Home (https://www.analyticsvidhya.com/) › Machine Learning (https://www.analyticsvidhya.com/blog/category/machine-lear...

Winner's Approach – Rampaging DataHulk MiniHack, AV DataFest 2017

MACHINE LEARNING (HTTPS://WWW.ANALYTICSVIDHYA.COM/BLOG/CATEGORY/MACHINE-LEARNING/)

SHARE (http://www.facebook.com/sharer.php?u=https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/&t=Winner's%20Approach%20–%20Rampaging%20DataHulk%20MiniHack,%20AV%20DataFest%202017) (https://twitter.com/home?status=Winner's%20Approach%20–%20Rampaging%20DataHulk%20MiniHack,%20AV%20DataFest%202017+https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/) (https://plus.google.com/share?url=https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/) (http://pinterest.com/pin/create/button/?url=https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/&media=https://s3-ap-south-1.amazonaws.com/av-blog-media/wp-content/uploads/2017/04/07053819/MH_Hulk_icom.png&description=Winner's%20Approach%20–%20Rampaging%20DataHulk%20MiniHack,%20AV%20DataFest%202017)

Introduction

> *Who are you competing with?*

While participating in a hackathon, a lot of people think that they are competing against the top data scientists. While, in reality, most of us really compete with ourselves. The ones who improve themselves, the ones who competing with their own previous self and push their limits to become better are always the eventual winners.

We see this happen very frequently on Analytics Vidhya. We saw this again in our first ML contest of DataFest 2017 (https://analyticsvidhya.com/datafest-2017/) – Rampaging DataHulk (https://datahack.analyticsvidhya.com/contest/avdatafest-rampaging-datahulk/). In this minihack, we saw experienced professionals, students & previous winners compete with each other for the top 3 ranks. A total of 1458 people participated in the minihack. The competition began at 6 PM on 2 April marking the first competition in DataFest.

After a fist to fist battle in true "Hulk-athon" style, we saw something remarkable. Something which hasn't happened for a while on Analytics Vidhya. The top 3 ranks were bagged by first-time winners. To top it up, the winner is still in his college days! That is just a testimony to the competitiveness and openness of the platform.

Like always, the winners of the competition have generously shared their detailed approach and the codes they used in the competition.

If you missed out the fun this weekend, make sure you participate in the upcoming Machine Learning Hackathon (https://datahack.analyticsvidhya.com/contest/machine-learning-hackathon/) & The QuickSolver MiniHack (https://datahack.analyticsvidhya.com/contest/avdatafest-the-quicksolver/).

The problem statement

The problem statement revolved around a hedge fund company "QuickMoney". They rely on automated systems to carry out trades in the stock market at inter-day frequencies. They wish to create a machine learning-based strategy for predicting the movement in stock prices for maximizing their profit. So they were seeking out a help from top data scientists.

(https://datahack.analyticsvidhya.com/contest/avdatafest-the-quicksolver/)

Stock markets are known to have a high degree of unpredictability but it is possible to beat the odds and create a system which will outperform others.

The participants were required to create a trading strategy for maximizing their profit in the stock market. The task was to predict the probability whether the price for a particular stock for next day market close will be higher(1) or lower(0) compared to the price for market close today.

Winners

The winners used different approaches and rose up on the leaderboard. Below are the top 3 winners on the leaderboard:

Rank 1: Akash Gupta (https://datahack.analyticsvidhya.com/user/profile/akashgupta222)

Rank 2: Prince Atul (https://datahack.analyticsvidhya.com/user/profile/prince.p13029)

Rank 3: Santanu Pattanayak (https://datahack.analyticsvidhya.com/user/profile/santanu.pattanayak011183@gmail.com)

Here are the final rankings of all the participants at the leaderboard (https://datahack.analyticsvidhya.com/contest/avdatafest-rampaging-datahulk/lb).

All the Top 3 winners have shared their detailed approach & code from the competition. I am sure you are eager to know their secrets, go ahead.

Rank 3, Santanu Pattanayak

Santanu Pattanayak

Santanu Pattanayak (https://www.linkedin.com/in/santanu-pattanayak-99843812/) is Lead Data Scientist at GE Digital. He often participates in machine learning competitions on Analytics Vidhya. He likes to challenge himself.

Following is the approach he took for the Analytics Vidhya Rampaging Datahulk Competition. He secured 3rd place in the competition with a private Leaderboard Score of 0.678784:

1. First, I did some exploratory data analysis. I checked the number of records in train and test datasets and checked whether there is any class imbalance that we need to deal with. The training dataset was quite balanced with 45% of the data belonging to the positive class. Since the dataset sizes were satisfactory i.e. 702739 train records and 101946 test records hence class imbalance adjustments were not necessary. Then I checked the number of different stocks in both train and test and checked whether all the stocks in test are there in train dataset or not. The train dataset

has 1955 stocks while the test dataset has 2118 stocks. Since the test has more stocks clearly stock id cannot be used as a feature since the model would learn nothing about those stock ids that are there in test but not in train.

2. The main task as in most of the machine learning tasks is to do proper feature engineering. So, spend quite a bit of time thinking what would be good features with respect to the output that we are going to predict – that is whether the sales of tomorrow's market close is going to be higher than today's market close.

There were missing values in the below fields:

```
Three_Day_Moving_Average
Five_Day_Moving_Average,
Ten_Day_Moving_Average
Twenty_Day_Moving_Average
```

I replaced the missing values with 99999 and created indicator variables indicating whether these fields have missing values.

Then I created few variables capturing the difference in the moving averages. For example – (Three_Day_Moving_Average – Ten_Day_Moving_Average). I created such variables for each pair of the moving average variables.

I created couple of features by taking the sum and difference of the variables Positive_Directional_Movement and Negative_Directional_Movement. Similarly, I created two features by taking the sum and difference of the variables True_Range and Average_True_Range.

Also, I created few features to hold the moving average of the days prior to a specific period as below:

```
df['MA_last_10_3'] = (df['Ten_Day_Moving_Average']*10 – df['Three_Day_Moving_Average']*3)/
7


df['MA_last_10_5'] = (df['Ten_Day_Moving_Average']*10 - df['Five_Day_Moving_Average']*5)/5


df['MA_last_5_3'] = (df['Five_Day_Moving_Average']*5 - df['Three_Day_Moving_Average']*3)/2
```

Here the first variable is computing the average of the 7 days prior to the last 3 days.

3. Once I build these features then I split the training data into two parts – 80% of the data for training the models and 20% for validation purpose. Below are the models that I tried –

Gradient boosting from graphlab – It's always easy to work with graphlab since you can input a dataframe along with the features and target unlike most of the other packages wherein you would have to create a numpy matrix or a sparse matrix before the algorithms can be invoked. Experimented with 300,500 and 700 trees, with the class weights set to "auto", tree depth of 6, min child weight and minimum loss reduction set to "4" each. Also, the column subsample and the row subsample was set to 80 percent.

It gave good performance with validation logloss of around 0.6820 and public leaderboard of around 0.6855

I tried my hand at a small **neural network through Keras** with two hidden layers of 300 units each and dropout of each hidden layer set to 0.5. For the hidden layers I chose activation as 'RELU' and the output layer as 'sigmoid' and got a logloss of around 0.688 in both validation and in leaderboard.

Since the neural network and Gradient boosting are very different models I tried to take the mean of their predicted probabilities and the public leaderboard logloss improved to 0.6831.

Still I was not able to enter the 0.67 range.

Next, I tried my luck at **xgboost** with kind of similar configuration as that of the graphlab gradient boosting model.

I experimented a bit with the number of trees and finally got the best results with the below parameters.

No of trees	700
Column subsample	0.8
Row subsample	0.8
L2 regularization	2(lambda)
L1 regularization	0.02(alpha)
Minimum child weight	4

Objective	Binary:logistic
Booster	Gbtree
Eta	0.02
Early stopping round	20

The above model gave me 0.6780 logloss on Public leaderboard (9th rank) and 0.6787 logloss on the private leaderboard (3rd place).

Solution: Code File (https://github.com/analyticsvidhya/rampagingdatahulk-2017/blob/master/Rank%203%20-%20Santanu%20Pattanayak.py)

Rank 2, Prince Atul

Prince Atul (https://www.linkedin.com/in/princeatul/) is a Senior Scientist at Cognizant. Prince has been participating in various competitions at Analytics Vidhya. Prince is also a volunteer for Analytics Vidhya and helps us with our community efforts. This is his approach:

Prince Atul

I decided to approach this hackathon with more focus on feature engineering than on model selection and data processing. After reading the problem, I decided to use gradient boosting with binary logistics.

I always submit a preliminary model, generally with all the variables, to set a benchmark score.

There were 4 moving averages in the data set and I expected them to be correlated. So, I plotted correlation matrix and as expected 10 days and 20 days moving average were highly correlated with other moving averages. I removed these two variables and trained my model on rest of the data. This model was giving a 0.68 (approx.) score on public leaderboard.

I checked for null values and there were 4000+ rows which had missing values. I left it as it because it was very small percentage of the train data set. (Wanted to come back to it, didn't get time)

After this I started creating features. Features which improved my score were (1,0,-1 values) :- comparison of 3 days moving average with other moving averages, comparison of 5 days moving average with other moving averages and sum of these comparison value. I created this to use price movement direction based on moving averages. After creating this, my model was giving a score of 0.677(approx.) on public leaderboard.

I think that hardest part in any mini-hackathon is to create features. It takes some thinking and not every feature you create will add values. But, it is important to keep on doing it even if first few features are not able to improve your model.

Solution: Code File (https://github.com/analyticsvidhya/rampagingdatahulk-2017/blob/master/Rank%202%20-%20Prince%20Atul.py)

Rank 1, Akash Gupta

Akash Gupta

Akash Gupta (https://www.linkedin.com/in/akashgupta222/) is a final year student at IIT Roorkee. Akash is one of the most competitive students we have come across on Analytics Vidhya. He fetched his last win in The Ultimate Student Hunt competition by securing 5th rank.

Find out what's his secret for winning this minihack.

Initialization: I started out by trying a basic xgboost model using the given features and filling the missing values with -1. I generally start with xgboost because of its speed and good scores. I had removed the ID and timestamp features.

(https://github.com/akashgupta222/av_datahulk#cross-validation)**Cross Validation**: To set up a quick cross validation, I randomly sampled out 10% of the dataset and set that up as the eval data. I had planned to write for timestamp-based partitioning later. But the initial eval scores for this setup were similar to the ones I got on the public leaderboard, so I persisted with this setup.

(https://github.com/akashgupta222/av_datahulk#feature-engineering)Feature Engineering

On plotting the feature importances using the default set of features, I realized that the MA features were not contributing much. Also, to me using the absolute values of these features was not intuitive. Removing these gave me an improvement in the eval score as well as the public leaderboard score. Then I removed the volume traded feature because it was also having a low contribution and removing it gave me an improvement in both eval and public lb. Later, I created 3 new features:

- difference between three day moving average and five day moving average
- difference between five day moving average and ten day moving average
- difference between positive directional movement and negative directional movement I added these features one by one and saw an improvement in both the eval and public lb scores.

I tried creating a feature for differnce between three day moving average of nth day minus the three day moving average of (n-1)th day. This gave me improvement in eval dataset, but not on the public lb. Possibily this had overfit the data, so I removed this feature.

(https://github.com/akashgupta222/av_datahulk#parameter-tuning)Parameter Tuning

(https://github.com/akashgupta222/av_datahulk#max-depth)**max depth**

I usually start with shallow trees (max depth 3). I prefer to use shallow trees because they dont tend to overfit. I tried increasing the max depth to 4 and 5, but that made the scores worse for public lb. So I stuck to using max depth 3.

(https://github.com/akashgupta222/av_datahulk#min_child_weight)**min_child_weight**

Initially, I set the min_child_weight to 1000 because of the high number of data points. Later I moved it to 1500 and 500 and saw that 500 gave me a better score. Decreasing further to 300 didnt help so I stuck with 500.

(https://github.com/akashgupta222/av_datahulk#learning-rate-num_rounds-and-early-stopping)**Learning Rate,** num_rounds **and early stopping**

I set up the early stopping parameter to 50, i.e. if the eval score doesnt improve in 50 rounds, stop training further. The learning rate was initially set to 0.05 and num rounds were initially set to 1500. But this was very slow and the score was improving even after 1500 rounds. So I changed the learning rate to 0.2 and reduced the num rounds to 800. This gave me stopping near the 600th round and quicker training as a result.

Well, thats it, I did not have the time to try ensemble models which I believe could have improved the score further.

(https://github.com/akashgupta222/av_datahulk#running-the-code)Running the code

1. Keep all the files(python script, train.csv and test.csv) in the same directory and set the working directory to that directory.
2. Run the script by command: python try1.py.
3. The submission is saved as submission_xgb.csv.

Solution: Code File (https://github.com/analyticsvidhya/rampagingdatahulk-2017/blob/master/Rank%201%20-%20Akash%20Gupta.py)

End Notes

It was great interacting with these winners and know their approach during the competition. Hopefully, you will be able to evaluate where you missed out.
Take a cue from these approaches and participate in upcoming Machine Learning Hackathon (https://datahack.analyticsvidhya.com/contest/machine-learning-hackathon/) & The QuickSolver MiniHack (https://datahack.analyticsvidhya.com/contest/avdatafest-the-quicksolver/). If you have any questions feel free to post them below.

Check out all the upcoming competitions **here (https://datahack.analyticsvidhya.com/contest/all/).**

Share this:

in (https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/?share=linkedin)

f (https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/?share=facebook)

G+ (https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/?share=google-plus-1)

(https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/?share=twitter)

(https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/?share=pocket)

(https://www.analyticsvidhya.com/blog/2017/04/winners-approach-of-rampaging-datahulk-minihack/?share=reddit)

RELATED