Retweet Count Prediction

Shala Chen
(647)778-8874
shalachen1.0@gmail.com

I. Data Pre-process

For data pre-process, I fetched most of the data from the csv file and transformed them into numerical form, except ids and names.

- created_days: counted how many days from the day it was posted till today. A tweet posted earlier may have more retweets.
- compound,neg,neu,pos: sentiment values from the list. I used 0 for all missing values.
- postLength: length of the post. A longer post may contain more information that there might be more people want to retweet it.
- favorite_count: more favorite count may indicate it was popular.
- coordinates: if there's an associated coordinate then set this to 1, if there's none, set this to 0.
- user_descrip_length :the length of user's description. A user with a long description may have more detailed information posted.
- followers_count: an account with more followers might be better exposed, thus get more retweets.
- url: if the user has a url in file then set it to 1, if it's none, set to 0.
- verified: if the user is verified, set this to 1, else set this to 0.
- friends_count: count friends. A tweet posted by users with more friends might be better exposed, thus get more retweets.
- location: if a tweet had a location information, then set this to 1, else, set this to 0.
- hashtags: counted how many hashtags this tweet had, not the actual hashtags.
- place: if there's an associated place information, then set this to 1, if there's none, set this to 0.
- check_URL: if there was an URL, set this to 1, else set this to 0.

I excluded tweets that didn't have a complete brand information or didn't have rts_likes information. Some samples didn't have a json_data, I filled those necessary features with 0. rts_likes column was separated into another file. The finalized new data had 154514 samples with 17 features each. I took 140000 samples as training data set and the rest 14514 samples as testing data set.

II. Classifications

I. Decision Tree

I used decision trees classifier as my first try. Unlike images data, the features in this data set were not related with each other directly. It would not affect the final result even if I changed features' sequence. Decision tree was good for this kind of data set. It was able to split based on single features. After training and testing, the accuracy was 0.788, and the variance was 1254753.

II. Random Forest

I applied various number of estimators from 1 to 31 while building the random forest model. When there were 18 estimators, the accuracy was 0.873 and the variance was 1340817. However when there were more than 5 estimators, the accuracies and variances were all similar.

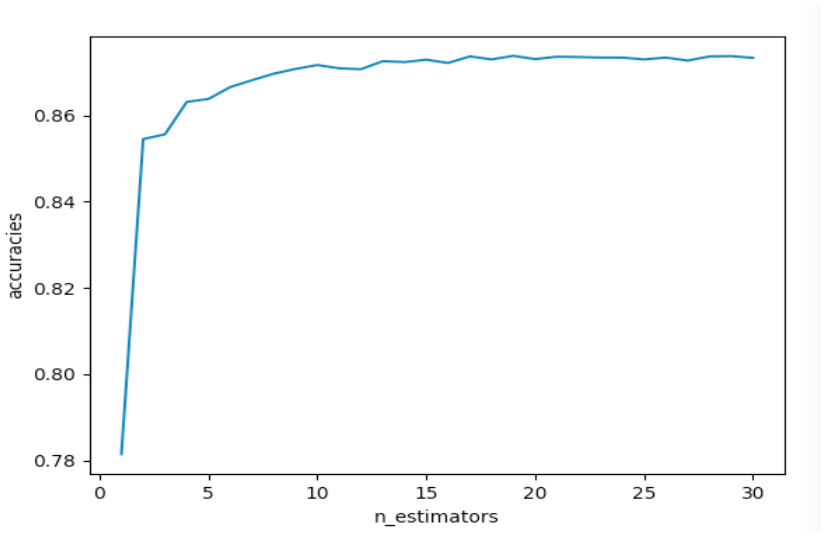

Figure 1: Random Forest accuracy for different estimators

III. Neural Network

I tried this data set with neural network classifier with 3 hidden layers with 10 units each.The accuracy was 0.870 and the variance was 1326306.

IV. Linear Regression and Logistic Regression

Then I applied linear regressions to the samples and got the mean squared error of 801897.87 and variance score of 0.41. For logistic regression, the mean squared error of 1333062.04 and variance score of 0.87.

III. Discussion

Table 1: Accuracies and Variances for classifiers

	Accuracy	Variance\Mean Squared Error
Decision Tree	0.788	1254753
Random Forest	0.873	1340817
Neural Network	0.870	1326306
Linear Regression		801897
Logistic Regression		1333062

The Decision Tree had the highest accuracy, I believed it was because there wasn't many features in each sample, thus random forest and neural network classifiers didn't had a change to show their strength.

I used classifiers to predict rts_likes, but it didn't mean the prediction results were classes. rts_likes was the number count of retweets, rather than have a high accuracies, a lower variance was more important. If the class prediction was off by 1, it would be counted as a "wrong class", however, while there were so many "wrong classes", it's better to have a class that was closer to the true label, that's why we wanted to use prediction variance rather than accuracy to make final decisions.

In this case, although the random forest classifier had a higher accuracy, the linear regression model had the lowest variance or mean squared error. It was reasonable because linear regression was looking for a value while other classifiers were predicting a class label. So I'd choose linear regression to predict rts_likes with a prediction variance of 801897.