Stock Prediction

- Tanmay Thakar
tthakar@umassd.edu

"Google's self-driving cars and robots get a lot of press, but the company's real future is in machine learning, the technology that enables computers to get smarter and more personal."

– Eric Schmidt (Google Chairman)

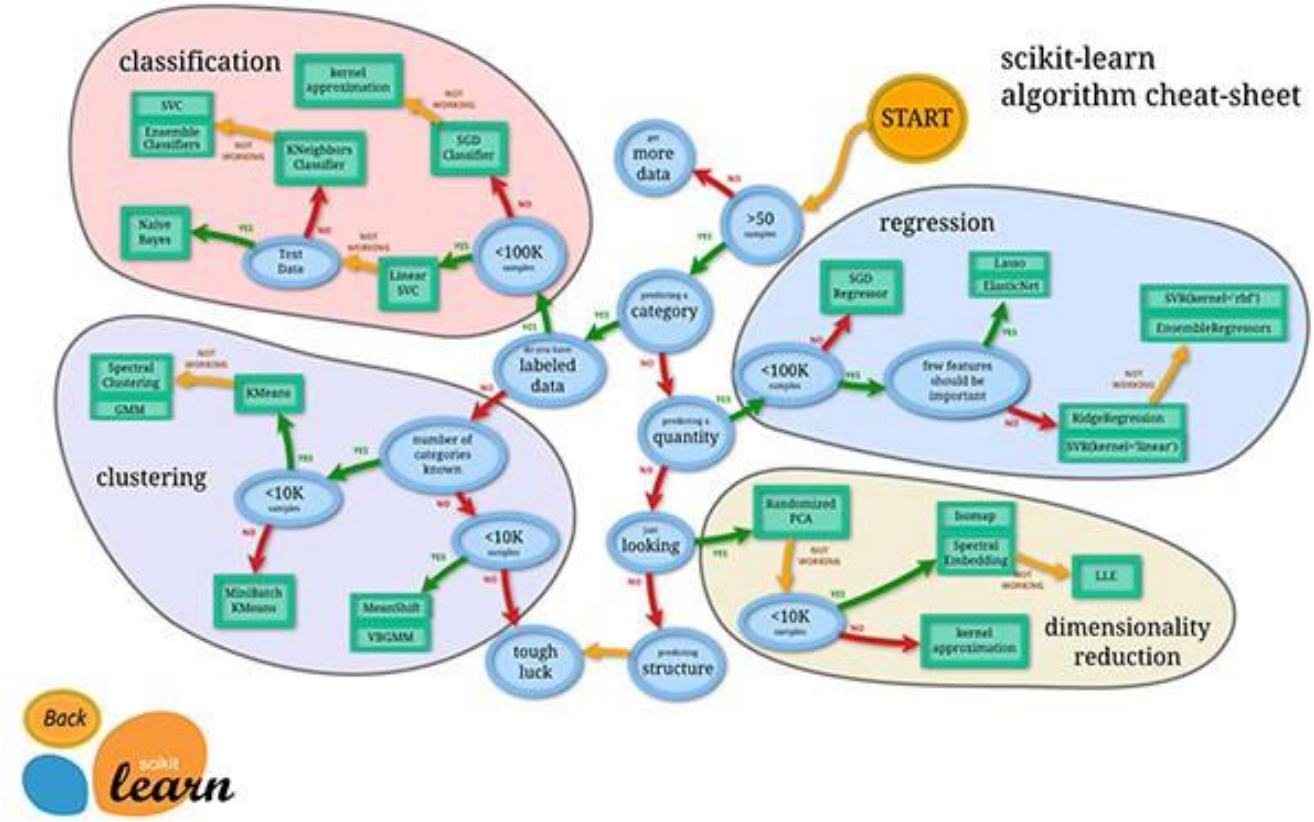
ABSTRACT

In this project, I would like to predict the stock rate for the intraday and next two day. For this I use the data set from The Winston Stock Market Challenge from kaggle which held competition for data science and machine learning. I used Python for this data analysis.

INTRODUCTION

Stock Market has always been interesting topic for researcher and for me it's interesting because you can earn money from it if you can implement good automated trading system. Machine learning, In Simple language you have to train your model by Past data set for it and you can predict the future value. There's different Machine learning library available for Python like Scikit-learn, Pybrain, PyML, MILK etc. From that I found Scikit-learn is effective and simple library for Machine learning, Statistic analysis , Feature extraction, Preprocessing Data , Cross validation For model selection etc.. Basically there are two type of Machine learning algorithm Supervise learning algorithm in which you have target value which is to be predicted from given set of features. And other one is unsupervised algorithm in which you don't have target value to predict its use for clustering. You can find out which algorithm you have to implement from the algorithm cheat sheet.

There are two type of Models in Supervised learning Regression and Classification. So As per Algorithm cheat-sheet we have quantity data so we need Regression. There are lots of Regression algorithm implemented in Scikit-learn like. We used three different model for prediction Linear Regression, Decision Tree, SVM.



OVERVIEW

In my Training Dataset there are 25 unknown features which may be affect the return values of stock. And they provide return for the given stock instead of price, Return is the financial term in Stock market. They don't provide any equation for that or how they calculate the return basic way to find out the return is

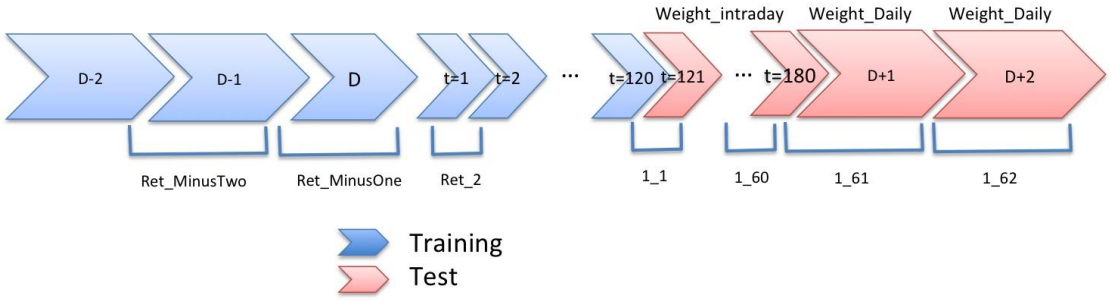
$$\text{Return} = P_2 - P_1 / P_1$$

Where P1 is the price at the end of period 1, and P2 is the price at the end of period 2. There are more complicated ways which take other things into account such as dividends paid out during the period.

They Provide the Return value for the five day window Day-2 ,Day -1 , and for the given day they provide the intraday value for four hour from ret_2 minute to ret_180 minute and Day+1,Day+2.

For Example:

Ret_MinusTwo: this is the return from the close of trading on day D-2 to the close of trading on day D-1 .



So the return for the intraday start with ret_2 because its return for the first minute.

And they also provide the Weight for the Intraday and Daily Prediction which we can use when we are train our machine. As shown in figure in test Dataset we have the return which are in blue color and we have to predict the return which are in red color.

First of all I load all the csv file (train.csv,test.csv) into data frame Using panda's library and find that there are Na or NaN value for some of the futures. So I try to fill them with different strategy but I think it's good to replace it with the 0. Because in Financial data and if we don't know what is the feature and how it relate to the stock so false value for that feature can create false prediction. Because in the Regression data quality is more important for the best prediction.

The other important thing for better prediction is feature selection for that we can use some of the methods from the same library like

```
sel = VarianceThreshold(threshold=(.8 * (1 - .8)))
```

It will remove the feature that are either 0 or 1 for 80% of data.

For feature selection first of all I have to find out which type of relation have with the return value for that I have plot that to find out

```
In [19]: #train_df.ix[:,1:26].hist()
         #plt.show()
```

As shown in figure I can find out that feature 16 have same value for all dataset so it's not effective for the change in Return value. I find out this type of assumption that help me to select the better model.

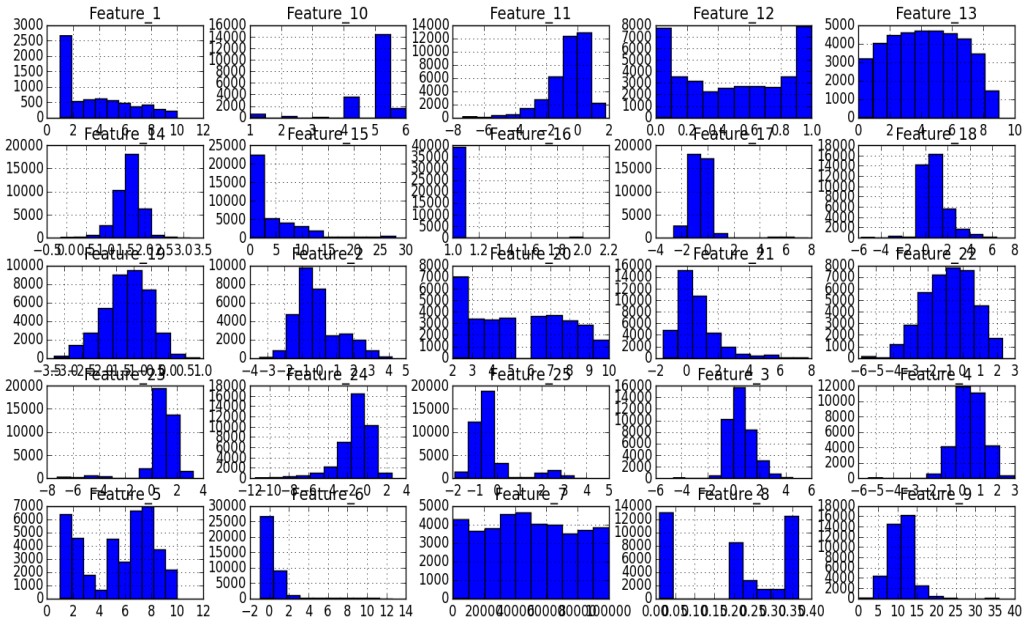


Figure Shows the features value For all the data using hist().

After that I plot the scatter plot for the all the feature vs all the return value turn by turn to identify the feature that have more effect on return value .

After that I used different model for supervise learning from that library like LinearRegressin(),DecisionTreeRegressor(),Svc() to train with the data.

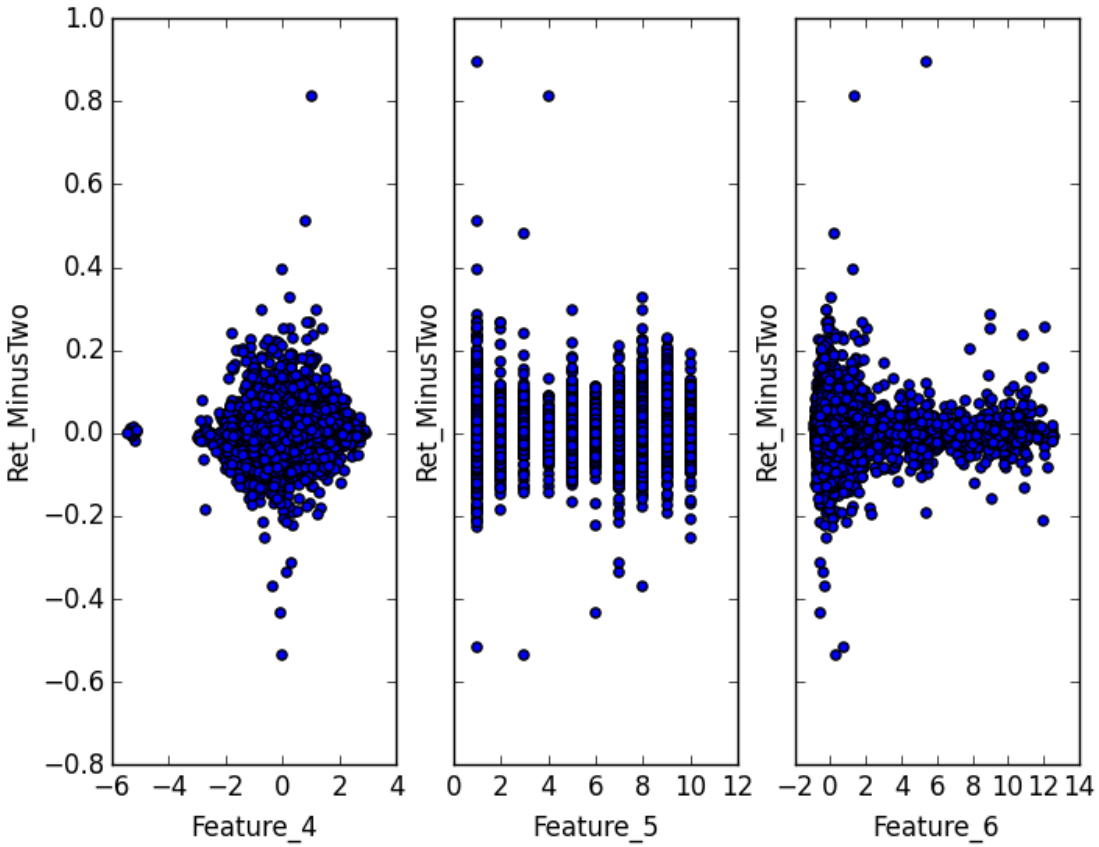
```
models = [("LR", LinearRegression()),("DT", DecisionTreeRegressor()),("SVM",SVC(C=1000000.0, cache_size=200,
            coef0=0.0, degree=3, gamma=0.0001, kernel='rbf',
            max_iter=-1, probability=False, random_state=None,
            shrinking=True, tol=0.001, verbose=False))]

# Iterate through the models
for m in models:

    # Train each of the models on the training set
    m[1].fit(X_train, y_train,sample_weight=np.asarray(train_df['Weight_Daily'][train_df['Weight_Daily'].index < 20000]))
```

This can be formalized by considering a response Y with p different features x1, x2,...,xp. If we utilize vector notation then we can define X = (x1, x2,..., xp), which is a vector of length p. Then the model of our relationship is given by:

$$Y = f(X) + \varepsilon$$

Where f is an unknown function of the predictors and Ɛ represents error or noise term. Importantly, Ɛ is not dependent on the predictors and has a mean of zero. This term is included to represent information that is not considered within f. Thus we can return to the stock market

index example to say that Y represents the value of the Return whereas the xi components represent the values of feature and train data.



Scatter Plot for Feature vs Return value

Another important thing is finding the series is mean reverting or not. This process refers to a time series that display the tendency to revert to the historic mean value. Which one also useful to generate trading strategy for profit.

```
def hurst(ts):
    """Returns the Hurst Exponent of the time series vector ts"""
    # Create the range of lag values
    lags = range(2, 100)
    # Calculate the array of the variances of the lagged differences
    tau = [sqrt(std(subtract(ts[lag:], ts[:-lag]))) for lag in lags]
    # Use a linear fit to estimate the Hurst Exponent
    poly = polyfit(log(lags), log(tau), 1)

    # Return the Hurst exponent from the polyfit output
    return poly[0]*2.0
```

- H < 0.5 - The time series is mean reverting
- H = 0.5 - The time series is a Geometric Brownian Motion
- H > 0.5 - The time series is trending

RESULT

Measuring the Forecasting accuracy is important to select the best fitted model for prediction. The simplest question that we could ask of our supervised classifier is "How many times did we predict the correct direction, as a percentage of all predictions?". Given by this formula

$$\frac{1}{n}\sum_{j=1}^{n} I(y_j = \hat{y_j})$$

And the Scikit-learn provide the **score(x,y)** and **cross_validation.cross_val_score(m[1], X, y, cv=5), (accuracy_score(y_test, y_pred)** method to find the best model .

This are the output for my dataset.

(1)

```
LR:
-0.002
DT:
-0.756
```

(2)

```
LR:
-0.002
[ 0.00108204 -0.00182361 -0.01184265 -0.00080929 -0.00113704]
```

(3)

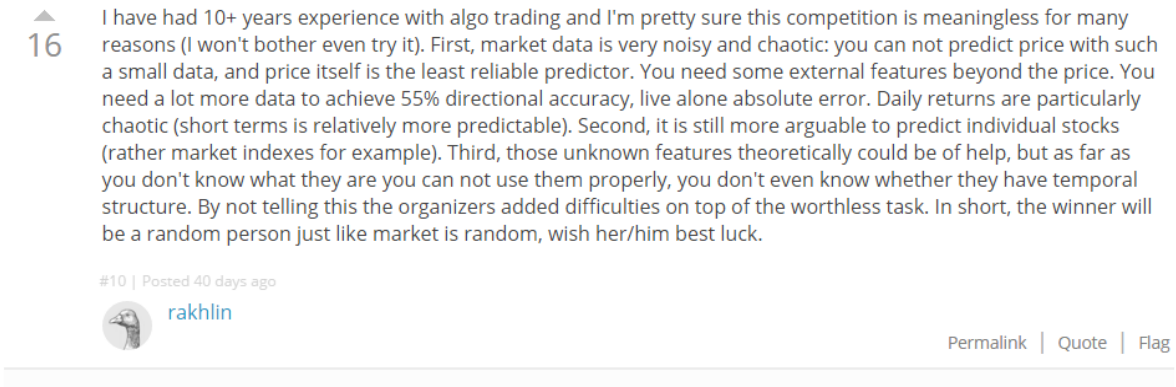
```
predisction of Ret_PlusOne using
LR
[ -4.95350301e-04   5.16403982e-04  -6.48728392e-04 ...,  -1.21007118e-03
  -1.64048177e-04  -4.85274816e-05]
predisction of Ret_PlusOne using
DT
[ 0.00052721 -0.0204129   0.00306271 ..., -0.08560744 -0.00188966
 -0.00647517]
```

(4)

```
prediction of Ret_121 using
LR
[ -7.43911802e-05   3.61519038e-04   5.93122651e-05 ...,   4.13846344e-04
  -5.12912987e-05   1.26555901e-04]
prediction of Ret_121 using
DT
[  1.20674804e-05   8.80349121e-05  -2.17605902e-03 ...,  -2.44820627e-04
  -1.15782529e-05   7.11973864e-05]
```

CONCLUSION

From the above result I can find out the Linear Regression give the accurate result compare to other. But this dataset are noisy and unpredictable and they don't provide us the feature related information so there's lots of information we need for better prediction for financial stock data. Because data cleaning for financial data is painstaking process so data quality from the vendor and continuity of data is important. and in stock market to label the data means the stock going to "UP" "DOWN" is effective compare to predict the return .



FUTURE WORK

For future work I want to implement algorithmic trading using the time series analysis (TSA) using stats models library for python. There are two broad families of time series which are useful for algorithmic trading like auto regressive integrated moving average (ARIMA) and autoregressive conditional heteroscedasticity (ARCH).

REFRENCES

[1] https://www.kaggle.com/c/the-winton-stock-market-challenge

[2] SUCESSFUL ALGORITHMIC TREDING Applying The Scientific Method For Profitable Trading Result By Michel L. Halls-Moore

[3] http://www.analyticsvidhya.com/blog/2015/08/common-machine-learning-algorithms/

[4] http://murphy.wot.eecs.northwestern.edu/~pzu918/EECS349/final_dZuo_tDing_vFang.pdf

[5] http://datascience.stackexchange.com/