Predicting Volatility in Equity Markets Using Macroeconomic News

By : Hadeel Sameh Hassan

Introduction

- These market movements are not uncommon.
- Macroeconomic sentiment immediately impacts the volatility in markets (Ex:Greek crisis).
- The input of system will be the word count of each bucket of tweets in a dictionary we created. We then use prediction methods to estimate the increase in the VIX.

DATASET and Preprocessing

- Our data is **tweets Headlines** from twitter API.
- Firstly, removing **stop words** like “the” , “a”, “in”, “at”…… will be useful to reduce the amount of meaningless words and do **tokenization** (**NLTK** library in Python).
- Secondly, making word count for all words then start to rank words to get **most popular words** .
- Finally use our dictionary for **positive and negative words** to categories tweets in to positive ,negative ,and neutral.

Features

- Merging the 2 types of data (financial ,twitter) based on **date and company name**, we get a dataset where each row contained company information and their performance (**Pandas** library for merging data-frames).
- I will **divide** **NUM_NEG, NUM_NEU, NUM_POS** to get **percentages** then drop **TW** column(the column of total number of tweets).
- After pre-processing the data, the dataset contains a total of 8 features with a split of 70% train , 15% validation and 15% test data.

Model Implementation

- **Naive Bayes :**Firstly , we applied Naive Bayes. The key assumption of this model is the conditional independence of the features. The basic role is: $P(x_i,x_j|y = 1) = P(x_i|y = 1\ x_j)$.
- **Support Vector Machines :** we employed the support vector machine method , which performs linear regression in the high dimension feature to maximize the functional margin. The loss function of Linear SVM is:

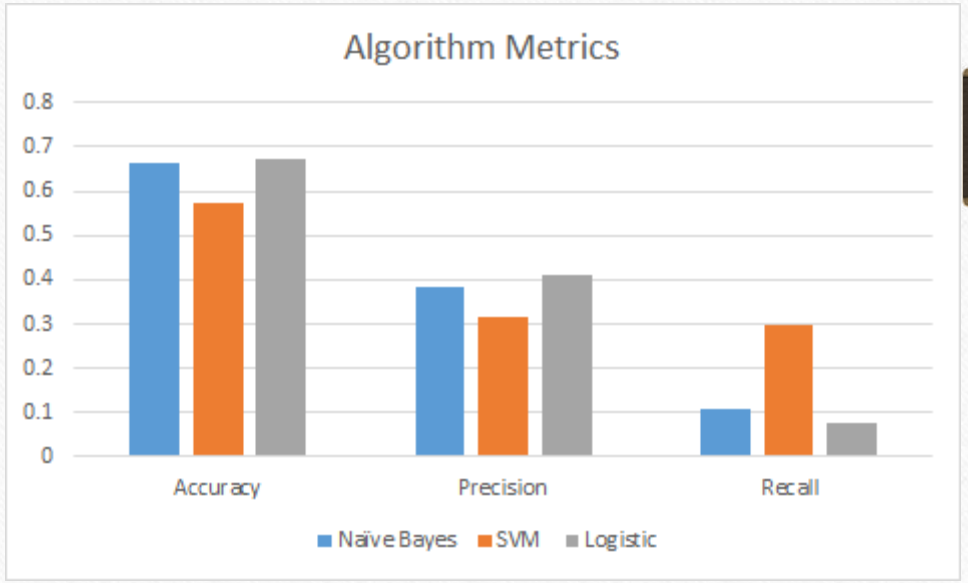
$$\begin{aligned} \underset{w}{\text{minimize}} \quad & \frac{1}{2}||w|| \\ \text{subject to} \quad & y^{(i)}(\langle w, x^{(i)}\rangle + b) \geq 1, i = 1, \cdots, m \end{aligned}$$

$$\underset{\alpha}{\text{maximize}} \quad W(\alpha) = \sum_{i=1}^{m} \alpha_i - \sum_{i,j=1}^{m} y^{(i)} y^{(j)} \alpha_i \alpha_j K(x^{(i)}, x^{(j)})$$

- **Logistic Regression with PCA :**we performed logistic regression after performing PCA to reduce dimension of data. Logistic regression makes the least amount of assumptions on our dataset. Logistic regression is a model that measures the relationship between a categorical binary dependent variable, which we have taken to be whether or not the VIX has increased by a certain amount, and the independent variables. It estimates the probabilities of the categorical variable using the logistic function. Our hypothesis has the form is **Sigmoid** function.

Results (supposed to be achieved)

- we see that **Logistic Regression** has the best accuracy and precision at 67% and 41% respectively, with Naive Bayes trailing with 66% and 38% for accuracy and precision.
- Although SVM did much better than both Naive Bayes and Logistic Regression in Recal ,SVM is considered to be the worst performing model for our purposes. This is because our trading strategy will only enter into a position in the market if we predict a positive result from the data "**the most important metric for us is precision**" ,
- The SVM algorithm performed much **worse** because of the noise in our data, and small number of data points.



Deployment

- In order to deploy our product we will create web application using python ,Flask for Backend ,HTML,CSS,Bootstrap for our model.
- Finally deploying our app on Heruko .