Black Friday Data Science Hackathon

Analytics Vidhya, a community of analytics professionals and data scientists hosted an online data analytics hackathon on 20–22nd of November, 2015. After my very bad performance on their previous hackathon, I was geared up to do well, try new things and learn during the process.

Problem Statement

The challenge was to predict purchase prices of various products purchased by customers based on historical purchase patterns. The data contained features like age, gender, marital status, categories of products purchased, city demographics etc.

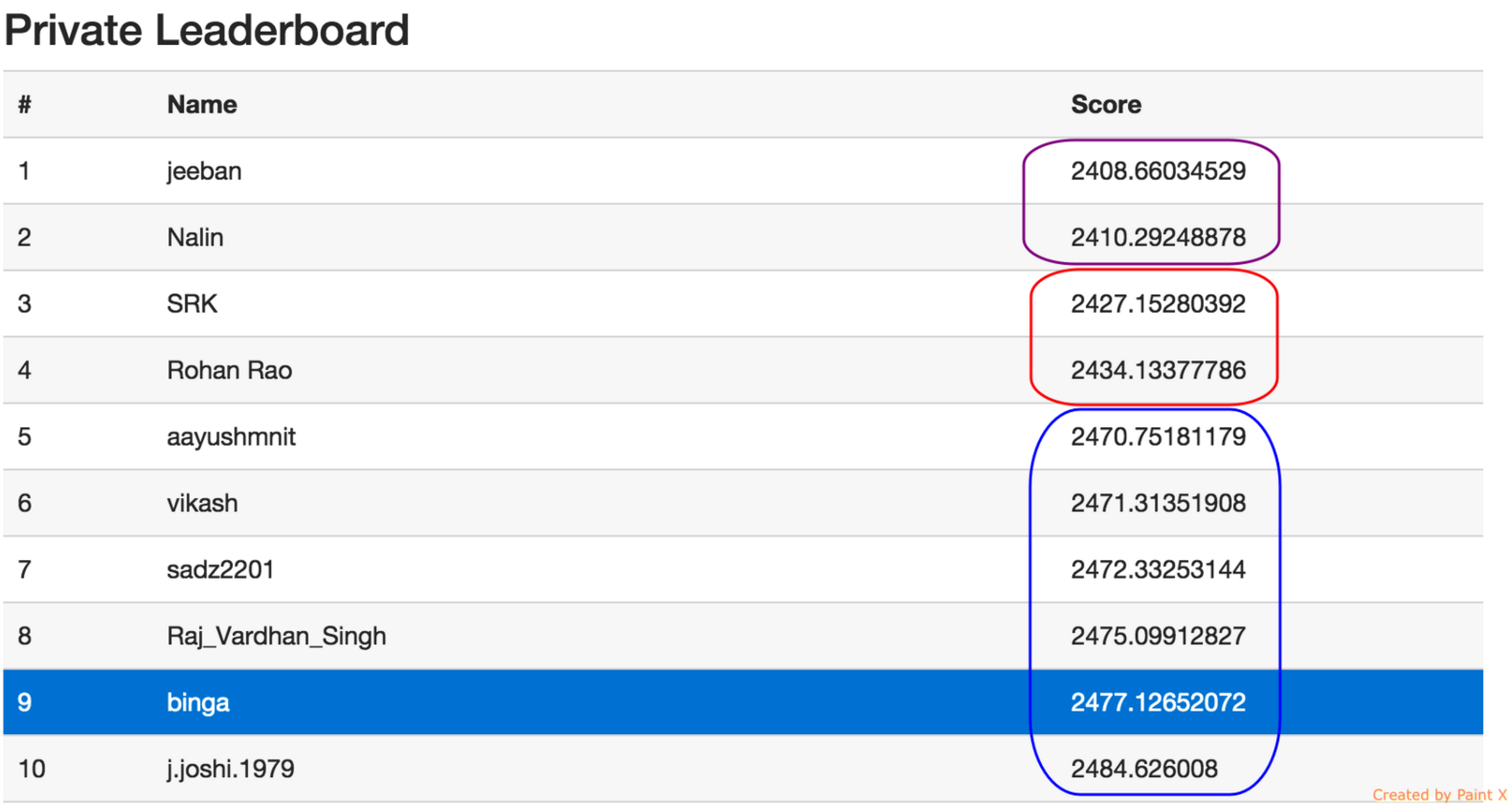
After working on a host of classification problems, a regression problem this time was very refreshing for me and I wanted to make sure I try different approaches to find out various intricacies of how to effectively handle regression problems.

Approach

As a first step, I did some exploratory analysis work, trying to identify if there are any outliers and any missing values. The dataset size was pretty big as well. This is when I realized that it is a very nice dataset *(hunch)*. No outliers. Only a couple of columns having missing values and a couple of submissions showed nice correlation between Cross-validation scores and Leaderboard scores. My hunch was validated!

As usual, I started with linear models and moved soon to Random Forests and XGBoost. One of the unique aspects of this competition was apart from XGBoost, no other model was doing well. Not even marginally close. I guess this happened mainly due to the large number of user_ids and product_ids in the dataset. Simpler models were unable to capture the non-linearity in the dataset.

Even after extensive tuning of the model hyperparameters, I couldn’t break the 2470 RMSE score barrier which was 8th place on leaderboard. My XGBoost was giving me around a 4-fold cross-validation score of 2477 RMSE. I thought the guys at the top were using something more (than just tuning the model) to reach lower RMSE. You could see that (1,2), (3,4) and (5,6,7,8,9,10) form clusters on leaderboard.



Private Leaderboard

#	Name	Score
1	jeeban	2408.66034529
2	Nalin	2410.29248878
3	SRK	2427.15280392
4	Rohan Rao	2434.13377786
5	aayushmnit	2470.75181179
6	vikash	2471.31351908
7	sadz2201	2472.33253144
8	Raj_Vardhan_Singh	2475.09912827
9	binga	2477.12652072
10	j.joshi.1979	2484.626008

Private leaderboard

So, we were clearly missing out on some information. So, what did they do? I mentioned it below. Keep reading!

After trying many more models (explained below), I just averaged 5 XGBs to ensure seed stability and finished 9th on the leaderboard. Finally, capping the highest target to 99.9 percentile and replacing negative predictions to minimum target value in train set improved my score by ~1–2 .

What didn’t work for me

Linear models and Random Forests—very bad CV (around 2680).
Tried something new—Regularized Greedy Forests. Gave second best CV score after XGBoost (2544) but not so good to ensemble.
Tried Neural Networks. But, I couldn’t get a good network configuration in a 3-day hackathon.
Tried Matrix Factorization. But, I had some infrastructure issues.
Feature Engineering. Tried creating new features. But, I couldn’t obtain a good CV score—probably due to leakage.

What set winners apart and my thoughts

3rd and 4th place winners (SRK and Rohan Rao) used good feature engineering to gain significant lead. I should learn that from them
2nd place winner (Nalin) used Matrix factorization to get a good lead. I tried this too but had some infrastructure issues. In retrospect, this looks to be a great idea because of a large number of levels in the product_id variable and user_id variable. MF captures

the feature interactions automatically and thus it proved to be a differentiator. Awesome Nalin.
1st place winner (Jeeban) used an ensemble of Gradient Boosted Trees and Deep Learning. Well, his neural network wasn't technically *'deep' Lol!* I had a mini heart-attack when I came to know his neural network structure. It was input x 6 x 1. Neural networks are soo easy! Hehehe. I could have never come up with such a simple structure. But, it worked great. Awesome Jeeban!

Learnings, tips and tricks!

When you have categorical variables with high cardinality, try (definitely try) matrix factorization. It might work (it actually works more often that not!)
If you are interested in dealing with large number of levels in categorical variables, don't step back because of RAM. Try H2o. It's brilliantly engineered.
Feature engineering always sets you apart!

Well, I finished 10th and 9th on public and private leaderboards respectively out of 162 participants. It's a decent finish, but I think I could have done much better. In retrospect, I am happy that I tried new approaches and it was a weekend well spent.

Thank you Analytics Vidhya for conducting this hackathon. Come up with nice datasets and we will never disappoint you! **;-)**

For my code: https://github.com/binga/AnalyticsVidhya_BlackFriday

The datasets are available at: https://drive.google.com/folderview?id=0B4-c0K3SnXFcbWFRbjNUaWl1ekU&usp=sharing—go ahead and give it a shot if you are interested in exploring the dataset.

#DataScience #Hackathon #MachineLearning #XGBoost #DeepLearning #MatrixFactorization