interaction}} \tag{2}$$

Model	PRAUC	RCE
FM	0.024	-5.63
FFM (fields of Table 6	0.018	3.04
FFM (fields of Table 6 + fasttext embeddings)	0.033	5.554
FFM (fields of Table 11)	0.047	6.067
Prior	**0.504**	**-0.005**

Table 9. Results with Field-aware Factorization Machines for $retweet\ with\ comment$ feedback

5.2 Field-Aware Factorization Machines

Field-Aware factorization machines (FFM) [8] provide an improvement over FM by learning different latent vectors for different feature interactions. For FFM, I tried with features listed in the above section 6 as well as additional text features.

Model	PRAUC	RCE
FM	0.132	11.657
FFM (fields of Table 6)	**0.133**	**13.763**

Table 10. Results with Factorization Machines for $reply$ feedback

Table 11

Features used for retweet-with-comment with FFM
language,engaged_with_user_id,engaging_user_id,day_of_tweet, engaged_with_user_follower_count,engaged_with_user_following_count, engaging_user_follower_count, engaging_user_following_count

Parameter	Values
Optimization method	SGD
Dimension	4
Learning Rate	0.2
Regularization	0.0002
Iterations	3
threads	96

Table 12. Best Parameters for FFM for retweet

Parameter	Values
Optimization method	SGD
Dimension	4
Learning Rate	0.2
Regularization	0.0002
Iterations	2
threads	96

Table 13. Best Parameters for FFM for like

Parameter	Values
Optimization method	SGD
Dimension	1,1,16
Learning Rate	0.01
Regularization	0.028,0.01,0.01
Initial Standard Deviation	0.1
Iterations	10

Table 14. Best Parameters for FM for reply

5.2.1 Architecture of the machine. For implementation, I used the library by the original authors of FFM [3]. All experiments for FFM were run on aws instance having Ubuntu 18.04 having 96 cores and 192 GB of RAM.

[3]https://github.com/ycjuan/libffm

Parameter	Values
Optimization method	SGD
Dimension	4
Learning Rate	0.2
Regularization	0.00002
Iterations	2
threads	96

Table 15. Best Parameters for FFM for retweet-with-comment

5.2.2 FastText Embeddings. In this data, we were given BERT [5] tokens instead of raw text. I first converted the tokens into raw text using the hugging face library [4]. I then converted these raw tweets into sentence embedding using FastText library. So, eventually text of each tweet sentence was represented as 20 dimensional vector. Please also note that I use different embeddings for different languages. This is due to the fact that FastText provides a different model for each language and aligning embeddings of same word in a different language is a research problem in itself. So, in the end each language has its own 20 dimensional space and all the tweets belonging to a particular language are projected to its respective 20 dimensional space. I then proceed to augment my feature vectors as given in Table 6 with these newly learned 20 dimensional feature vectors. In the end, I ended up using *42,964,698* features and *30* fields for FFM. All the numerical fields were scaled between 0 and 1.

5.2.3 Best model for reply. For learning with Factorization machines, I used various learning methods such as markov chain monte carlo (MCMC), Alternating Least Squares (ALS) and Stochastic Gradient Descent (SGD). SGD gave the best results compared to other two and eventually gave the best results for *reply* engagement. I applied feature selection methods such as Gini Index for categorical features and Lasso for numerical features. Eventually, best predictive features used for reply are given in Table 6. The best parameters for reply are given in the Table 14 and results are shown in the Table 10.

5.2.4 Best model for retweet. For *retweet* engagement, I first used the features in the Table 6 and used FM model to learn the scores. It gave the *prauc* of 0.421 and *rce* of 18.47. Then, I experimented with FFM using the fields as in Table 6 and that boosted score of *prauc* at 0.46 and *rce* at 23.62. For the best model, I first used fields as shown in Table 6 and in addition to those original fields, I added 20 extra fields corresponding to fasttext embeddings to represent the text, which user interacted with. Adding these 20 extra fields gave an uplift in both *prauc* to 0.49 and RCE scores to 25.7. All these results are summarized in Table 7. These fields, finally gave rise to *42,964,698* features. I did a grid search over parameters and used early stopping for avoiding overfitting. Parameters used for the best model are shown in Table 12.

5.2.5 Best model for like. I tried both FM and FFM based models for *like* engagement. FM with the features of Table 6 gave prauc of 0.687 and rce of 14.529. FFM eventually performed better than FM with the same feauters giving prauc of 0.724 and rce of 21.188. Finally, for my best model of FFM, I used the fields as shown in Table 6 and added 20 more numerical fields corresponding to tweet text embeddings. This resulted in uplift of both prauc to 0.736 and rce to 22.333. The paramters for the best model after doing parameter tuning are shown in the Table 13. The results are summarized in the Table 8.

[4] https://huggingface.co/transformers/

5.2.6 Best model for retweet-with-comment. *retweet-with-comment* was the hardest to predict as the signal was really sparse with only 0.8% of the users having made the positive engagement in training data. I first tried with FM with features in Table 6 . I gave prauc 0.024 and rce of -5.63 respectively. FFM with the same set of features gave me the results of prauc of 0.018 and rce of 3.05. Then I applied feature selection methods to understand, which features are most predictive of *retweet-with-comment*. My approach for optimizing for *retweet-with-comment* was FFM with features as shown in Table 11. It gave the PRAUC of 0.047 and RCE of 6.07. Finally, a non-personalized approach of taking the prior probability of $retweet - with - comment$ gave me the better rank on the public leaderboard. All these results are summarized in Table 9. The parameters for the FFM based model are described in Table 15.

6 FUTURE WORK

One thing I could have done better in this challenge was to be able to represent text in a better way. I did not use topic modelling based techniques. A very straight forward approach I would like to apply is to run Latent Dirichlet Allocation (LDA) [2] and add topics as features in classification based models, that I described earlier. Secondly, I wanted to add content based recommendation scores as features in classification based approaches. Thirdly, I want to see if and ensemble of models performs better. Additionally, I would like to see the performance of well known learning to rank based methods such as BPR-MF [10] on this data. Finally, I would like to see the performances of recently proposed deep learning based models for implicit feedback such as wide and deep learning [3] and Neural Factorization Machines [6].

REFERENCES

[1] Belli, L., Ktena, S.I., Tejani, A., Lung-Yut-Fon, A., Portman, F., Zhu, X., Xie, Y., Gupta, A., Bronstein, M., Delić, A., and Sottocornola, G. 2020. Privacy-Preserving Recommender Systems Challenge on Twitter's Home Timeline. arXiv:cs.SI/2004.13715

[2] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2001. Latent Dirichlet Allocation. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani (Eds.). MIT Press, 601–608. http://papers.nips.cc/paper/2070-latent-dirichlet-allocation

[3] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 7–10.

[4] Corinna Cortes and Vladimir Vapnik. 1995. Support-Vector Networks. *Mach. Learn.* 20, 3 (1995), 273–297. https://doi.org/10.1007/BF00994018

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, Jill Burstein, Christy Doran, and Thamar Solorio (Eds.). Association for Computational Linguistics, 4171–4186. https://doi.org/10.18653/v1/n19-1423

[6] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. *CoRR* abs/1708.05027 (2017). arXiv:1708.05027 http://arxiv.org/abs/1708.05027

[7] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008), December 15-19, 2008, Pisa, Italy*. IEEE Computer Society, 263–272. https://doi.org/10.1109/ICDM.2008.22

[8] Yu-Chin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In *Proceedings of RecSys'16, Boston, MA, USA, September 15-19, 2016*. 43–50.

[9] Steffen Rendle. 2010. Factorization Machines. In *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, Geoffrey I. Webb, Bing Liu, Chengqi Zhang, Dimitrios Gunopulos, and Xindong Wu (Eds.). IEEE Computer Society, 995–1000.

[10] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI '09)*. AUAI Press, Arlington, Virginia, United States, 452–461.