News Popularity Prediction

Gomathi Ganesan
SBU ID: 112743868

May 17, 2020

Abstract

With the increasing usage of social media platforms, anything delivered via those are receiving high attention from the public than any other media services. Thus, knowing how popular a news would be on a given social media platform will help news providers in getting high reachability for their news by targeting the right social media to share the news content. The feedback obtained from these social media platforms are of great use for news providers to develop future news content. The research paper `https://arxiv.org/pdf/1801.07055.pdf` has collected real world data about news feedback on social media platforms GooglePlus, Facebook and LinkedIn for a period of 8 months from November 2015 to June 2016. Our goal here is to study the performance of various prediction algorithms to predict popularity of news if the feedback data follows the distribution similar to the above mentioned dataset.

Keywords: *prediction algorithm, social media news, gradient boosting, multi-level perceptron*

1 Dataset description

The dataset used is available at `http://archive.ics.uci.edu/ml/datasets/News+Popularity+in+Multiple+Social+Media+Platforms`. The datset consists of metadata about the news and the feedback data for the news from social media platforms GooglePlus, Facebook and LinkedIn. The datase contains news in four categories namely Obama, Palestine, Economy and Microsoft. The data is covered for a period of 8 months from November 2015 to June 2016. The social media feedback data contains level of popularity of a given news across 2 days split into 144 timeslices. The final popularity of a news under each social media platform is available as part of the news data. Our aim is to develop models to predict the final level of popularity of a news under each social media platform given the data about the news and level of popularity across two days after the day of publishing the news. There are a total of 12 social media feedback files ie each social media platform has a feedback file for each category of news, eg: Facbook_ economy, GooglePlus_ Obama, Facebook_ Palestine, etc.
Now let us have a look into the attributes of the news data and social media feedback data:

Table 1: Attributes in News data.

Attributes	Explanation
IDLink (numeric)	Unique identifier of news items
Title (string)	Title of the news item according to the official media sources
Headline (string)	Headline of the news item according to the official media sources
Source (string)	Original news outlet that published the news item
Topic (string)	Query topic used to obtain the items in the official media sources
PublishDate (timestamp)	Date and time of the news items' publication
SentimentTitle (numeric)	Sentiment score of the text in the news items' title
SentimentHeadline (numeric)	Sentiment score of the text in the news items' headline
Facebook (numeric)	Final value of the news items' popularity according to the social media source Facebook
GooglePlus (numeric)	Final value of the news items' popularity according to the social media source Google+
LinkedIn (numeric)	Final value of the news items' popularity according to the social media source LinkedIn

Table 2: Attributes in Social media feedback data.

Attributes	Explanation
IDLink (numeric)	Unique identifier of news items
TS1 (numeric)	Level of popularity in time slice 1 (0-20 minutes upon publication)
TS2 (numeric)	Level of popularity in time slice 2 (20-40 minutes upon publication)
TS... (numeric)	Level of popularity in time slice ...
TS144 (numeric)	Final level of popularity after 2 days upon publication

2 Pre-processing and normalizing the data

The data cleanup and pre-processing was done before training the prediction algorithms with the training set.

1. **Join data:** Since the news data and feedback data are separate it will be difficult to train with separated attributes. Hence the data from news data and feedback files of each social media platform are joint based on news_ id. This results in 3 data files where each file contains all news and feedback data for one social media platform. The attributes in the joint data includes the following:

Table 3: Attributes in Social media feedback data.

Attributes	Explanation
IDLink (numeric)	Unique identifier of news items
Title (string)	Title of the news item according to the official media sources
Headline (string)	Headline of the news item according to the official media sources
Source (string)	Original news outlet that published the news item
Topic (string)	$\langle media \rangle$ _ $\langle category \rangle$ eg: Facebook_ economy, Facebook_ Obama
PublishDate (timestamp)	Date and time of the news items' publication
SentimentTitle (numeric)	Sentiment score of the text in the news items' title
SentimentHeadline (numeric)	Sentiment score of the text in the news items' headline
Facebook (numeric)	Final value of the news items' popularity according to the social media source Facebook
GooglePlus (numeric)	Final value of the news items' popularity according to the social media source Google+
LinkedIn (numeric)	Final value of the news items' popularity according to the social media source LinkedIn
TS1 (numeric)	Level of popularity in time slice 1 (0-20 minutes upon publication)
TS2 (numeric)	Level of popularity in time slice 2 (20-40 minutes upon publication)
TS... (numeric)	Level of popularity in time slice ...
TS144 (numeric)	Final level of popularity after 2 days upon publication

2. Drop the **Headline** and **Title** attributes. We already have sentiment scores for headline and title. So we can safely ignore these attributes as its already present in the form of sentiment scores.

3. Among the remaining attributes, we have **Source** and **Topic** which are string values. Each unique value of these attributes weremapped to an integer value starting from 0. This is the data normalization applied to these fields.

4. The **PublishDate** attribute was parsed as a date and converted to integer(integer value of date $\mod 10^9$) .

5. Now, all the attributes hold numeric values and hence can be considered for prediction algorithm.

6. The values of the attributes and the targets are **standardized** by removing mean and scaling to unit variance.

3 Prediction Algorithms

After all pre-processing and data normalization, we now have a dataset for each media with information about news, popularity value across a 2 day interval after publication which is divided into 144 timeslices and the target final popularity prediction in that media. For all the algorithms, we are using **Mean absolute error(MAE)** and **Root mean squared error(RMSE)** as performance metrics.

3.1 Random Forest Regressor

Random forest regressor is a meta estimator that averages values from various classifying decision trees built for subsets of training data. The averaging controls over-fitting and

improves accuracy.

The random forest regressor has been ran for different values of estimators like 10, 30, 50, 70 and 90. The following hyper-parameters are used for the random forest regressor:

Table 4: Parameters for Random forest regressor.

Parameters	Values
n_ estimators	values = 10,30,50,70,90
random_ state	0, in order to get same value for each run
min_ samples_ split	2 , min no.of samples o split a node
depth	expanded till all leaves are pure or till all leaves contain less than min_ samples_ split samples
minimal cost-complexity pruning	0
minimum impurity split	None, no threshold specified in order to early stop splitting

- **Facebook data:** Among all the random forest regressor with different no.of estimators run, the best prediction achieved was with **n_ estimators = 10 with MAE=0.459 and RMSE=18.172**. The model seems to overfit with training data with more number of decision trees.

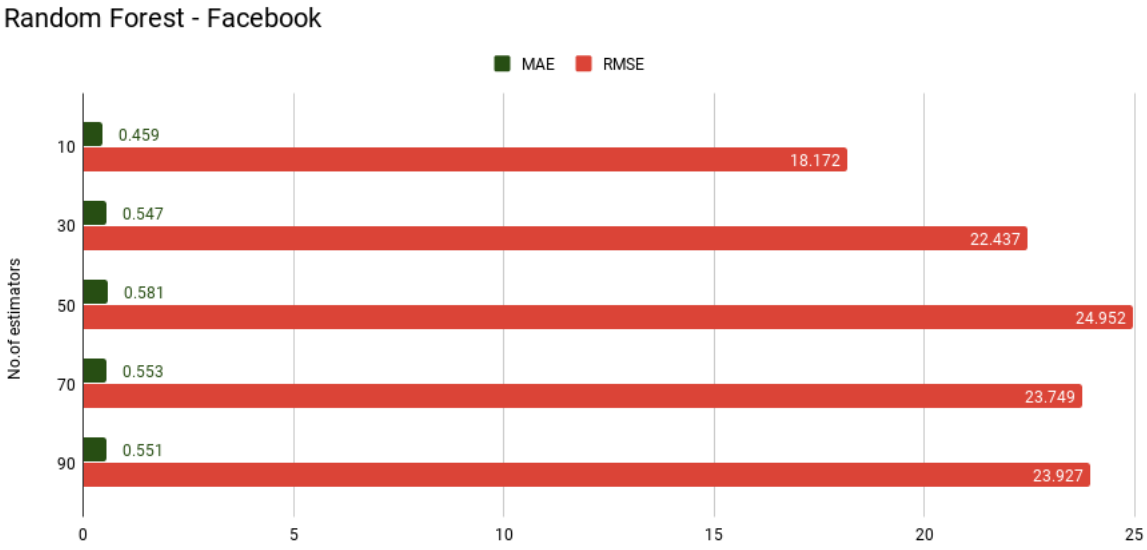

Figure 1: Popularity prediction for Facebook data using Random Forest

- **GooglePlus data:** Among all the random forest regressor with different no.of estimators run, the best prediction achieved was with **n_ estimators = 70 with MAE=0.018 and RMSE=0.917**. There is no overfitting of training data with more number of decision trees.

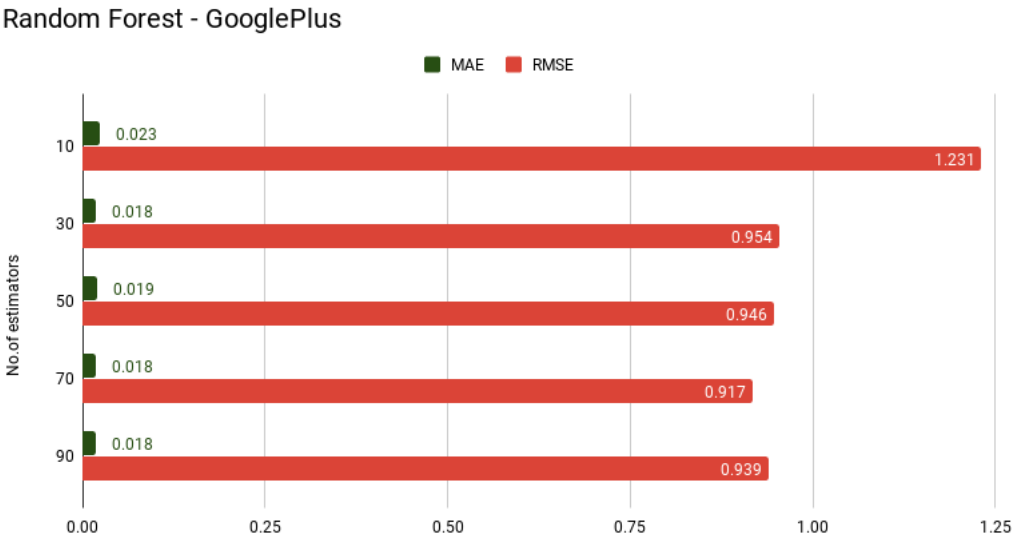

Figure 2: Popularity prediction for GooglePlus data using Random Forest

- **LinkedIn data:** Among all the random forest regressor with different no.of estimators run, the best prediction achieved was with **n_ estimators = 30 with MAE=0.13 and RMSE=5.872**. The model seems to overfit with training data with increasing number of decision trees. The model underfits or overfits if the number of decision trees in the random forest is below 30 or more than 30.

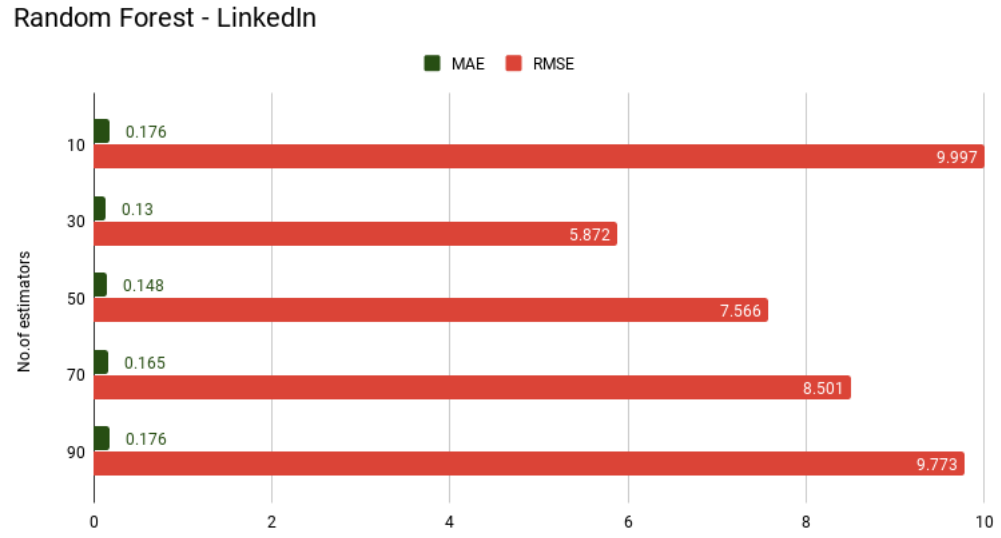

Figure 3: Popularity prediction for LinkedIn data using Random Forest

3.2 Multi-Layer Perceptron

Multi-Layer Perceptron(MLP) consists of combination of neural networks. The loss square function is optimised using functions like SGD(stochastic gradient descent), etc.

Here the MLP Regressor was ran for multiple combination of layers. Also, since the networks need to converge the number of attributes can't be enormous, for MLP an extra data pre-processing set was performed where popularity values for all the 144 timeslices were accumulated to a single value.

The following hyper-parameters are used for the MLP regressor:

Table 5: Parameters for MLP regressor.

Parameters	Values
layers	3 layers each with 100 units, 3 layers with (100,75,50) units, 2 layers with (100, 50) units
maximum iteration	500
activation function	RELU
solver	adam
alpha	0.001
learning rate	constant
random state	0

- **Facebook data:** The best accuracy with **MAE=27.365 and RMSE=161.079** was obtained while using a MLP regressor with **3 layers of (100,75,50) units.**

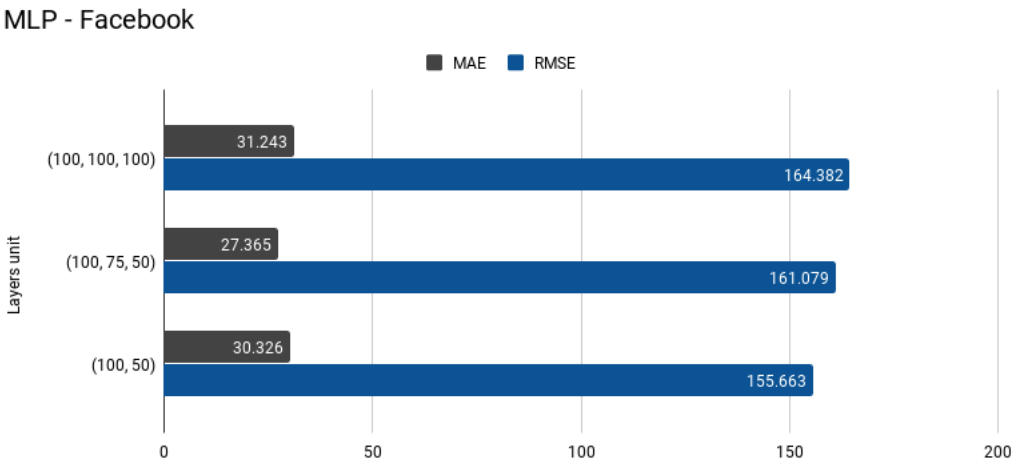

Figure 4: Popularity prediction for Facebook data using MLP

- **GooglePlus data:** The best accuracy with **MAE=1.043 and RMSE=4.604** was obtained while using a MLP regressor with **3 layers of (100,100,100) units.**

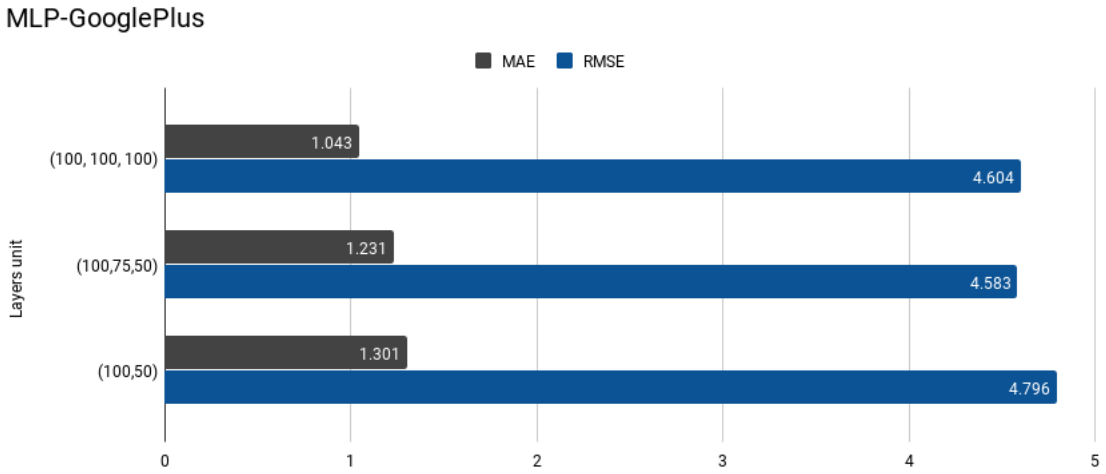

Figure 5: Popularity prediction for GooglePlus data using MLP

- **LinkedIn data:**The best accuracy with **MAE=4.449 and RMSE=23.77** was obtained while using a MLP regressor with **3 layers of (100,100,100) units.**

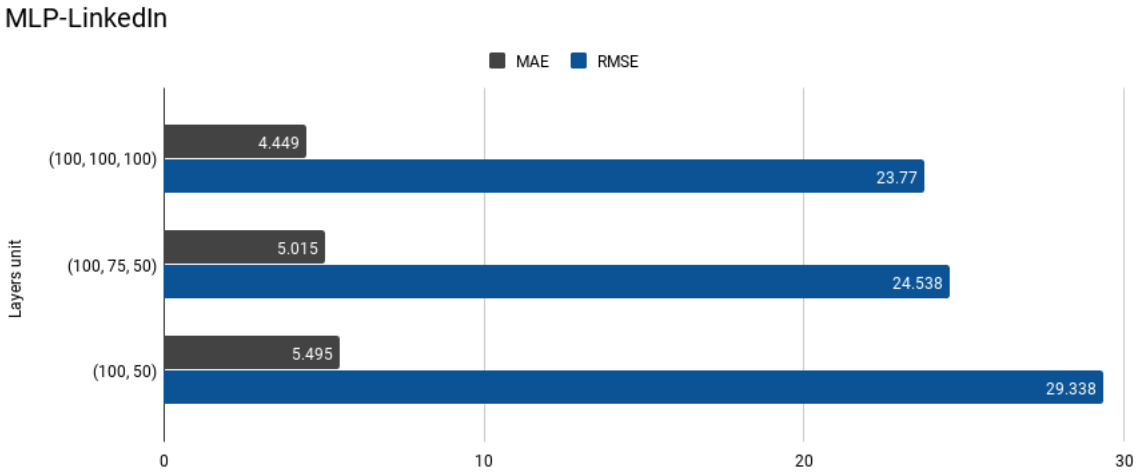

Figure 6: Popularity prediction for LinkedIn data using MLP

A good performance was observed using 3 layers than 2 layers.

3.3 Gradient Boosting Regressor

Gradient boosting is an additive model. In each stage a regression tree is fit on the negative gradient of the given loss function. Here differentiable loss functions are optimized.

The gradient boosting regressor has been ran for different values of estimators like 3,6,9,12,15. The following hyper-parameters are used for the gradient boosting regressor:

Table 6: Parameters for Gradient boosting regressor.

Parameters	Values
n_ estimators	values = 3,6,9,12,15
random_ state	0, in order to get same value for each run
min_ samples_ split	2 , min no.of samples o split a node
depth	2
loss(loss function to be optimized)	least square regression
minimum impurity split	None, no threshold specified in order to early stop splitting

- **Facebook data:** Among all the gradient boosting regressor with different no.of estimators run, the best prediction achieved was with **n_ estimators = 15 with MAE=15.118 and RMSE=69.419**. The model seems to underfit with training data with less number of decision trees.

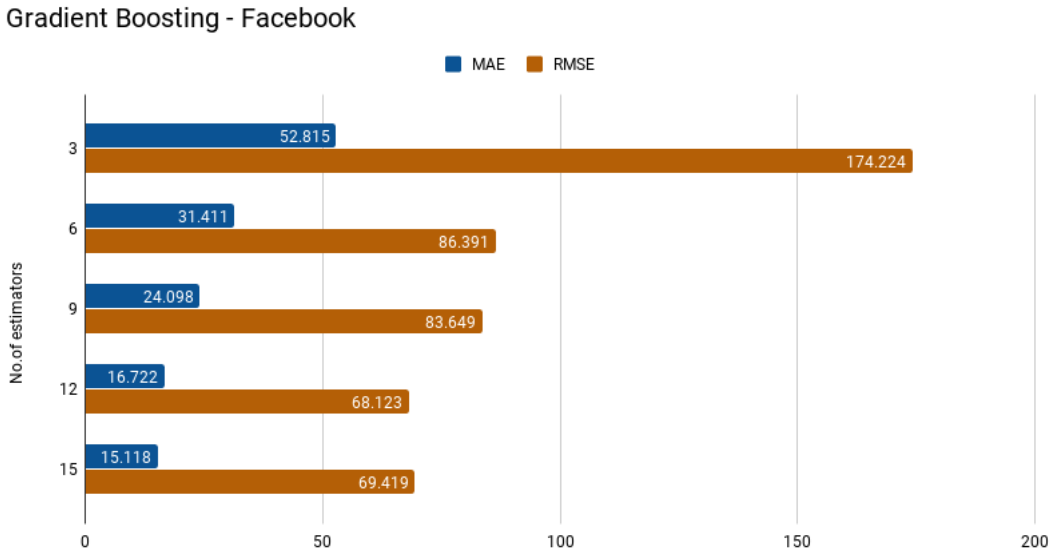

Figure 7: Popularity prediction for Facebook data using Gradient Boosting

- **GooglePlus data:** Among all the gradient boosting regressor with different no.of estimators run, the best prediction achieved was with **n_ estimators = 12 with MAE=0.578 and RMSE=2.235**. There is no overfitting of training data with more number of decision trees.

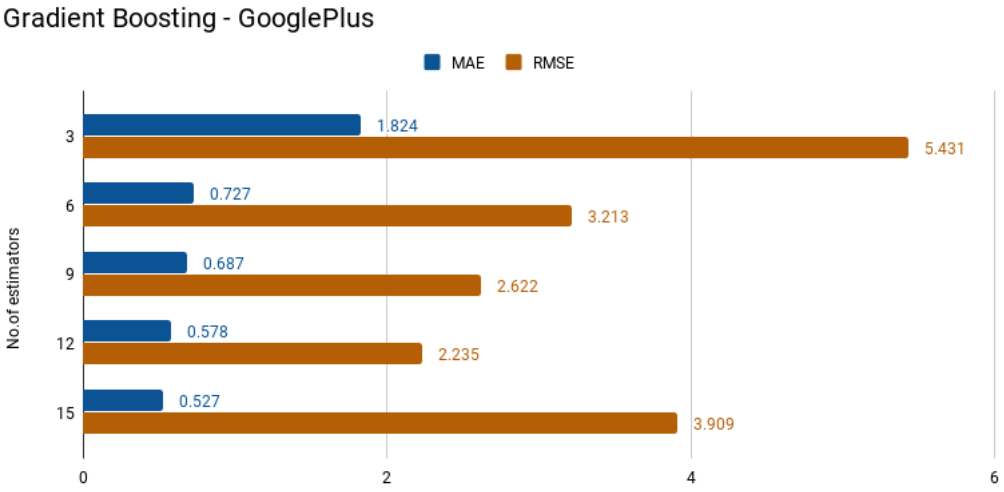

Figure 8: Popularity prediction for GooglePlus data using Gradient Boosting

- **LinkedIn data:** Among all the gradient boosting regressor with different no.of estimators run, the best prediction achieved was with **n_estimators = 12 with MAE=3.256 and RMSE=20.646**. The model seems to overfit with training data with increasing number of decision trees. The model underfits or overfits if the number of decision trees in the random forest is below 12 or more than 12.

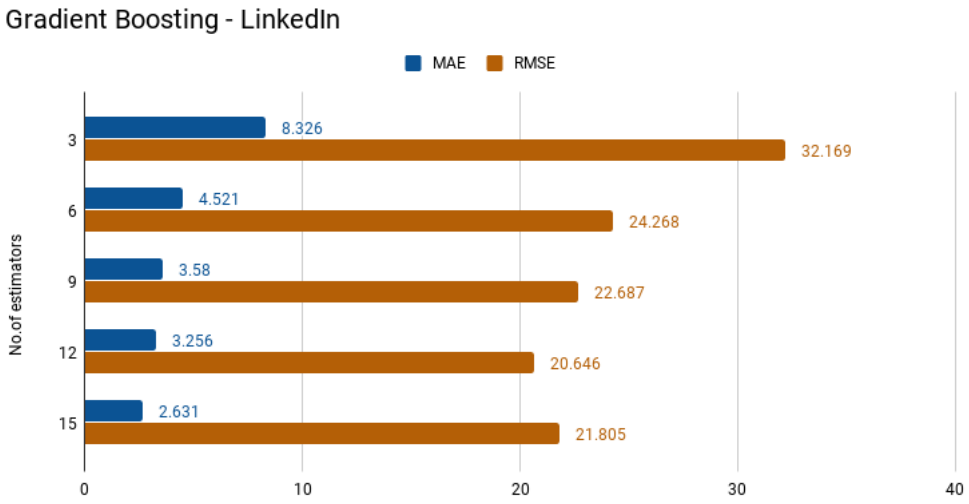

Figure 9: Popularity prediction for LinkedIn data using Gradient Boosting

4 Discussion and Conclusions

The best performance metrics for popularity of news prediction by different algorithms for each social media platform is shown below.

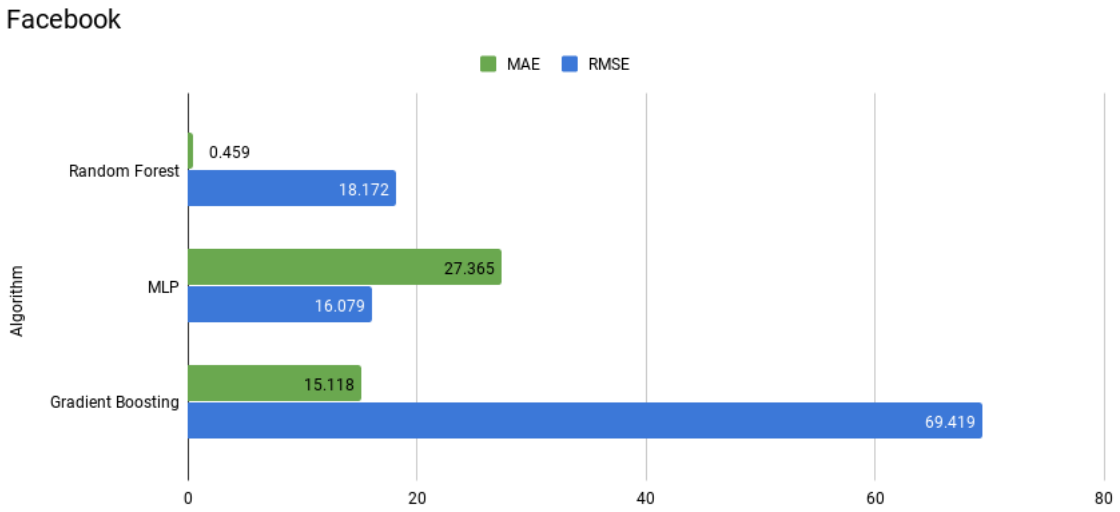

Figure 10: Performance metrics for Facebook data

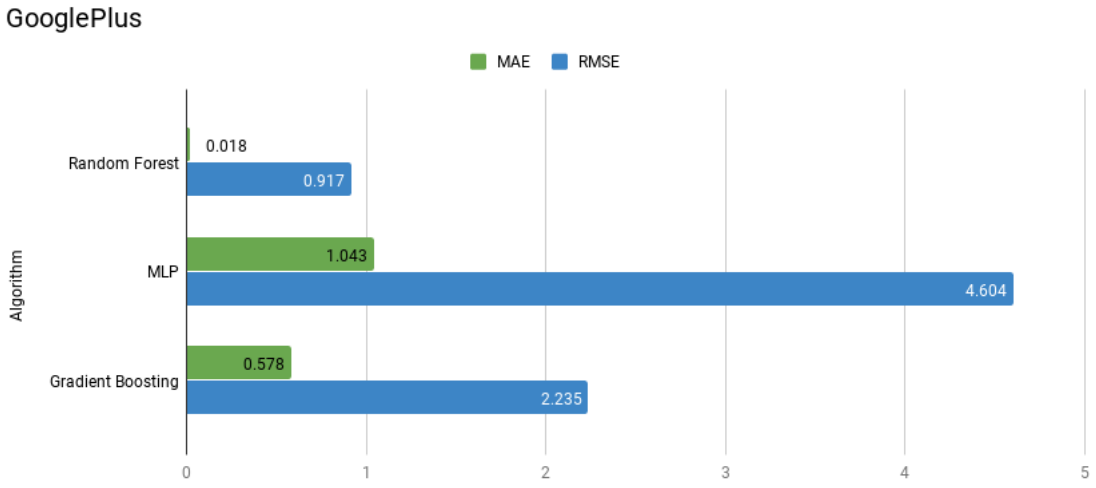

Figure 11: Performance metrics for GooglePlus data

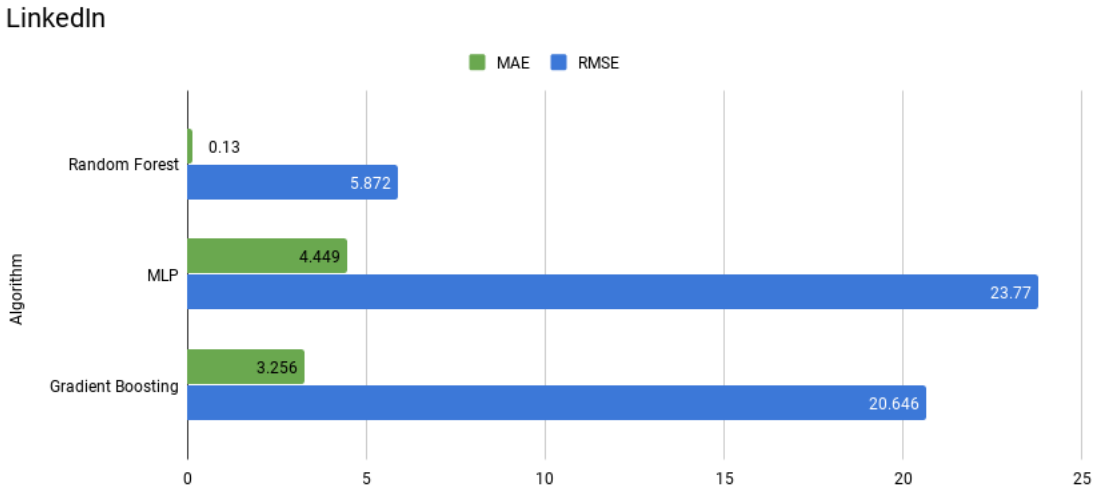

Figure 12: Performance metrics for LinkedIn data

It could be clearly seen that **Random Forest regressor** performed well for all the social media platform. Training with GooglePlus data was successful for all the algorithms. Improvisation to yield better results can include reducing the feature dimension by using feature selection to select high influential features, cross-validation to know more about overfit and underfit in the modelsand tuning the hyper-parameters.

5 References

- Template for report referred from https://www.overleaf.com/project/5ec0b395648f7b000
- https://arxiv.org/pdf/1801.07055.pdf
- http://www.techscience.com/cmc/v61n1/23099/pdf
- http://archive.ics.uci.edu/ml/datasets/News+Popularity+in+Multiple+Social+Media+Platforms
- https://www.geeksforgeeks.org/random-forest-regression-in-python/
- https://stats.stackexchange.com/questions/305046/best-way-to-evaluate-a-random-forest-model-accuracy-on-continuous-data
- https://stackoverflow.com/questions/44101458/random-forest-feature-importance-chart-using-python
- https://stackoverflow.com/questions/14037540/writing-a-python-list-of-lists-to-a-csv-file
- https://stackoverflow.com/questions/27135470/python-how-to-create-a-directory-and-overwrite-an-existing-one-if-necessary
- https://stackoverflow.com/questions/41892680/how-can-i-specify-the-discrete-values-that-i-want-to-plot-on-the-x-axis-matplot
- https://blog.datadive.net/selecting-good-features-part-iii-random-forests/
- https://medium.com/@AI_with_Kain/understanding-of-multilayer-perceptron-mlp-8f179c4a135f