Capstone Project – Final Report

Sentimental analysis for Walmart using Twitter data.

Group Number: 17

Group members (Name, ID):

- **Able Raj Karimpanakal Rajan. (#0755853)**
- **Anvesh Chamakuri. (#0753863)**
- **Mohammed Taquee(#0754265)**

Introduction:

Social networks have revolutionized how people communicate. The information available from social networks is beneficial for analysis of user opinion, for example measuring the feedback on a recently released product, looking at the response to policy change, or the enjoyment of an ongoing event.

Sentiment analysis or opinion mining is a relatively new area, which deals with extracting user opinions automatically. Sentiment Analysis is a method that uses Natural Language Processing (NLP) or Machine Learning techniques to identify, extract information, and study customer sentiment on various aspects. An example of positive sentiment is, “natural language processing is fun” alternatively, negative sentiment is “it’s a horrible day, I am not going outside”. Objective texts such as news headlines does not seem to express any kind of sentiment, for example, “Government bans flights to Canada”.

Walmart being one of the biggest retailers in the world estimates to sees close to 300,000 social mentions every week. Walmart uses sentiment analysis to keep up with market trends, product performances, monitor brand reputations, etc.

In this paper, we present a tool that performs sentiment analysis of Walmart using data extracted from Twitter. We present a tool for sentiment analysis that can analyze Twitter data. We have demonstrated how the method to collect a corpus and using the corpus, we build a sentiment classifier, that can determine positive, negative, and neutral sentiments. We can get a lot of insights like what kind of products customers like and how often they buy it, if a product is negative what are the aspects affecting it, how online services are performing like whether the items are shipped fast, study about the most trending item and improving its sales, how the competitors are performing and how Walmart can improve or be unique from their competitors and list goes on as this is a vast topic to explore.

Related works:

A large amount of research has been done in the field of "Sentiment Analysis on Twitter" in recent years by several researchers. It was initially designed for binary classification, which assigns opinions or reviews to bipolar classes such as positive or negative only.

Work done by Vishal A. Kharde and team (2016) [8], show that machine learning methods, such as SVM and naive Bayes, have the highest accuracy and should be considered the baseline learning methods, whilst lexicon-based methods are very effective in some cases and require little effort in human-labeled documents. They also state that when compared to other models, the bigram model provides better sentiment accuracy.
Work done by Pranav Waykar and team (2016) [9], states that field of sentiment analysis research is to detail the preprocessing steps required to extract bags of words from Twitter data and the development of a topic-based sentiment analysis method. This approach is based on the fact that the complexity of an analysis can be reduced by focusing the algorithm on a narrower range of topics.
Hamid Bagheri's paper (2017) [10], demonstrated the use of sentimental analysis, as well as how to connect to Twitter and run sentimental analysis queries. They ran experiments on various topics ranging from politics to humanity and present the interesting results. They discovered that the neutral sentiment for tweets is significantly high, highlighting the limitations of the current works.

Methods

Data collection

The data for our project is the raw tweets that has been collected from the twitter. Twitter has a lot of data of real data, and we can easily obtain the tweets related to our field of interest. Since our project was focused on analyzing sentiments for Walmart data, we had to gain an access to collect the twitter data. This can be done using Twitter API where we get credentials to pull out the tweets that we want.

For our data, we collect over 50000 tweets in general for analysis using different keywords relating to Walmart. Our purpose was to compare the sentiments with different competitors of Walmart like BestBuy, Amazon, Freshco (grocery) etc. We also collected tweets for Walmart on particular events like black Friday, Boxing day and compared it with its competitors. These data were collected using specific keywords like #Walmart, #WalmartBlackFriday, #WalmartGrocery etc. Data is acquired by using a function called tweepy which provides a package for simple twitter API.

```
try:
  auth = OAuthHandler(consumer_key, consumer_secret)
  auth.set_access_token(access_token, access_token_secret)
  api = tweepy.API(auth)
  return api
except:
  print("Error")
  exit(1)
```

Data cleaning and preprocessing:

The tweets we obtained are highly unstructured and informal. So data preprocessing is very important step for our project.

Steps for preprocessing:

a) Remove twitter handles
b) Remove punctuations, special characters
c)converting texts to lower to make all texts uniform.
d) removing stopwords as it does not contain any meaning
e) Tokenization: Process of splitting the text into smaller units called tokens
f) Stemming: Process of removing last common end word or suffix
g) Lemmatization: Process of converting the word into base form.

Exploratory data analysis:

We used the library called TextBlob for assigning a sentiment for our cleaned tweets from the preprocessing stage. TextBlob is used for performing natural language processing tasks and is built on top of NLTK. TextBlob returns polarity and subjectivity features in a text. This assignment of the sentiments will help us to determine how our data is distributed. The observations are briefly discussed in result section.

Feature extraction:

Bag of words:

In order for machine to understand the text we might have to convert it into numerical vectors. One such method to do it is Bag of words. This is a representation of text describing the frequency of occurrence of word within the document. We keep track of the counts of the word and ignore the grammatical aspect of it.

```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features= 300)
```

TF-IDF:

TF- IDF is another technique to extract features, it's a combination of two terms : Term frequency and inverse document frequency.

- TF = (Frequency of a word in the document)/(Total words in the document)
- IDF = Log((Total number of docs)/(Number of docs containing the word))
-

It is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. This is done by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents.

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer (max_features=300, min_df=7, max_df=0.8, stop_words=stopwords.words('english'))
```

Modelling

Once the data is ready for modelling, we imply different classification models on our dataset like Niave bayes, SVM, Random Forest, Decision tree. All these models were implemented on both techniques of Bag of Words and TF-IDF. The dataset was divided as 80% training set and 20 % testing set.

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y, test_size=0.2, random_state=853)
```

a)**Modelling using Naïve bayes**

```
[180]: from sklearn.naive_bayes import GaussianNB
       nv = GaussianNB() # create a classifier
       nv.fit(X_train,y_train) # fitting the data
```

```
[180]: GaussianNB()
```

```
[181]: from sklearn.metrics import accuracy_score,confusion_matrix
       y_pred = nv.predict(X_test) # store the prediction data
       accuracy_score(y_test,y_pred) # calculate the accuracy
```

```
[181]: 0.6555555555555556
```

b) **Modelling using SVM**

[185]:
```
# Building a classification model for linear SVM after scaling
from sklearn.svm import SVC
clf_svm = SVC(kernel='linear',C=2)
clf_svm.fit(X_train, y_train)
y_pred = clf_svm.predict(X_test)
# Making the Confusion Matrix
con_mat = confusion_matrix(y_test,y_pred)
# Evaluating the results
print("Confusion Matrix is : {}".format(con_mat))
```

```
Confusion Matrix is : [[14  8]
 [ 7 61]]
```

[186]:
```
# Accuracy of the linear SVM
# calculate the accuracy of the Linear SVM on the training data
train_accuracies_m = []
test_accuracies_m = []
acc_train = clf_svm.score(X_train, y_train)
# calculate the accuracy of the Linear SVM on the test data
acc_test = clf_svm.score(X_test, y_test)
print(f'Train accuracy is {acc_train:.2f} and test accuracy is {acc_test:.2f}\n')
train_accuracies_m.append(acc_train)
test_accuracies_m.append(acc_test)
```

```
Train accuracy is 0.99 and test accuracy is 0.83
```

c)**Modelling using Decision tree**

```
# Building a classification model for Decision tree Classifier
from sklearn.tree import DecisionTreeClassifier
clf_dt = DecisionTreeClassifier()
clf_dt.fit(X_train,y_train)
y_pred = clf_dt.predict(X_test)

# Making the Confusion Matrix
con_mat = confusion_matrix(y_test,y_pred)

# Evaluating the results
print("Confusion Matrix is : {}".format(con_mat))
```

```
Confusion Matrix is : [[16  6]
 [ 8 60]]
```

[189]:
```
#Calculating the accuracies for the decision tree
# calculate the accuracy of the Decision tree with no parameters on the training data
acc_train = clf_dt.score(X_train, y_train)
# calculate the accuracy of the Decision tree with no parameters on the test data
acc_test = clf_dt.score(X_test, y_test)
print(f'Train accuracy is {acc_train:.2f} and test accuracy is {acc_test:.2f}\n')
# Adding the train and test accuracies to the lists so we can plot them later
train_accuracies_m.append(acc_train)
test_accuracies_m.append(acc_test)
```

```
Train accuracy is 0.99 and test accuracy is 0.84
```

d)**Modelling using Random Forest**

[191]:
```
for i in range(5, 12, 2):
    from sklearn.ensemble import RandomForestClassifier
    clf=RandomForestClassifier(n_estimators=i)
    clf.fit(X_train,y_train)
```

[192]:
```
# Fitting classifier to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_state = 853)
classifier.fit(X_train, y_train)
y_predict =classifier.predict(X_test)# Predicting the Test set results
con_mat = confusion_matrix(y_test,y_predict)
# Evaluating the results
print("Confusion Matrix is : {}".format(con_mat))
```

```
Confusion Matrix is : [[14  8]
 [ 6 62]]
```

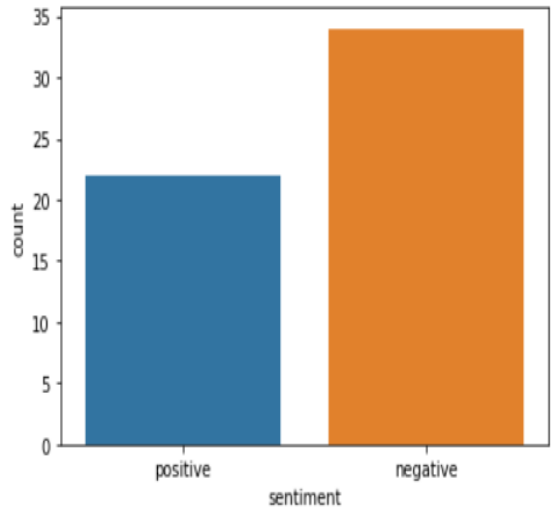
[193]:
```
print("Accuracy Score is : {}".format(accuracy_score(y_test, y_predict)))
```

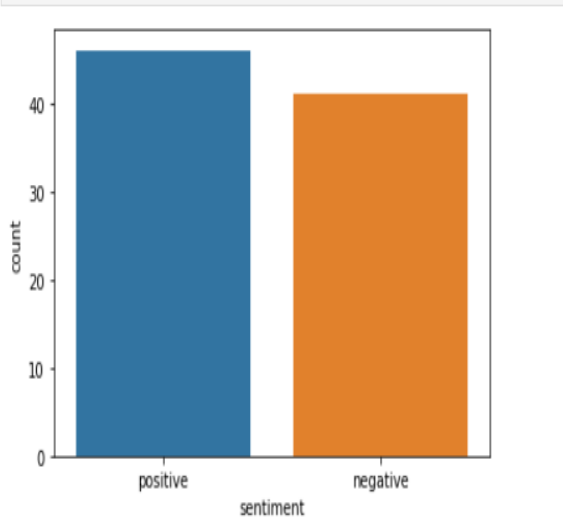
```
Accuracy Score is : 0.8444444444444444
```

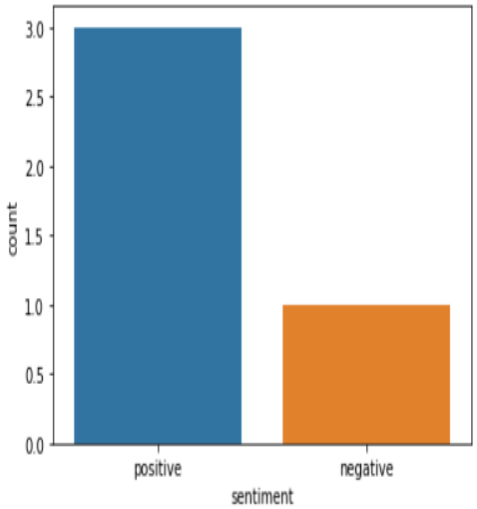
Results

The results observed from analysis shows or returns largely positive sentiments when we just use the keyword Walmart. This shows the general sentiment for Walmart is positive. But when we started using different parameters like black Friday, boxing day or Walmart grocery, the positive tweets kept on decreasing.

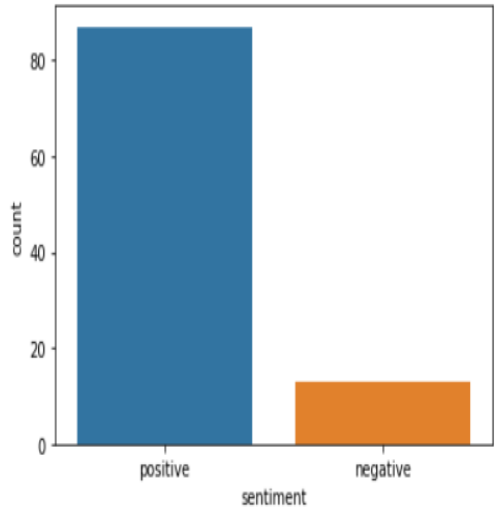
A) Fig 1 compares the positive tweets and negative tweets of Walmart sales on black Friday. The number of positive tweets has been comparatively less on black Friday. Fig 2 shows black Friday sale sentiments in Amazon where there more positive tweets related to negative. Fig 3 which is that of BestBuy had the best or highest positive sentiment among the three. This showed the people opinion on Walmart during black Friday was mostly negative and they performed poorly compared to their competitors.

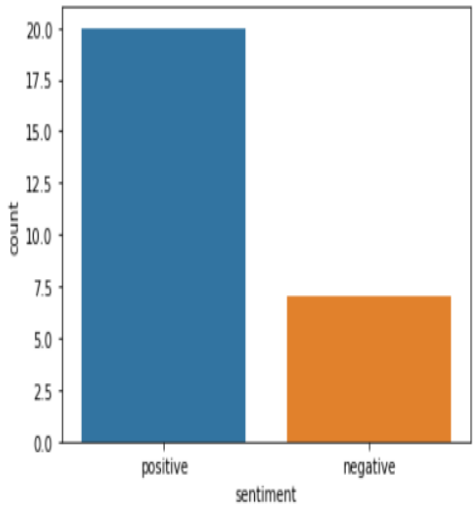






B) With grocery, Walmart showed comparatively better results compared to freshco. Fig 1 shows sentiments on Walmart grocery and fig 2 shows sentiments on freshco grocery. This shows Walmart is leading compared to their competitors in fresh market.





When it comes to modelling, we built different classification models like Naïve bayes, SVM, decision tree and Random forest. The following chart shows the accuracy charts obtained for all the models. Decision tree under TF-IDF was able to get the highest accuracy with 87%

Accuracy chart

Classification models	Bag of Words	TF-IDF
Naïve bayes	65.55%	42.22%
SVM	83%	81%

Decision tree	**84%**	**87%**
Random forest	**84.44%**	**83.33%**

Discussion

Our main objective was to study and analyze the customer sentiments. We were able to get hands on experience working with NLP by using different toolkit like NLTK and Textblob. After analyzing the sentiments, we were able to build different machine learning models like Naïve bayes, SVM, Random Forest, Decision tree etc. Although Naïve bayes had least accuracy of 65%, other models where able to get an accuracy of more than 80% for all cases which is comparatively good when we talk about the field of sentimental analysis. We believe our models where able to effectively predict the sentiments.

One of the most difficult challenges we faced in our project was cleaning the data because Twitter data is highly unstructured and informal, and we spent a significant amount of time cleaning it. The following are some of the difficulties encountered in Twitter Sentiment Analysis- Recognizing subjective textual elements: Subjective content is represented by subjective parts. In one case, the same word can be considered subjective, while in another it can be considered objective. As a result, it is difficult to identify the subjective parts of the text. Then there is Explicit Negation of sentiment, Entity Recognition, Handling comparisons as Bag of words model does not handle comparisons very well.

Sentiment analysis is typically performed on longer text. Twitter messages are short due to text limitations, and the algorithm has fewer features available for analysis. Also there needs to be work done to integrate Sarcastic sentences as they express negative opinion about a product using positive words in unique way.

Conclusion

In this paper, we provide a comparative study on sentiment analysis of Walmart using data extracted from Twitter in the form of tweets. We have demonstrated how to collect tweets from Twitter using Twitter API and convert them into a dataset. We have looked at common NLP processes and machine learning models to help us derive the class of the tweet, in this case positive, negative, or neutral. We discovered that a lot of the tweets that we collected were neutral and very few of the tweets turned out to be positive. This was predicted at the start of the project as it is a general nature as people tend to tweet or complain more about a failed product than write positives about the same. We tried to implement various machine learning models like Naïve Bayes, SVM, Decision Tree, and Random Forest.

The best performing model was Decision tree under TF-IDF with an accuracy of 87%. For Bag of words, the highest accuracy of 84.44% was achieved by Random Forest model followed by 84% for the Decision Tree and 83% for the SVM model. For TF-IDF, it was Decision tree followed by Random forest and SVM with accuracies of 87%, 83.33% and 81% respectively. The Naïve Bayes

model had the lowest accuracy for both Bag of words and TF-IDF with accuracy of 65.55% and 42.22% which comparatively very low.

Future work:

- Since our project focused more on classification models, the future scope of study will be to implement deep learning models like LSTM and RNN.
- Another feature worth investigating is whether the information about the relative position of words in a tweet has any effect on the classifier's performance.
- To apply sentiment analysis to real-world application cases, neutral data must be handled. As a result, future research should concentrate on these neutral data.

Contributions

Since there was a limit to the number of tweets a person could extract in a given amount of time, all the members have contributed towards data collection. Each one was able to collect different types of data and data with different keywords.

- **Able Raj –** Along with collecting tweets, he has performed data preprocessing, cleaning, and performed EDA on the data. He was responsible for feature extracation and implementing methods like Bag of Words and TF-IDF. He has created the Random Forest and Decision tree models for both bag of words with accuracies of 84.44% and 84% respectively and for TF-IDF with accuracies 83.33% and 87% . Along with that, he has contributed to creating the final report of the project.
- **Mohammed Taquee –** He has contributed to data collection by extracting tweets. Taquee has helped in creating the presentation file and the final report of the project. He has created the SVM model for both bag of words and TF-IDF that gave an accuracy of 83% and 81% respectively. Taquee has created the GitHub repository for the project.
- **Anvesh Chamakuri –** He was responsible for creating the presentation file and the final report of the project. He was also responsible for data collection. Anvesh created the Naïve Bayes model with an accuracy of 65.55%.

References

1. https://monkeylearn.com/sentiment-analysis/
2. https://www.businessnewsdaily.com/10018-sentiment-analysis-improve-business.html
3. https://www.mytotalretail.com/article/how-retailers-and-tech-can-use-sentiment-analysis-to-tap-into-consumer-demand/
4. https://www.mindtree.com/blog/future-of-data-driven-sentiment-analysis-retail-industry

5. https://www.dezyre.com/article/how-big-data-analysis-helped-increase-walmarts-sales-turnover/109
6. https://arxiv.org/ftp/arxiv/papers/1601/1601.06971.pdf
7. https://www.researchgate.net/profile/Sujithra_Muthuswamy/publication/325359363_Sentiment_Analysis_on_Twitter_Data_Using_Machine_Learning_Algorithms_in_Python
8. https://arxiv.org/ftp/arxiv/papers/1601/1601.06971.pdf
9. http://ijarcet.org/wp-content/uploads/IJARCET-VOL-5-ISSUE-1-79-81.pdf
10. https://osf.io/6xc4y/

Appendices

- We have used below tools to implement our project
  i)Anaconda Navigator
  II)jupter notebook
  iii)python 3
- Our project consisted of multiple excel files as we have differentiated tweets based on our case study. All are included in final submission.
- Our project has two python files
  i)Sentiment_Textblob.ipynb: Includes connecting to twitter API, extracting tweets, assigning sentiments, EDA.
  ii)SentiAnalysisModels.ipynb : Includes all machine learning models implemented
- https://github.com/taquee94/Sentiment-Analysis-on-Walmart-using-Twitter-Data: Our github repository