**E6893 Big Data Analytics Final Project**

***Earnings Predictor***

*Predicting if a stock will beat estimated earnings.*

*eps > estimated\_eps?*

*Roberto Martin*

*Kedar Patil*

Table of Contents

1. Summary 3

1.1 Motivation 3

1.2 Data Sources 3

1.3 Feature Selection 3

1.4 Training Technique 3

1.5 Results 3

2. Modeling Approach 4

2.1 Getting Data 5

2.2 Feature Extraction 5

2.3 Training and Evaluation 6

3. Data Sourcing and Feature Extraction 8

3.1 Data Sources 8

3.2 Data Extraction and Manipulation 11

3.2.1 Yahoo Finance data for Daily Quotes 11

3.2.2 Estimize API 13

3.2.3 Merging Yahoo stocks extract with Estimize 15

3.3 Feature Extraction 17

3.3.1 Price rose more than 1/2 the time 18

3.3.2 Price above 20dma more than 1/2 the time 18

3.3.3 Percentage of times price was above 10d ema above 50% 18

3.3.4 Percentage of times price went up was above 50% 18

3.3.5 Percentage of times price went up 3times above 50% 19

3.3.6 Number of times price and volume went up together 19

3.3.7 Percentage of times volume went up was over 50% 19

3.3.8 Putting them together 19

4. Model Training and Validation 21

4.1 Model Training 22

4.1.1 Logistic Regression Training and Validation 22

4.1.2 Decision Tree Training and Validation 23

4.2 Model Validation Outputs 24

4.2.1 Logistic Regression 24

4.2.2 Decision Tree 25

4.2.3 Random Forest 25

5. References 27

# 1. Summary

IGNORE THIS FOR NOW TO BE FINALIZED IN END

## 1.1 Motivation

Many analysts are paid to give estimates of earnings for different companies. The consensus earnings estimate is an average of these estimates. This consensus is correct approximately 60% of the time. There are a number of reasons why analysts incorrectly predict earnings for companies; some of the common ones being conflicting interests, personal biases, and manipulation. We aim to build a model that predicts the likelihood of a company’s actual eps beating analysts’ estimates.

## 1.2 Data Sources

The major data sources were

* Yahoo Finance daily stock data
* Zacks Investment Research on Quantdl
* Estimize – crowdsourced data on earnings estimates

## 1.3 Feature Selection

Our features are “hand” engineered features. They’re based on a company’s past prices and volume. We use Technical Analysis tools such as Momentum, moving averages and other indicators as a starting point in engineering our features.

## 1.4 Training Technique

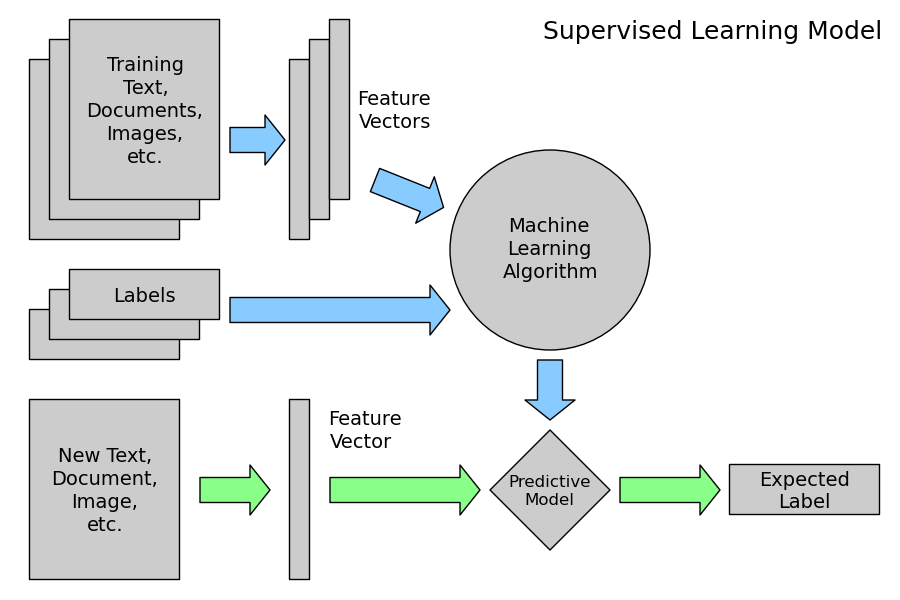
We trained the following classifiers: Decision Tree, Logistic Regression and Random Forest.

## 1.5 Results

We evaluated the results using well-known metrics such as confusion matrix and ROC curves.

# 2. Modeling Approach

The nature of this problem lends itself to the use of supervised learning techniques. Figure 3.1 below shows the general steps involved, and is representative of our approach in this project.



*Source: www.astroml.org*

Fig 2.1: Supervised Learning Model

The process essentially involves following steps:

* Data Collection
* Feature extraction
* Training and Evaluation
* Predicting on new datasets

## 2.1 Data Collection

The process involves identifying relevant data sources, and taking steps to access data from these sources. This could range from a simple flat file download to accessing through an API key or web scraping.

In our case, the data was mainly acquired from following sources –

* Yahoo Finance daily stock data
* Zacks Investment Research on Quantdl
* Estimize – crowdsourced data on earnings estimates

The detail code and extraction technique is documented in section 3.

## 2.2 Feature Extraction

This involves manipulating and munging extracted datasets to create features that would be relevant to solving the given problem at hand. Stock and earnings data that we acquired are time series data. While the OHLCV (Open-High-Low-Close-Volume) stock data from Yahoo is at a daily cadence, actual earnings and analyst estimates data from Zacks and Estimize are at a quarterly cadence.

The aggregation involved *Split-Apply-Combine* idiom, where daily data was split into quarterly intervals, aggregation functions were applied repeatedly to all quarters, and the data was finally combined to create features corresponding to earnings data for each quarter. Target labels were derived from a combination of actual earnings from Zacks on Quantdl, and analyst consensus estimates procured from Estimize. Finally both these parts were combined into one unified dataset that was ready for training modeling algorithm.

## 2.3 Training and Evaluation

We considered different algorithms such as Decision Tree, Random Forrest and Logistic Regression for following reasons

*Logistic Regressions* scales very well to large datasets. They are also less prone to over-fitting as compared to decision trees. However, they are more suited for modeling data with single decision boundary. We feel that logistic regression is a good baseline model to use.

*Decision Trees* work well with non-linear data and don’t require normalization. They are also easy to interpret. However, Decision Trees tend to overfit data, which is why we choose to ensemble them in a Random Forrest model.

*Random Forests* corrects for the possible overfitting in Decision Trees. Ensembles are also known to do better than the actual learning algorithm.

Model evaluation involves monitoring following metrics:

***Accuracy***looks to answer the question – how often do we make the right decision. In other words, does model correctly classify 1’s when actual value is 1, and 0’s when actual value is 0. This is captured in the formula : . This metric is however sensitive to imbalanced classes (cases when number of 1’s is greater than 0’s, as is the case here since companies more often beat consensus estimates based on the way we calculate a beat) .

***Precision*** looks at how often our model has false positives. This is represented by the formula: . Higher value in this case is better, since it means that we have very little actual 0’s falsely classified as 1’s.

***Recall*** looks at how sensitive our model is. It seeks to answer the question – how many of the actual 1’s did we correctly identify.

This is represented by the formula: . Here too, higher number is better, as it means that we have fewer actual 0’s classified as 1’s.

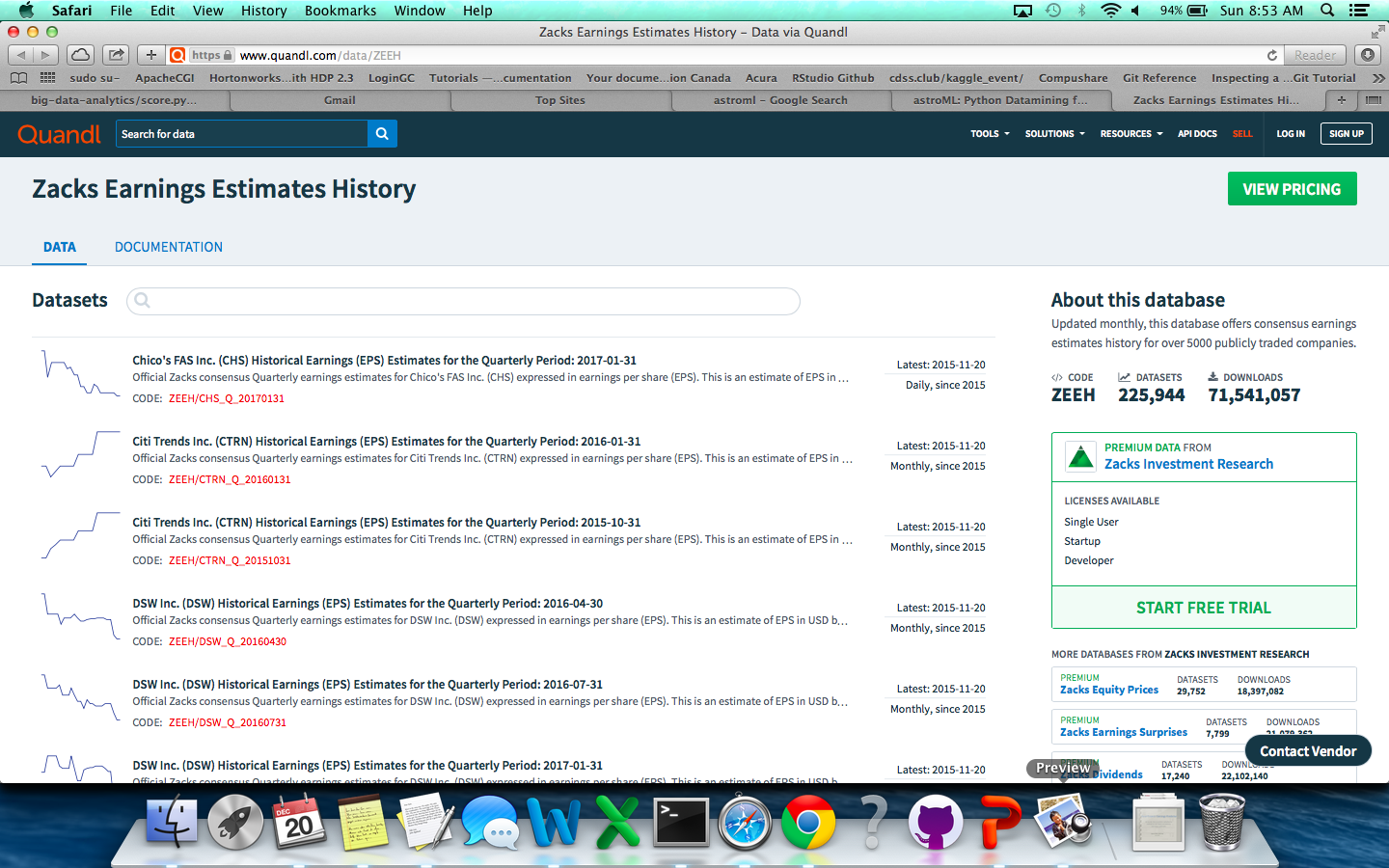
# 3. Data Sourcing and Feature Extraction

As discussed in section 2, this stage involved identifying and extracting relevant datasets, followed by manipulating and munging them to create features that would be relevant to predicting whether or not company would beat forecasted earnings.

## 3.1 Data Sources

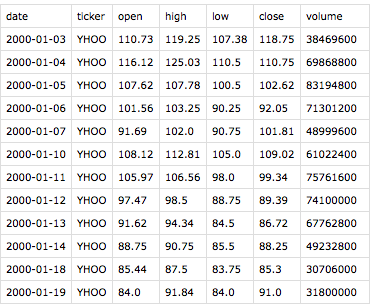
Stock and earnings data that we extracted was mainly time series in nature. While the OHLCV (Open-High-Low-Close-Volume) stock data from Yahoo was at a daily cadence, actual earnings and analyst estimates data from Zacks and Estimize respectively, was at a quarterly cadence.

Consensus estimates data was toughest to find. Most datasets for this are available for a fee through aggregators. Quantdl and Zacks Investment Research were kind enough to give us free access to their dataset. Normally, cheapest license for this would have cost $1800/year. Quandl is a website/search engine all kinds of data, including financial. Zacks is an investment research company that also provides a lot of data, some of it through Quandl. The data we were given access to can be found at: <https://www.quandl.com/data/ZEEH>.



*Fig 3.1: Zacks Investment Research data on Quandtl*

*Yahoo Finance data for Daily Quotes* includes OHLCV (Open, High, Low, Close, Volume) daily data history on stock prices. We pulled data going back to Jan 3, 2000.



*Fig 3.2: OLHCV data from Yahoo Finance data for Daily Quotes*

Estimize is an open financial estimates platform which facilitates the aggregation of fundamental estimates from independent, buy-side, and sell-side analysts, along with those of private investors and students.

**

*Fig 3.3: Estimize financial estimates*

## 3.2 Data Extraction and Manipulation

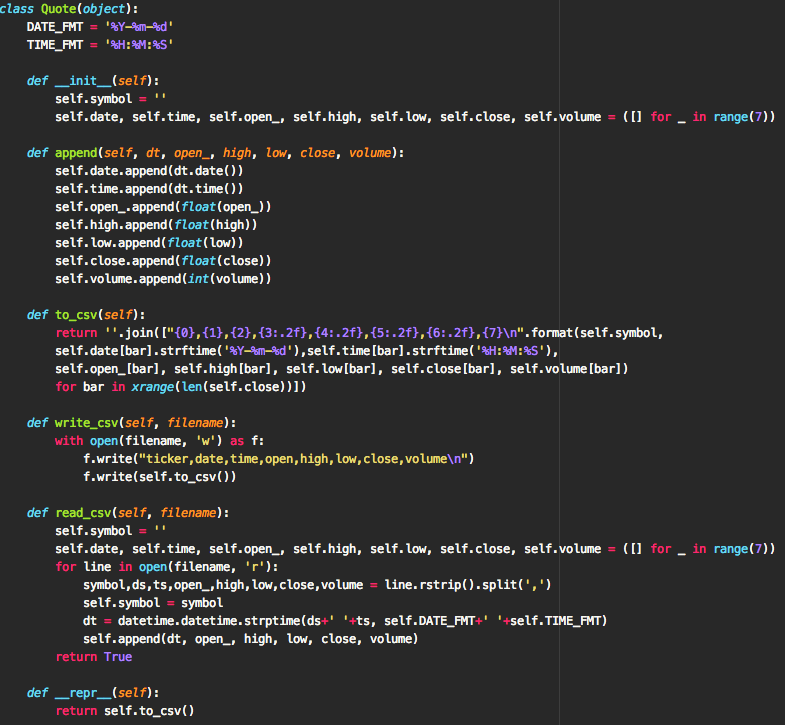
We used Python (v2.x) for acquiring and manipulating relevant data. Packages such as Pandas (for dataframes, as well as groupings and aggregations), Numpy, SciPy, Matplotlib were used for etl and other manipulations and visualizations.

### 3.2.1 Yahoo Finance data for Daily Quotes

This code has 2 parts - *yahoo\_quote.py* uses *quote.py*.

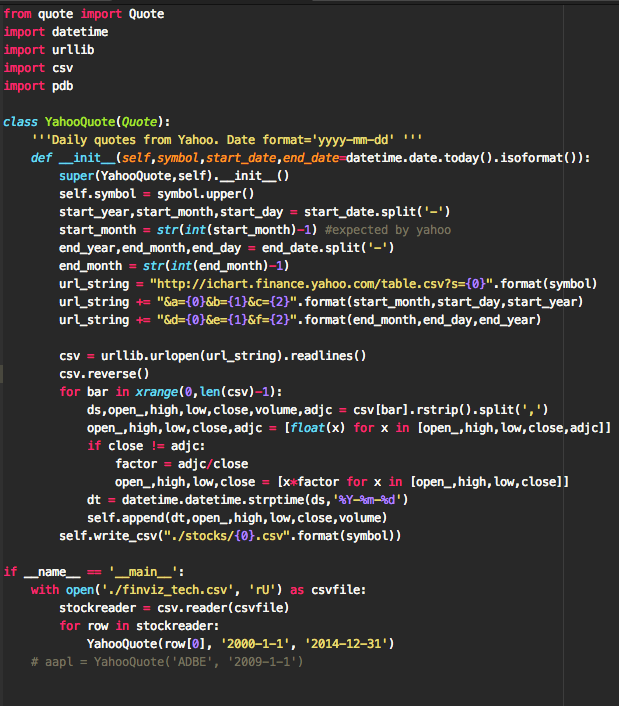
*Create Class Object - Quote*

The following is a sampling of the Python code used to download stock market data.



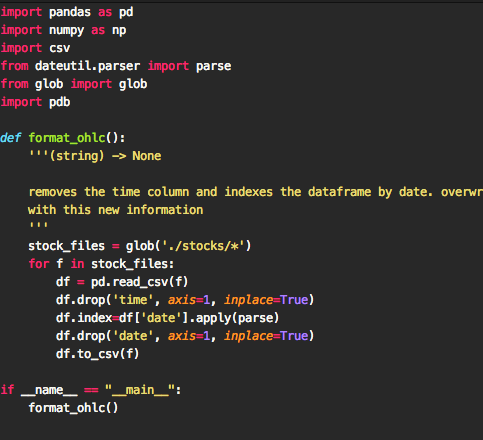
*Extract daily quotes from Yahoo*

This part uses class object "Quote", and helps extract OLHC and volume data for a given symbol from a user defined start date to current date.



*Format Yahoo stocks extract*

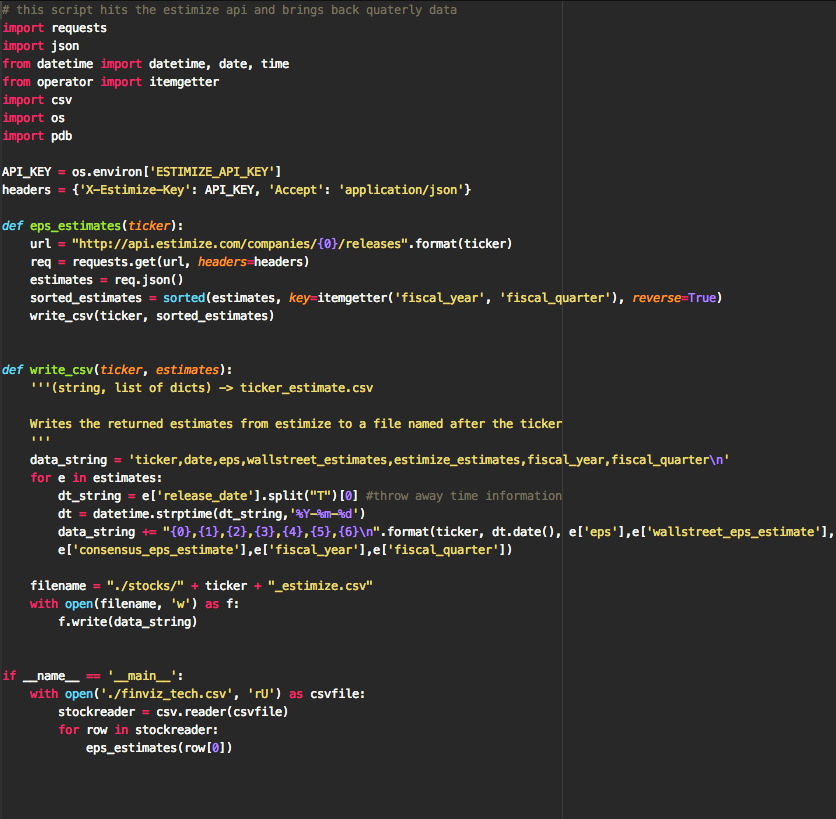
This script removes the time column and indexes the dataframe by date.



### 

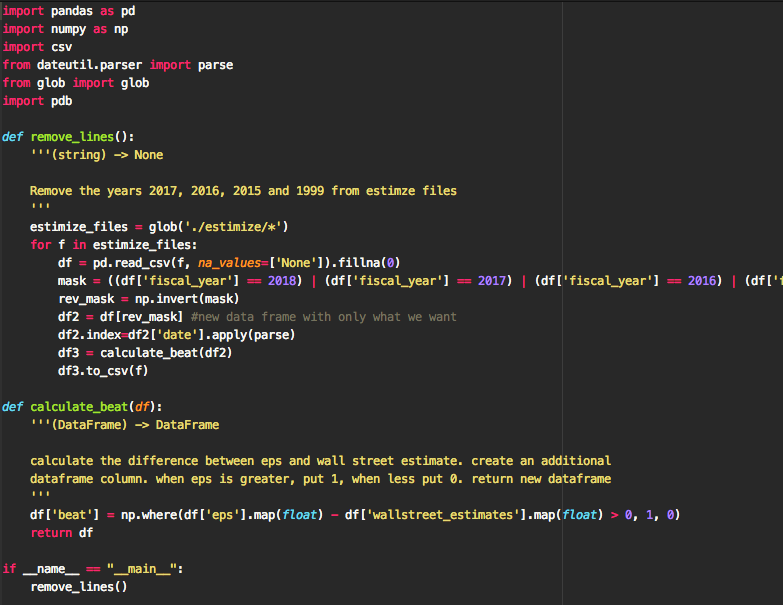
### 3.2.2 Estimize API

Fiscal year and fiscal quarter data from Estimize was also added. This helps us manage the fact that companies report at different times of the year, and determining quarter boundaries keeps us from overshooting. For example, while Google (GOOG) and Apple (AAPL) are have similar fiscal years, Adobe (ADBE) is quite different. This script hits the estimize api and returns quaterly earnings estimates. We also used stock ticker list from finviz data (<http://finviz.com> )



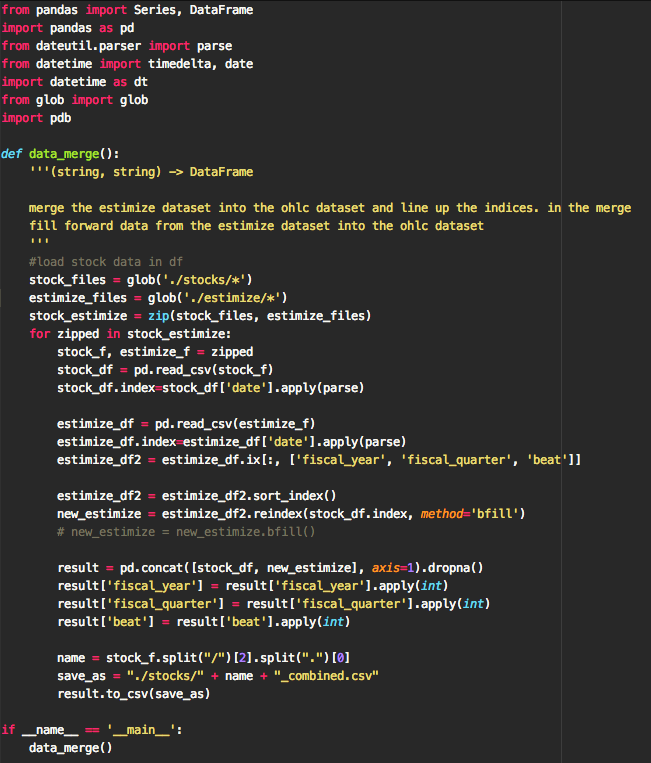
*Format Estimize extract*

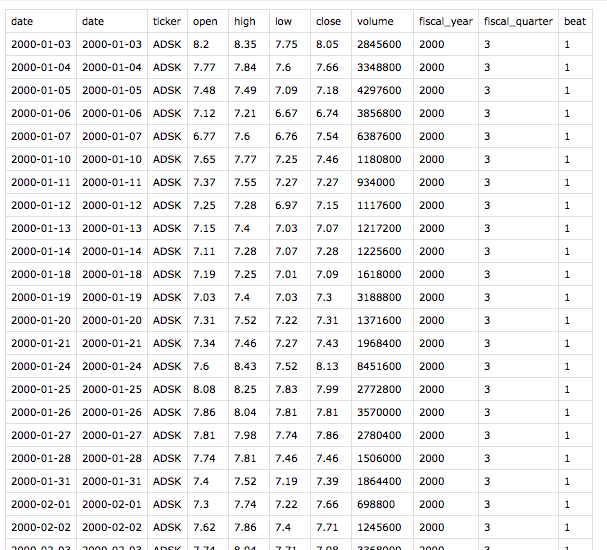
This script removes years 2017, 2016, 2015 and 1999 from Estimize files and creates target variable by comparing EPS with Wall Street estimates.



### 3.2.3 Merging Yahoo stocks extract with Estimize

This portion merges the two extracts to create a starting dataset.

An example of the combined file, including stock and quarterly data and the target variable, **beat**.



3.3 Feature Extraction

As discussed in section 2, data aggregation involved the *Split-Apply-Combine* idiom, where daily data was grouped into quarterly intervals, aggregation functions were applied to all quarters, and the data was finally combined to create features corresponding to earnings data for each quarter.

Stock data was relatively clean. However, there were few issues that we need to consider

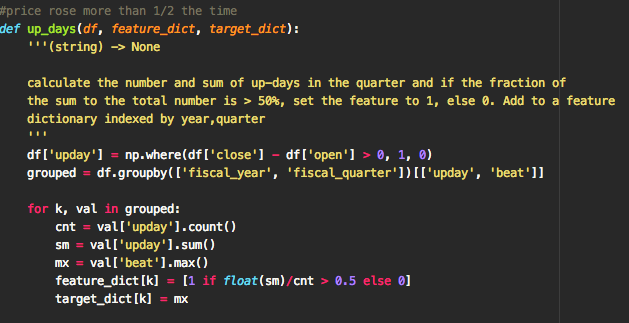
* We had to adjust for Splits/Reverse Splits
* Re-index earnings estimate timelines to match OHLC Data
* Empty estimates were removed
* Estimates were filled to enable aggregation

While selecting features, we had to bear few things in mind. The challenge was to select features that were not only predictive of the desired target (1:beat estimates/0: did not beat estimates), but also that they made logical senseto a human analyst. This is just one of the many ways for selecting features, based on domain knowledge. However, we think it’s worth exploring other possibilities in the future.

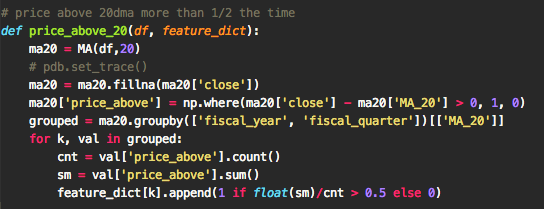
A few of our features came directly from the raw data, however most of our features are calculated from Technical Analysis indicators, engineered so to speak. We used price and volume dataas well as their various moving averages to generate features. The process of generating these features involved grouping prices first by year, and then on a quarter, and finally applied aggregation functions.

A few examples of the features we generated follows.

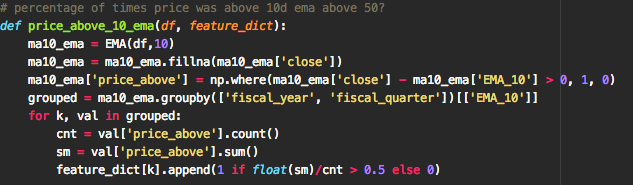
### 3.3.1 Price rose more than 1/2 the time

**

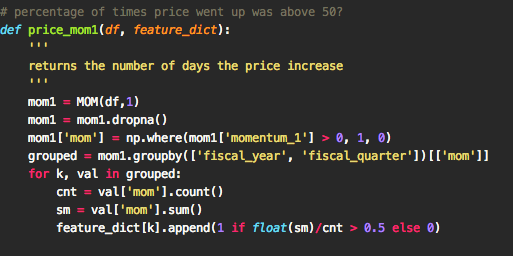
### 3.3.2 Price above 20dma more than 1/2 the time

**

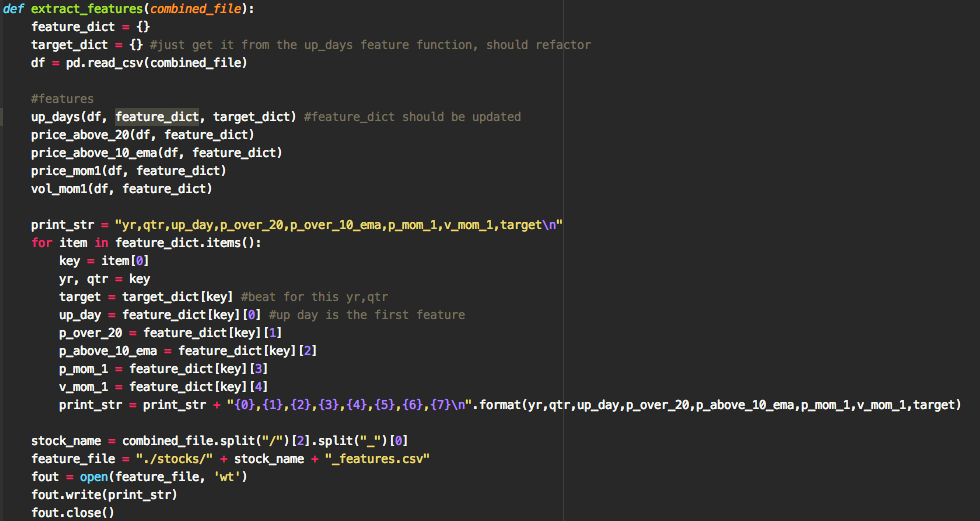
### 3.3.3 Percentage of times price was above 10d ema above 50%

**

### 3.3.4 Percentage of times price went up was above 50%

**

### 3.3.8 Putting them together

****

As can be seen from the code snippets above, we’re guessing at features of price action over the quarter that will most accurately predict whether this stock will beat earnings or not. This however is not ideal. As well, we’re only using technical features, solely based on the stock price and volume; however, we could have considered fundamental features as well. We also think there are other predictive technical indicators that would help, but were not implemented.

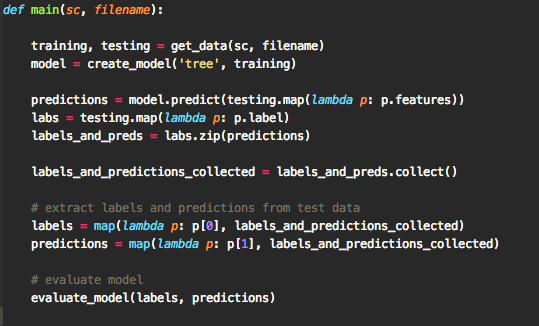
# 4. Model Training and Validation

As discussed earlier in section 2, we experimented with the following algorithms: Decision Trees, Random Forrest and Logistic Regression for following reasons.

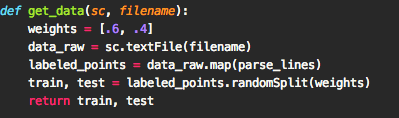
## 4.1 Model Training

We used Spark’s MLlib library for training models as shown below. In the screenshots that follow, we present the code that was used to train and evaluate the models.

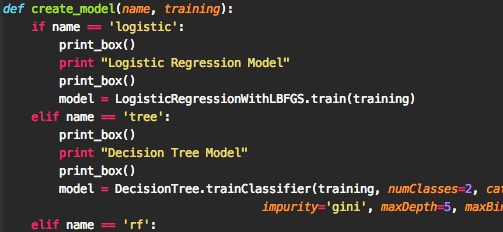
Main Runner file



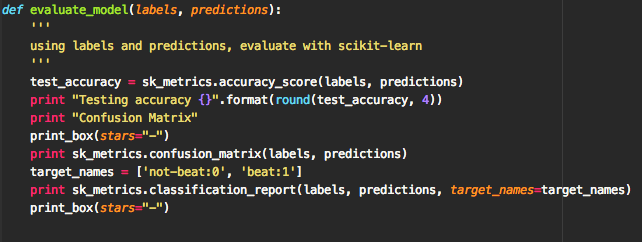
Preparing the data



Creating the models



Running the evaluations

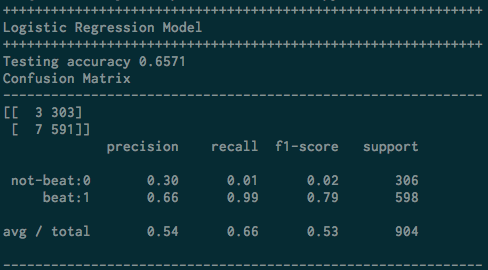


## 4.2 Model Validation Outputs

Outputs from previous section are discussed here

### 4.2.1 Logistic Regression

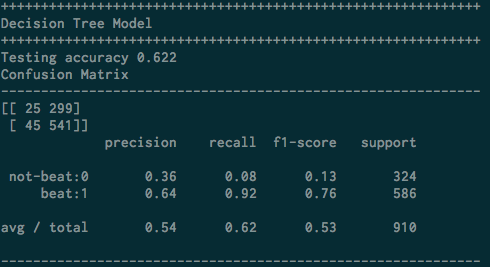
Output from Logistic Regression piece is shown below in fig 4.1 below.



*Fig 4.1: Logistic Regression Output*

### 4.2.2 Decision Tree

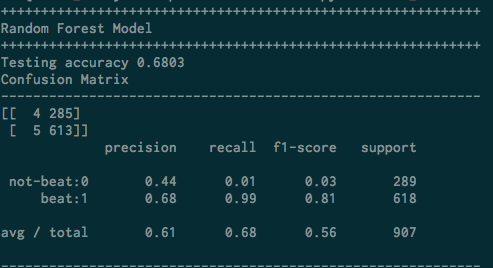
Output from Decision Tree piece is shown below in fig 4.2 below.



*Fig 4.2: Decision Tree Output*

### 4.2.3 Random Forest

Output from Random Forest piece is shown below in fig 4.3 below.



*Fig 4.3: Random Forest Output*

# Discussion

We can see that for this dataset and run (multiple runs), the models all gave similar metrics. While Random Forest gave the best precision and recall overall, it is not stellar. These results are not ground-breaking, but they do predict with a better than random accuracy.

So far, we’ve demonstrated how to answer the question initially posited: Can we build a model to predict whether earnings will beat estimates? We have shown a process by which such a model could be built and evaluated. What remains now is for us to further develop our model by getting more data and engineering more features as well as exploiting parameter tuning and cross validation to build better quality models.

# Sources

* Kaggle document template - <https://www.kaggle.com/wiki/WinningModelDocumentationTemplate>
* Yahoo Finance daily stock data - <http://finance.yahoo.com>
* Zacks Investment Research on Quantdl -<https://www.quandl.com/data/ZEEH>
* Estimize – crowdsourced data on earnings estimates - <https://www.estimize.com>
* Principals of Data Mining – David Hand, Heikki Mannila, Padhraic Smyth
* Decision tree image - <http://www.scielo.org.za/img/revistas/wsa/v35n5/a20fig03.gif>
* Finviz.com – <http://finviz.com>