Project Final Report

Rajiv Mantena (u1007484), **Kameswari Devi Ayyagari** (u1010635)
Team name : **ColdPizza&Coffee**
CS 6350 Machine Learning
University of Utah, Fall '16
December 15, 2016

Problem Statement

Predicting an airline's stock price based on on-time performance and twitter sentiment analysis.

We are trying to understand if there is a hidden trend on how stock prices of an airline company, **American Airlines**, are affected by it's on-time performance and customer satisfaction.

Suppose, on a particular day, if majority of flights of a particular airline company are cancelled/delayed (as it had occurred in case of delta airlines recently, link), it can be intuitively understood that this would affect the company's stock prices. A drop in the stock prices could be anticipated.

However, this trend in stock prices may not be effectively predicted on regular days. We were intrigued by this idea as there might be a hidden trend in nature which may not be intuitive to understand, but Machine Learning ideas could be used in this scenario to learn a close approximation of this hidden trend.

Strategy

To solve this problem, we had to come up with the following strategies :

1. Obtaining data to quantify customer satisfaction and stock trends
2. Feature extraction
3. Forming assumptions in relevance to the dataset
4. Performing experiments on the dataset using ML algorithms

Obtaining data

We intended to quantify customer satisfaction by two methods:

- **On-time performance data :** On-time performance data of an airline would give us the the number of cancelled/delayed flights on a particular date. This data (in a very crude form) was already available from the US Department of Transportation.

We considered the time-frame between August 2015 and August 2016 for collecting the required data. The data provided by the US Department of Transportation consisted of performance statistics of over 1 million flights operated by **American Airlines** in the mentioned time frame.

Based on this data, we have consolidated the following metrics for each day : Number of *Cancelled* and *Diverted* flights; Percentage of flights with *Departure Delay* greater than 15 mins, 45 mins & 90 mins; Percentage of flights with *Arrival Delay* greater than 15 mins, 45 mins & 90 mins; Total number of *Carrier Delay* minutes, *National Airspace System (NAS) Delay* minutes, *Weather Delay* minutes, *Late Aircraft Delay* minutes.

- **Twitter data :** We planed to fetch all tweets relevant to a particular airline on a particular day and run a customer sentiment analysis on the tweets. This would provide us a customer satisfaction ratio for a given day.

  We intended to use Twitter's APIs to obtain this data. However, we weren't able to successfully obtain this data, as Twitter's API would only provide data from Tweets within 7 days. After investing considerable time into data extraction from Twitter, we realized that Twitter doesn't allow any way of extracting historic data for *free*. **Gnip** is a paid Twitter API to extract historical data.

  The only other way which might have worked was by using Web-Scrapping techniques. However, this was very computationally expensive and very time-consuming. Due to time constraints, we could not extract data using this method.

  After these futile efforts, we gave up on the idea of quantifying customer satisfaction based on Twitter data.

We plan to use the obtained data, in addition to the stock price history for the past year, to predict if the stock price of American Airlines would increase/decrease on the following day. Please find the details of the dataset which have been formed as mentioned above.

- Number of features : 147 binary features
- Number of records :

Training	366 records	August '15 to July '16
Test	31 records	August '16

- DataSet Baseline :

Positive	52.296%	*Increase in Stock Price*
Negative	47.703%	*Decrease in Stock Price*

Please note that the Test set is not a sub-set of the Training set. We have considered the data from August '15 to July '16 as the Training set and the data from August '16 as the Test set.
We intended to do it in this manner to keep the data as realistic as possible, as it is in this fashion that these class of algorithms would have to make predictions in real-life.

Feature extraction

The above specified metrics from the **on-time performance data** have been used as the features for our dataset. However, these features are **real-valued** and they have to

be featurized into booleans. This is required as several ML algorithms which we intend to use required boolean features. Therefore, we have converted these **real-valued** metrics into **binary features** by breaking each metric into sufficient number of ranges and using $1's$ & $0's$ to represent each range wherever necessary.

Assumptions

In order to link the *on-time performance* dataset and *stock trends*, we have made the following assumptions

- Today's closing stock price of the airline depends on yesterday's on-time performance.
- Increase/decrease in stock price is calculated as the change in the closing stock prices of of two consecutive weekdays.
- As stocks are updated only on Weekdays, for the certain experiments, we have assumed that Monday's stock price variation depends on Friday's on-time performance. This was later optimized to consider the performance of Friday, Saturday and Sunday.

Experimentation

We were able to implement several Machine Learning algorithms on the obtained Dataset. The algorithms include *Perceptron*, *Winnow*, *Support vector Machines*, *Logistic Regression Classifiers*, *Random Forests* and *k Nearest Neighbors*.

Please be reminded the positive base-line accuracy was 52%. These algorithms produced accuracies varying from under 50% upto 75%.

1. **Perceptron**

   We were first able to implement Perceptron on the obtained binary-feature dataset. After performing experiments on different variants like *Vanilla Perceptron*, *Margin Perceptron* and *Average Perceptron* (all with simple and aggressive update policies), our accuracy was just a little greater than that of base-line of the dataset. It was even under 50% for certain variants.

2. **Winnow**

   After not finding any significant increase in accuracy by Perceptron, we have implemented variants of Winnow : *Vanilla Winnow* and *Balanced Winnow*. We have found that after slightly tweaking the Vanilla Winnow, we were able to achieve 65% accuracy. This is almost 15% increase in accuracy over the base-line.

   At this point we were optimistic that Batch Algorithms like SVMs would be able to achieve much better accuracies.

3. **Support Vector Machines**

   We have now implemented *Support Vector Machines* on the dataset and the results were puzzling. Inspite of performing cross-validation, and choosing the optimal

hyper-parameters, there was just a slight increase in accuracy to around 60%. We expected SVM to outperform Winnow, but our hypothesis turned out to be false. However, one noticeable difference was that SVM had better accuracy over Winnow on Train data. SVM had consistently achieved 70% accuracy in classifying the Training dataset.
The results of cross - validation to find the best values for C and γ_0 are shown below.

Results of SVM with C as hyper parameter	
C values	**Test accuracy**
0.01	50.793
0.05	49.392
0.1	49.878
0.5	50.344
1	**57.415**
5	51.675
10	51.728

Results of SVM with γ_0 as hyper parameter	
γ_0 values	**Test accuracy**
0.0005	53.626
0.001	52.479
0.005	**53.869**
0.01	51.559
0.05	49.783
0.1	48.805

Using the optimal **C** and γ_0 values found using cross validation, the Test dataset accuracy which we have obtained is **59%**.

4. **Logistic Regression Classifier**

   With no significant improvement with SVM, we moved forward and implemented Logistic Regression classifier with

$$\min_{\mathbf{w}} \left\{ \sum_{i=1}^{m} \log(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)) + \frac{1}{\sigma^2} \mathbf{w}^T \mathbf{w} \right\}$$

   as the loss function.

   Similar to SVM, we have performed cross-validation and the optimal σ and γ_0 values were found to be **80** and **0.005** respectively.

   Using these values for σ and γ_0, the accuracies on training set and test are **71%** and **56%** respectively.

5. **Random Forests**

   It seemed quite counter intuitive that batch algorithms were not performing any better than an on-line algorithm like Winnow. At this point we were fairly convinced that the dataset might not have been linearly separable and hence, we have tried to use the most computationally expensive tool : *Random forests*.

Based on the understanding from Machine Learning lectures, it is known that *Random forests* were the best ML classifiers for datasets that have < 300 features. As our Dataset just had 147 features, we were quite convinced that this would be our best bet at understanding this data.

Instead of implementing *Random forests* on the binary-featurized dataset, we presented the dataset with real-valued features. Also, the weekend assumption on how Monday's stock depended on Friday's performance (and not Saturday's and Sunday's) was also tweaked. *Random forests* assumed that Monday's stock prices depended on Friday, Saturday and Sunday's performance.

Random forests performed better than the best algorithm which we previously had, Winnow, when the number of trees was set between 25 to 40. We tried to narrow this down further, but owing to randomness, we could not. Below is the table which shows the trend in accuracies w.r.t Number of trees.

Results of Random Forest with number of trees as hyper parameter		
Number of trees	**Training accuracy**	**Test accuracy**
10	94.15%	60.2%
25	98.84%	73.9%
30	**98.84%**	**75.2%**
35	99.16%	69.3%
40	99.61%	69.5%
50	100%	73.9%
80	100%	69.5%
100	100%	62.3%

We can see that there is a drop in accuracies when number of trees exceeds 40.
We think this is where the algorithm might actually start over fitting data. We attribute it to the True error deviating away from the Empirical error.
We can see that the best accuracy is **75.2%** for **30** trees.

6. **k-Nearest Neighbours**

Results of kNN with k as hyper parameter	
k values	**Test accuracy**
1	52.1739130435
3	43.4782608696
5	43.4782608696
7	52.1739130435
9	60.8695652174
11	65.2173913043
13	**69.5652173913**
15	60.8695652174
17	47.8260869565
19	43.4782608696
21	60.8695652174
27	56.5217391304
33	39.1304347826

From the above experiment, we can see that the non-linear classifier like Random forests has worked much better than all the linear classifiers. Therefore, we were motivated to try out another non-linear classifier. We chose to implement *k-Nearest Neighbours* expecting better accuracy. We have performed cross validation over the best value for k and the results have been presented in the table above.

Conclusion

The best accuracy for the dataset that we had is **75.2%** using Random Forest algorithm for **25** to **40** trees.

In-spite of several attempts to improve the performance over the considered dataset using *Random Forests*, the maximum accuracy which we could obtain has been limited to **75%**. Based on the prior knowledge that we have about *Random forests*, we can say that there probably isn't a better classifier for this Dataset, which consists of < 300 features.

I would assume that this accuracy is limited only by design of the problem. Let us recall that we are trying to understand if there is a underlying relation between **customer satisfaction** and **trend in stock prices**. Therefore, we can conclude that *on-time performance of an airline* and **it's stock prices** are not conclusively relative to each other.

However, if we could integrate certain other metrics which also quantify *customer satisfaction* (e.g. Twitter sentiment analysis), we might be able to predict the relation between **customer satisfaction** and **stock trends** of a particular airline more accurately.

Future Work

- We could employ web-scraping techniques and fetch twitter data to see if adding the customer satisfaction ratio to the feature space would result in better accuracy.
- We could see if there are other feature sets that could be used along with our existing feature space to better predict the accuracy.
- Currently, we only used the American Airlines data for one year as our sample space. We could try to run the algorithms on a larger sample space. With increased sample space, neural networks might yield better results than the Random Forest algorithm.
- Currently, the labels in both are Training and Test sets are binary. We could try to quantitatively predict the increase or decrease in the stock prices instead of just trying to predict if the stock price would increase or decrease.