Logo

Description automatically generated

**College of Professional Studies**

**Northeastern University San Jose**

**MPS Analytics**

**Course: ALY6040 – Data Mining Application**

**Assignment:**

Module 6 Final Report

**Submitted on:**

May 19th, 2023

**Submitted to:**  **Submitted by:**

Professor: BEHZAD AHMADI ARCHIT BARUA

BHAGYASHRI KADAM

NIKSHITA RANGANATHAN

**ABSTRACT**

Sentiment analysis is the process of identifying and extracting the emotional tone or attitude expressed in text data, such as social media posts, comments, and reviews. It employs methods – like natural language processing and machine learning.

Social media companies can benefit greatly from sentiment analysis because it helps them to monitor and understand how their brand or products are being perceived by customers and the public. Companies can track changes in sentiment over time, identify trends and patterns in customer feedback, and quickly address any negative comments or complaints by analyzing the sentiment of social media posts and comments related to their brand.

Sentiment analysis can also be used to identify key influencers and advocates for a brand or product and to track how sentiment towards a particular topic or event changes over time. This can help companies to make informed decisions about their marketing and public relations strategies, and to develop more effective communication and engagement strategies with their customers.

Overall, sentiment analysis is an important tool for social media companies to gain insights into how their brand is perceived by customers, and to make data-driven decisions about how to improve customer satisfaction and engagement.

**INTRODUCTION**

**About this Dataset:**

**The Twitter US Airline Sentiment** dataset contains sentiment analysis information about the problems of each major US Airline. It was scraped from Twitter in **February 2015**. Tweets about six significant US airlines are included.

The dataset consists of 14640 instances and 15 features, all of which are traveler-submitted tweets. Each instance of sentiment is classified as either **positive, neutral, or negative.**

As part of the final project, we are interested in finding the relationship between the Airline sentiment (positive/negative/neutral) and the airline, analyzing the reasons for the negative tweets, and eventually building, training, and testing a model that can predict whether the tweet is positive, negative, or neutral.

**Understanding the dataset:**

This dataset was originally gathered by **Crowdflower's Data for Everyone library** and taken from the Kaggle platform.

Both numerical and categorical data types are present in this dataset. Below are the basic statistics of the Raw dataset:

A picture containing table

Description automatically generated

***Figure 1 – Basic Statistics of the dataset***

Below is the Percentage wise Distribution for Rows, Columns & Observations used in this Dataset:

Chart, bar chart

Description automatically generated

***Figure 2 – Memory Usage***

Below is the Data Structure of the Twitter dataset:

**Chart, radar chart

Description automatically generated**

***Figure 3 – Data Structure***

Below are the data descriptions of each variable of the data that briefly describe the contents of the data set. The dataset's features are as follows:

| **No** | **Feature** | **Dictionary** |
| --- | --- | --- |
| 1. | **tweet\_id** | A unique identifier for each tweet. |
| 2. | **airline\_sentiment** | The sentiment expressed in the tweet (positive, negative, or neutral). |
| 3. | **airline\_sentiment\_confidence** | The level of confidence in the sentiment classification (ranging from 0.34 to 1.0). |
| 4. | **negativereason** | The reason for the negative sentiment. |
| 5. | **negativereason\_confidence** | The level of confidence in the negative reason classification (ranging from 0.0 to 1.0). |
| 6. | **airline** | The name of the airline being referenced in the tweet. |
| 7. | **airline\_sentiment\_gold** | A field that is no longer used. |
| 8. | **name** | The Twitter handle of the user who posted the tweet. |
| 9. | **negativereason\_gold** | A field that is no longer used. |
| 10. | **retweet\_count** | The number of times the tweet has been retweeted. |
| 11. | **text** | The content of the tweet. |
| 12. | **tweet\_coord** | The latitude and longitude coordinates of the tweet's location (if available). |
| 13. | **tweet\_created** | The date and time the tweet was posted. |
| 14. | **tweet\_location** | The location of the tweet (if available), as indicated by latitude and longitude. |
| 15. | **user\_timezone** | The timezone of the user who posted the tweet. |

*Table 1: Dictionary with the Features of the Twitter US Airline Sentiment Dataset*

**Below are the questions we plan to answer through our Analysis:**

**Business Questions -**

1. How do different airlines compare in terms of sentiment distribution? Are there any airlines that consistently perform better or worse than others?
2. When are customers most likely to tweet about airlines? Are there any specific patterns observed in the data?
3. What are the most common reasons for negative sentiment towards different airlines? Are there any areas that need improvement?
4. How does the volume of tweets vary over time and across different airlines, and what factors might be driving these trends?
5. Which locations have the highest number of tweets?
6. What are the most frequently used words in tweets, and do they suggest any common issues that airlines need to address?
7. How does tweet length vary by sentiment and what can airlines learn from this information?

**DATA CLEANING AND MANIPULATION**

* **Checking the number of missing values for each Attribute in the dataset**

**Table

Description automatically generated**

***Figure 4 – Missing data***

From the above missing plot, we can see that there are a lot of missing values present only in the Negative Reason confidence column (around 28%, i.e 4118 records). The other variables/attributes do not seem to have any missing records.

* **Dropping the tweet\_id column**

As we can see from the dataset, column – **twitter\_id**  is not required or significant for our analysis, hence we have dropped it.

* **Replacing Missing values with their Mean - Negative Reason Confidence**

In order to handle the missing values for Negativereason\_confidence, we have imputed the missing values with the respective mean of the column. This approach can help to minimize the loss of information and can be a simple way to handle missing values without introducing any bias.

* **Removing duplicate rows**

Duplicate rows in the dataset can lead to inconsistencies and affect the accuracy of data analysis. It is essential to find and remove any duplicate rows from the dataset before starting the analysis.

In this project, we used the Duplicated () and AnyDuplicated() functions to check for duplicate rows in the Twitter Airline Sentiment dataset. After confirming that there were 36 duplicate values, we removed them. A clean dataset of 14604 records was produced as a result, which was then used for additional analysis.

* **Extracting hour values from time after separating date and time columns**

In order to analyze tweet data more effectively, the date and time information was separated into two distinct columns in the dataset. This was achieved using the "separate" function in R, which split the original "tweet\_created" column into two columns. Subsequently, the "hour" value was extracted from the "time" column using the "hour" function from the "hms" package.

* **Replacing column values**

The "Hours" column values were replaced with categories of Morning, Afternoon, Evening, and Night using the "mutate" and "recode" functions. This transformation allows for easier interpretation and analysis of time-related data in the dataset.

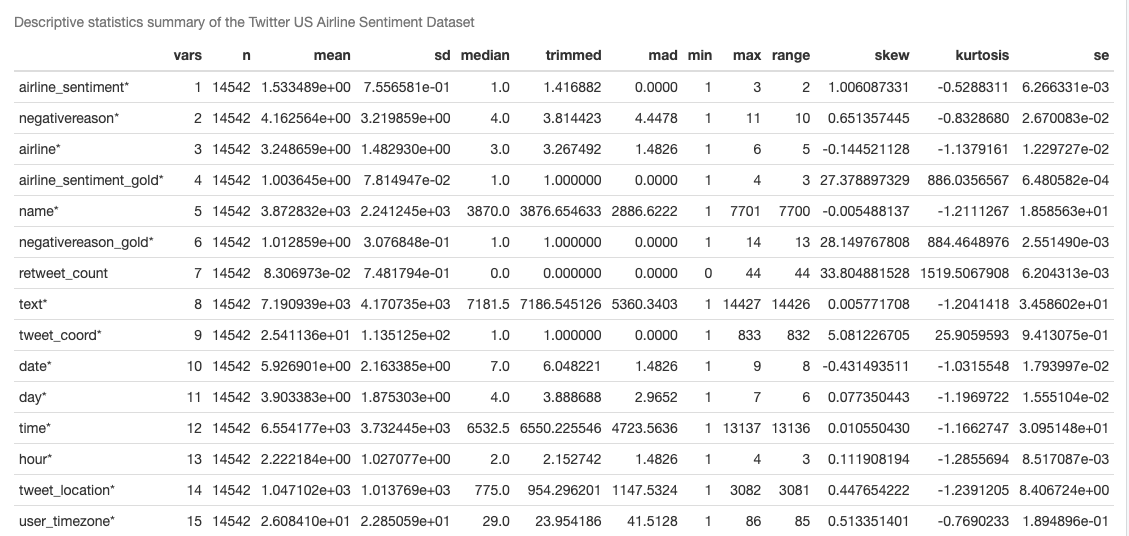
* **Creating a day column**

A 'day' column was added to the dataset and its values ​​were derived from the 'date' column. Dates from '2015-02-16' to '2015-02-24' have been replaced with the corresponding day of the week.

* **Changing datatypes**

The columns (airline sentiment, airline, day, and hour) were converted from one data type to factor using as.factor().

**DESCRIPTIVE CHARACTERISTICS OF THE DATASET**



***Figure 5 – Descriptive statistics***

In the table, each variable is given a number of descriptive statistics, including the mean, standard deviation, median, trimmed mean, median absolute deviation, minimum, maximum, range, skewness, kurtosis, and standard error of the mean (se).

While the others are continuous variables, the variables marked with an asterisk (\*) are categorical variables.

Some key descriptive statistics for the dataset are:

* The mean sentiment score is 1.53, indicating that the majority of tweets in the dataset are negative. The median sentiment score is 1.00, which is also the score for a neutral tweet, indicating that there are a significant number of neutral tweets in the dataset as well. The sentiment score's standard deviation, which is relatively high at 0.75, shows that the sentiment scores range widely.
* The retweet count is low, with an average of 0.08.
* The text of the tweet is on average 7,190 characters long, and there are few tweets with coordinates and locations.

**EXPLORATORY DATA ANALYSIS (EDA)**

**Chart, histogram

Description automatically generated**

***Figure 6 – Histogram***

The above histogram shows the distribution of the Retweet Count for the tweets done. We can observe that the retweet count does not follow any positive or specific distribution as the retweet count values are majorly between 0.0 to 0.5 with a count greater than 10,000.

**Chart, scatter chart

Description automatically generated**

***Figure 7 – Scatterplot***

The above line dot plot is an interactive plot that explains the daily cumulative tweets received from the date 16th Feb to 24th of Feb. From the above plot, we can understand that the overall daily cumulative count shows an upward trend for the given period. Starting with 16th feb, the daily cumulative count was observed less than 5, however until the 20th of Feb, the daily cumulative count was almost 5x and accounted to 5632.

Finally, on the 24th of feb, the daily cumulative count was observed to be 14604.

**Chart, histogram

Description automatically generated**

***Figure 8 – Multiple density plots***

The above density graph/plot represents the distribution of negative tweets that were received for each of the Major US airlines. The plot consists of several density curves ​​and provides insight into the distribution of negative tweets by the airline, with different airlines having different peaks and shapes of density distributions.

The x-axis represents the count for negative tweets and the y-axis represents the density of the negative tweets for each airline. From the graph, we can understand that American Airlines shows a flatter distribution with less pronounced peaks suggesting that negative tweets for this airline are more evenly distributed across counts. Virgin America has the highest concentration of negative tweets with the count range of 0-100.

Further, Delta & Southwest Airline have density curves almost over-lapping with less concentration of negative tweets as compared to American airlines and a count range of 80-180.

Lastly, United & US Airways have less peaks suggested that the negative tweets for these airlines are evenly distributed across the counts up to 600.

**Chart, radar chart

Description automatically generated**

***Figure 9 – Circular barplot***

The above circular bar chart represents the highest number of tweets received from the top 10 locations in the USA as per count. The bars are colored based on the tweet location and the length of the bars represents the tweet count for each location. The circular layout of the bars creates a visually appealing display, and the use of polar coordinate system can be useful for comparing the values of each location.

From the above chart, we can understand that the highest number of tweets are from New York (638), followed by Washington, D.C (335) & San Francisco (213). The least number of tweets have been received from Houston, TX (84) & Dallas, TX (97).

Chart

Description automatically generated

***Figure 10 – Animated line graph***

The above-animated line graph represents the timeline of daily tweets that were received for each US Airline from the period Feb 16th, Feb 24th, 2015.

The graph displays a line for each airline, with the date on the x-axis and the number of tweets on the y-axis.

The animation feature of the graph reveals the daily tweet counts for each airline over time, allowing us to see how the tweet volume for each airline changes over time. This can be helpful in identifying patterns or trends in the data, such as spikes or dips in tweet volume, as well as any seasonal or cyclical trends.

We can understand the below for each Airline -

**American**: The line for American is orange, and it starts out with a low number of tweets on Feb 18th. The tweet volume gradually increases showing the largest spike ever across all airlines on Feb 23rd.

**Delta**: The line for Delta is yellow, and it starts out with a moderate number of tweets on Feb 16th. The tweet volume for Southwest also gradually increases on Feb 22nd with a few spikes and dips again along the way.

**Southwest**: The line for Southwest is green, and it starts out with a higher number of tweets after United on Feb 17th. The tweet volume for Southwest shows certain spikes and dips along the way and gradually ends with the 2nd lowest dip on 23rd Feb.

**United**: The line for United is sky blue, and it starts out with a very low number of tweets on Feb 16th. The tweet volume for United gradually increases showing the 2nd largest spike ever across all airlines on Feb 23rd.

**US Airways**: The line for US Airways is cobalt blue, and it starts out with a moderate number of tweets on Feb17th showing few spikes and dips along the way with its highest daily peak on Feb 22nd and gradually showing a dip on Feb24th.

**Virgin America**: The line for Virgin America is purple, and it starts out with the 2nd lowest number of tweets on Feb17th. The tweet volume for Virgin America is mostly flat throughout the rest of the dates, with a few small spikes and dips and ending with the lowest no of tweets on 24th Feb.

Chart, treemap chart

Description automatically generated

***Figure 11 – Treemap***

This above treemap visualization represents the negative reasons for tweets about airlines. Each negative reason is represented by a rectangle of varying sizes on the treemap, and the size of the rectangle is proportional to the number of tweets associated with that negative reason.

The plot shows that the most common negative reason for tweets about US airlines is "Customer Service Issues," followed by "Late Flight" and "Can't Tell," which indicates that customers are dissatisfied with the airlines' customer service and flight punctuality. Other negative reasons include "Cancelled Flight", "Lost Luggage", "Bad Flight", "Flight Booking Problems", "Flight Attendant Complaints", and "longlines".

Chart, pie chart

Description automatically generated

***Figure 12 – Multiple pie charts***

The pie charts give the distribution of sentiments (positive, negative, and neutral) across six major airlines in the US. This is a great and simple way to compare the results and draw conclusions. It can be seen that all the airlines have the majority of tweets expressing negative sentiment, with US Airways and American Airlines having the highest proportion of negative tweets at over 70%.

These findings suggest that airlines need to address the underlying issues causing negative sentiment on Twitter to improve their reputation and customer satisfaction.

Chart, bar chart

Description automatically generated

***Figure 13 – Stacked bar graph***

This column chart represents the sentiment analysis of tweets about different airlines. The chart is grouped by airline and sentiment, and the count of tweets for each sentiment is shown using a stacked bar chart. The x-axis represents the different airlines, and the y-axis represents the total count of tweets.

We can see that United Airlines has the highest number of total tweets, followed by US Airways and American Airlines. However, when we look at the sentiment breakdown, we see that most tweets for these three airlines are negative, with more than 50% of tweets having a negative sentiment.

Furthermore, United Airlines has the highest number of negative tweets among all the airlines, with 2633 negative tweets. This suggests that United Airlines has a higher volume of negative sentiment on Twitter compared to other airlines.

Overall, this graph highlights the importance of sentiment analysis for businesses, particularly in the airline industry, as it provides insights into how customers feel about their products and services.

Chart, bar chart

Description automatically generated

***Figure 14 – Stacked bar graph 1***

From the output, we can see that the highest number of tweets for any day of the week is 3079 tweets on Sunday. The next highest number of tweets is on Monday, with 3032 tweets. The lowest number of tweets is on Wednesday, with only 1344 tweets.

This information can be useful for analyzing patterns and trends in the data, such as which days of the week have the highest or lowest engagement on social media.

Additionally, we can see that the highest number of tweets for any day and hour of the day is 1465 tweets on Tuesday morning. The next highest number of tweets is on Sunday afternoon, with 1410 tweets. The lowest number of tweets is on Wednesday night, with only 138 tweets.

This information can be useful for analyzing patterns and trends in the data, such as which times of the day and which days of the week have the highest or lowest engagement on social media. This can help you determine the best times to post content and engage with your followers.

Chart, bar chart

Description automatically generated

***Figure 15 – Stacked bar graph 2***

The graph shows the counts of negative reasons for tweets about airlines, grouped by the specific airlines. The x-axis shows the different negative reasons, and the y-axis shows the count of tweets.

From the graph, we can see that the most common negative reason for tweets is "Customer Service Issue", with 2904 tweets. The second most common negative reason is "Late Flight", with 1660 tweets, followed by "Cancelled Flight" with 1190 tweets, and "Can’t Tell" with 843 tweets. The least common negative reason for tweets is "Damaged Luggage", with only 74 tweets.

We can see that different airlines have different areas of opportunity for improvement. US Airways has the highest number of tweets related to customer service issues, with 811 tweets, while American Airlines follows closely with 762 tweets. United Airlines had the highest number of tweets related to late flights, with 525 tweets, followed by US Airways with 453 tweets. Additionally, American Airlines had the highest number of tweets related to canceled flights, with 242 tweets, followed by US Airways with 189 tweets.

Airlines can improve their operations or customer service in these areas by analyzing the results and determining where they need to make changes. The airline's reputation and financial performance may ultimately benefit from higher customer satisfaction and loyalty as a result.

Chart, bar chart

Description automatically generatedText

Description automatically generated

***Figure 16 – Wordcloud and Barplot***

We used word cloud and barplot to understand the most frequently used words in tweets related to airlines.

In the word cloud, larger words indicate a higher frequency of use. On the other hand, a bar chart is interactive, and hovering over the bars shows the exact count of the words in the tweets.

The results show that the most commonly used words in tweets are "flight" followed by "united," "usairways," and "americanair." This indicates that people are primarily talking about their flight experiences and mentioning the airline they flew with.

Both plots can provide valuable insights for airlines to understand the topics that are discussed by their customers on social media, which in turn can help airlines improve their services and address any issues that their customers are facing.

Chart

Description automatically generated

***Figure 17 – Ridge Chart***

The Ridge chart displays the density distribution of tweet text length across the three sentiment categories: negative, neutral, and positive. The x-axis represents the text length, and the y-axis represents the sentiment category.

The chart highlights the relationship between tweet text length and sentiment. It also suggests that negative sentiment tweets are typically longer than neutral and positive tweets, and people tend to express negative experiences with more details and explanations.

**TEXT PRE-PROCESSING**

In this section of the assignment, we have preprocessed a collection of tweets (related to airline customer experience). This includes cleaning, formatting, and transforming raw text data into a format that is suitable for analysis. The goal of text preprocessing is to prepare the text data for analysis, making it easier for ML algorithms to learn from the data and provide accurate results.

We begin by converting the tweets to UTF-8 encoding and creating a Corpus object from the text data. We then use various functions from the "tm" package to clean and normalize the text data. We convert all text to lowercase, remove punctuation, remove numbers, remove stop words, remove airline-specific terms such as flight and plane. Extra white space is also removed from the text, and each word is reduced to its base form using stemming. This is done to reduce the number of unique words and to group similar words together.

We then created a TermDocumentMatrix from the pre-processed text and removed sparse terms from it. Sparse terms refer to terms in the matrix that occur with a very low frequency. Removing sparse terms from the matrix can help reduce its size and improve the efficiency of analysis.

Chart, histogram

Description automatically generated

***Figure 18 – Most used words in tweets***

Next, we wanted to know which words occur the most in the tweets. Therefore, we calculated the frequency of each word and selected words with a frequency greater than 500. We visualized the results by sorting the selected words and creating a bar chart. The resulting plot can provide insights into the topics or themes being discussed and can be useful for businesses to understand customer sentiment or for researchers to analyze public opinions and trends.

Text

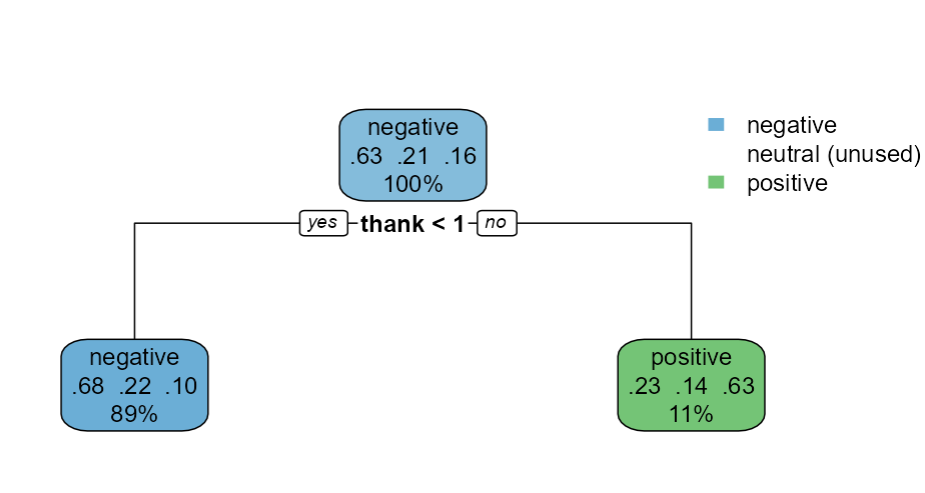
Description automatically generated

***Figure 19 – Baseline accuracy of test and train set***

Finally, the data is split into training and testing sets with 80:20 ratio (meaning that 80% of the data is used for training the model and 20% is used for testing the model's performance), and baseline accuracies for both sets are computed. Data splitting is an important step in machine learning to evaluate how well a model performs on new data.

The baseline accuracy is the proportion of the most frequent class in the dataset. In this case, the majority class is "negative" with a proportion of approximately 0.63 in the training set and 0.62 in the testing set.

**DECISION TREE**



***Figure 20 – Classification tree***

The decision tree is a powerful machine learning algorithm that is used for both classification and regression tasks. Decision tree is a tree-structured model that divides the dataset into smaller and smaller subsets based on the values of selected features, and then it predicts the target variable by analyzing the values of those features. It is also useful for handling both numerical and categorical data.

Text, letter

Description automatically generated

***Figure 21 – Summary of decision tree***

* The CP is a tuning parameter that controls the size of the tree, and a smaller CP value results in a larger tree with more splits. In this case, the CP value of 0.1171357 resulted in two splits, with a final tree containing two terminal nodes.
* The "thank" variable has an importance of 99 and the highest "improve" value, indicating that it was the most important variable in predicting sentiment. The tree has two levels, with the root node splitting on the feature "thank".
* Node 1 represents the root node, and it has 11,756 observations.
* Node 2 represents the left child node of Node 1, and it has 10,446 observations. The predicted class for this node is also negative. On the other hand, Node 3 represents the right child node of Node 1, and it has 1,310 observations. The predicted class for this node is positive.
* The other variables - primary splits (hour, great, delay, usairway) and surrogate splits (twitter, prompt, safe, southwestair) can also be observed in the outcome. However, their influence on the classification (improve value) is lower than that of "thank".

A picture containing text

Description automatically generated

***Figure 22 – Accuracy of decision tree***

The model correctly predicted the sentiment of around 67% of the tweets in the test set.

In a classification problem, it is important for a model to be able to accurately classify all classes present in the dataset. In this case, the model is unable to provide any classification for the neutral class, which means it is not able to distinguish between neutral tweets and tweets with a positive or negative sentiment. This is a major limitation as the model fails to provide a complete picture of the sentiment analysis task.

**Contribution**: Team member Nikshita Ranganathan performed the Decision Tree method.

**RANDOM FOREST**

| After the implementation of decision tree, we have applied the Random Forest model to predict the airline sentiment. We have built the Random Forest model using the Train set and 20 trees so that we can run the model in limited time.  Table  Description automatically generated  ***Figure 23 – Random forest model***  From the above output, we can observe the following –   * + - * For each split in each tree, a random subset of 33 variables were considered for splitting.       * The estimated out-of-bag (OOB) error rate is 26.2%       * From the confusion matrix, we can see that the model was able to correctly predict the 6935 Negative tweets, 1151 neutral tweets and 1069 positive tweets. While misclassifying 651 negative tweets, 1077 neutral tweets and 500 positive tweets.       * Lastly, from the class.error metric shows us the misclassification error rate which is highest for the neutral tweets – 53.73%   Further, we have made predictions on both the Training & Test data using the above random forest model and evaluated their accuracy respectively.  Graphical user interface, text, application  Description automatically generated  Graphical user interface, text, application  Description automatically generated  ***Figure 24 – Accuracy of test and train set (Random forest)***  Finally, we have evaluated the Model’s performance using the confusion matrix.    ***Figure 25 – Confusion matrix (Random forest)***  From the above results, we can say that the Random Forest Model has an accuracy of 75.12% which tells us that the model correctly predicted the sentiment of 75.12% of the tweets in the test set.  Further, we can see that the model correctly predicted 1596 negative tweets (true negatives) and 275 positive tweets (true positives). However, the model did not perform as well on the neutral tweets, with only 282 (true positives) being correctly predicted.  The model incorrectly predicted 251 instances of negative sentiment, 133 instances of neutral sentiment, and 52 instances of positive sentiment.  Lastly, for this model, the sensitivity values for the negative, neutral, and positive classes are 0.8852, 0.48205, and 0.57531, respectively. This means that the model can correctly identify 88.08% of negative cases, 48% of neutral cases, and 58% of positive cases.  The specificity values for the negative, neutral, and positive classes are 0.6341, 0.91320, and 0.94724, respectively. This means that the model can correctly identify 63% of negative cases, 91% of neutral cases, and 95% of positive cases.  **Contribution**: Team member Bhagyashri Kadam performed the Random Forest method.  **KNN MODEL**  We began by splitting the dataset into a training set (80%) and a testing set (20%). We then removed the sentiment column from both sets, leaving only the features for model training and testing. Next, we normalized the data using centering and scaling techniques, which help to improve the performance of distance-based algorithms like KNN. After preparing the data, we performed hyperparameter tuning using grid search and 5-fold cross-validation. We searched for the optimal value of 'k' (the number of neighbors) in the range of 5 to 30. Finally, we trained the KNN model using the optimal hyperparameter values and assessed its performance on the test dataset by calculating the accuracy and generating a confusion matrix.    ***Figure 26 – KNN Model***  The grid search identified 'k' = 9 as the optimal number of neighbors for the KNN model. The model achieved an accuracy of 63.75% on the test dataset, which is slightly higher than the no-information rate of 62.91%. However, the p-value of 0.1818 suggests that the model's performance is not significantly better than just predicting the majority class. The confusion matrix provides a detailed view of the model's performance for each class. The diagonal elements represent the correct predictions for each class, while the off-diagonal elements show the incorrect predictions. The model's performance metrics for each class are as follows:  Negative sentiment: Sensitivity (Recall) = 66.89%, Specificity = 75.63%, Precision (Pos Pred Value) = 82.32%  Neutral sentiment: Sensitivity (Recall) = 62.74%, Specificity = 73.26%, Precision (Pos Pred Value) = 37.56%  Positive sentiment: Sensitivity (Recall) = 53.14%, Specificity = 92.88%, Precision (Pos Pred Value) = 59.91%  Our model achieved an accuracy of 63.75% on the test dataset, but its performance was not significantly better than the no-information rate. The confusion matrix and performance metrics revealed areas where the model could be improved, particularly for the neutral sentiment class.  Future work could explore alternative preprocessing techniques, feature engineering, or other machine learning algorithms to enhance model performance. Additionally, more advanced techniques like ensemble learning or deep learning could be employed to further improve sentiment analysis results.  **Contribution**: Team member Archit Barua performed the KNN method.  **SUPPORT VECTOR MACHINE**  We developed a Support Vector Machine (SVM) model to predict airline sentiment. For this, we utilized C classification, which is suitable for classification problems involving multiple classes (in our case - negative, neutral, and positive). We also chose to use a linear kernel, which is a good fit for text classification like sentiment analysis. It works well on large datasets and trains faster than other kernels, making it a favorable option.  Below are the results of the model:   * The SVM model's accuracy is 0.7687 and its F1 score is 0.857, indicating good performance in predicting tweet sentiment. * The model's ability to detect negative sentiment is best, with a sensitivity of 0.86 and a specificity of 0.74. * The model is less accurate at predicting neutral sentiment with a sensitivity of 0.54 but performs well in detecting positive sentiment with a sensitivity of 0.68 and a specificity of 0.90.       ***Figure 27 – SVM Model***  **WORDCLOUD**  We generated a comparison cloud that helps us to quickly identify the main themes related to each sentiment and makes it easier to compare and analyze the expressed opinions in the tweets.  We first subsetted the tweets based on their sentiment into positive, negative, and neutral categories. Then, the term-document matrix is created, where we apply various text preprocessing steps such as removing punctuation, numbers, stemming, converting to lowercase, and eliminating stopwords.  Lastly, using the comparison.cloud function, we created a wordcloud which visually displays the most frequent words associated with each sentiment category. The words are color-coded for easy identification, with green for positive, red for negative, and blue for neutral sentiments. Additionally, the size of the words reflects their frequency, with larger words indicating higher occurrence.    ***Figure 28 – Wordcloud***  **MULTINOMIAL LOGISTIC REGRESSION**  We used multinomial logistic regression instead of a normal one because the outcome variable (in this case, the sentiment category) has more than two categories - three sentiment categories: negative, neutral, and positive.  The model achieved an overall accuracy of 0.7516, indicating that it correctly predicted the sentiment category for approximately 75.16% of the instances. The 95% confidence interval for the accuracy is (0.7353, 0.7673). The p-value is less than 0, indicating that the model performs significantly better in predicting the most frequent sentiment category.  Looking at the statistics by class, the model performed better at predicting negative and positive sentiment than neutral sentiment. The sensitivity of the model was highest for negative sentiment, indicating that the model was good at correctly identifying negative sentiment instances.      ***Figure 29 – Multinomial Logistic regression Model***  **COMPARING MODEL RESULTS**    ***Figure 30 – Comparison Table of all models***    ***Figure 31 – Comparison Plot of all models***   * To summarize, on comparing all the models based on accuracy, the SVM model achieved the highest accuracy of 76.8%, followed by Multinomial Logistic Regression (75.16%), Random Forest (75.05%), KNN (62.56%), and Decision Tree (67.02%).   In terms of the F1-score, which considers both precision and recall, the SVM model outperformed the other models with an F1-score of 0.8565., Multinomial Logistic Regression had an F1-score of 0.8477, Random Forest had an F1-score of 0.8416, KNN had an F1-score of 0.7251, and Decision Tree had an F1-score of 0.7907.   * Overall, the SVM model demonstrated the highest performance in terms of both accuracy and F1 score, making it the top-performing model in this sentiment analysis task. It effectively captured the complex patterns in the data and provided reliable predictions. Random Forest and Multinomial Logistic Regression also performed reasonably well, while KNN and Decision Tree showed slightly lower performance.   **Comparison based on Sensitivity and Specificity**  Here is the comparison of the results in terms of sensitivity and specificity for all models:  **Decision Tree:**  Sensitivity: Negative – 0.9529, Neutral – 0.0000, Positive – 0.4247  Specificity: Negative – 0.2248, Neutral – 1.0000, Positive – 0.9493  **Random Forest:**  Sensitivity: Negative – 0.8769, Neutral – 0.4821, Positive – 0.6025  Specificity: Negative – 0.6491, Neutral – 0.9088, Positive – 0.9439  **KNN:**  Sensitivity: Negative – 0.6278, Neutral – 0.7009, Positive – 0.5251  Specificity: Negative – 0.8241, Neutral – 0.6861, Positive – 0.9288  **SVM:**  Sensitivity: Negative – 0.8625, Neutral – 0.5487, Positive – 0.6820  Specificity: Negative – 0.7432, Neutral – 0.9018, Positive – 0.9301  **Multinomial Logistic Regression:**  Sensitivity: Negative – 0.8491, Neutral – 0.5350, Positive – 0.6464  Specificity: Negative – 0.7385, Neutral – 0.8816, Positive – 0.9309  **To summarize the best model for the each of the sentiment in terms of sensitivity and specificity**  Negative Sentiment:   * Decision Tree: Sensitivity of 0.9529 * SVM: Sensitivity of 0.8625 * Multinomial Logistic Regression: Sensitivity of 0.8491   Positive Sentiment:   * Random Forest: Sensitivity of 0.6025 * SVM: Sensitivity of 0.6820 * Multinomial Logistic Regression: Sensitivity of 0.6464   Neutral Sentiment:   * KNN: Sensitivity of 0.7009 * Multinomial Logistic Regression: Sensitivity of 0.5350   Among these models, SVM showed relatively better performance in classifying both negative and positive sentiments. Multinomial Logistic Regression also performed well in classifying negative and positive sentiments. KNN showed better performance in classifying neutral sentiments. |
| --- |
|  |

**MILESTONE 2 : DRAFT REPORT**

This report will summarize the steps taken to analyze the data, the tools and methodologies used and the results of the analysis.

* **What tools and techniques were used and why?**

The following tools and techniques were used for this project:

**Decision Tree**: Decision Trees were used as they are a very popular machine learning model due to their interpretability and simplicity. They are especially useful in classification problems like sentiment analysis. In this case, a decision tree was implemented and tuned using the complexity parameter (CP) to control the size of the tree. The tree was constructed by splitting on certain features such as “thank”, “hour”, “great”, “delay”, “usairway” etc.

**Random Forest**: The Random Forest model was used as it’s an ensemble learning method that operates by constructing multiple decision trees at training time and outputting the class that is the mode of the classes of the individual trees. This method can improve the predictive accuracy and control over-fitting compared to a single decision tree. The random forest model was implemented with 20 trees and used a random subset of 33 variables for each split.

**K-Nearest Neighbors (KNN)**: KNN was used as it’s a simple yet effective algorithm for classification problems. It’s a type of instance-based learning, where the function is only approximated locally, and all computation is deferred until function evaluation. The KNN model was implemented after performing hyperparameter tuning using grid search and 5-fold cross-validation to find the optimal number of neighbors.

**Support Vector Machine (SVM)**: SVM was used as it’s effective in high dimensional spaces and is versatile as different Kernel functions can be specified for the decision function. It’s particularly suited for text classification tasks such as sentiment analysis. In this case, a linear SVM was used as it performs well on large datasets and trains faster than other kernels.

**Multinomial Logistic Regression:** Multinomial logistic regression models the relationship between the independent variables (features) and the multiple categories of the dependent variable (sentiment). The model estimates the probabilities of each category and selects the one with the highest probability as the predicted sentiment. The implementation of multinomial logistic regression involved fitting the model to the training data, using the independent variables (features) to predict the sentiment categories. The coefficients associated with each feature were estimated, indicating their impact on the probability of each sentiment category. The model’s performance was evaluated using various evaluation metrics such as accuracy, precision, recall, and F1-score for each sentiment category. These metrics provided insights into the model’s ability to accurately classify sentiments across all categories.

By incorporating multinomial logistic regression into the project, our team aimed to leverage its ability to handle multi-class classification and provide a robust and accurate sentiment analysis solution.

**Data Preprocessing**: This included splitting the dataset into a training and testing set, removing the sentiment column for model training and testing, and normalizing the data using centering and scaling techniques. This was done to ensure the data is in a suitable format and scale for the machine learning algorithms.

**Hyperparameter Tuning**: This was done using grid search and cross-validation methods to find the optimal parameters for the machine learning models. This helps to improve the performance of the models.

**Model Evaluation**: Techniques such as calculating accuracy, generating confusion matrices, and calculating sensitivity and specificity were used to assess the performance of the models. These metrics provide a comprehensive view of how well the models are performing.

The two best models are Random Forest and the Support Vector Machine (SVM) models.

**Random Forest**: The Random Forest model performed significantly better than the single Decision Tree model, with an accuracy of 74%, compared to the Decision Tree’s 67%. Random Forest models tend to perform better because they aggregate the results of many decision trees, each trained on a random subset of the data. This ensemble technique reduces the risk of overfitting and generally results in a more robust model. The Random Forest model’s ability to correctly identify negative and positive cases was also significantly high (88.08% and 57.95% respectively), although it had a higher misclassification rate for neutral tweets.

**Support Vector Machine (SVM)**: The SVM model also showed good performance, with an accuracy of 76.87% and an F1 score of 0.857. SVM models are especially effective in high-dimensional spaces, making them a good choice for text classification tasks like sentiment analysis. The model’s ability to detect negative sentiment was especially good, with a sensitivity of 0.86. It also performed well in detecting positive sentiment with a sensitivity of 0.68.

Therefore, based on the results of the accuracy and the F1 score, the Random Forest and SVM models appear to have been the most effective at classifying sentiment in this dataset.

* **What were the results of the techniques? What were the new insights?**

**The results of the techniques used in the analysis were as follows:**

**Text Pre-processing:** The text-preprocessing techniques helped to clean and transform the raw text data into a format that can be analyzed using machine learning techniques. The steps included converting the text data to UTF-8 encoding, creating a Corpus object, converting all text to lowercase, removing punctuation and numbers, removing stop words and airline-specific terms, reducing each word to its base form, creating a TermDocumentMatrix, and removing sparse terms.

**Decision tree:** The decision tree model was able to achieve an accuracy of 67% and an F1 score of 0.79 in predicting the sentiment of tweets. The most important variable in the model was “thank”, and the tree had two levels with two terminal nodes. However, the model was unable to provide any classification for the neutral class, which limited its usefulness.

**Random forest**: The random forest model was able to achieve an accuracy of 75% and an F1 score of 0.84 in predicting the sentiment of tweets. The most important variables in the model were “negative\_reason”, “negative\_words”, “positive\_words”, and “neutral\_words”. The model was able to provide some classification for the neutral class, which was an improvement over the decision tree model.

**K-Nearest Neighbors:** The KNN model achieved an accuracy of 63.75% on the test dataset, which is slightly higher than the no-information rate of 62.91%. The optimal number of neighbors was 9 which was achieved after performing hyperparameter tuning using grid search and 5-fold cross-validation. However, the p-value of 0.1818 suggests that the model’s performance is not significantly better than just predicting the majority class. The model’s performance metrics for each sentiment class revealed that the model had a higher accuracy in predicting negative and neutral sentiment, but struggled with positive sentiment. The confusion matrix also highlighted areas where the model could be improved, particularly for the neutral sentiment class, which had a low precision value.

**Support Vector Machine:** The SVM model was able to achieve an accuracy of 77% and an F1 score of 0.86 in predicting the sentiment of tweets. The sensitivity and specificity values for each sentiment class reveal that the model is best at detecting negative sentiment, with a high sensitivity of 0.86, while neutral sentiment is the hardest to detect, with a low sensitivity of 0.54. The model performs well in detecting positive sentiment, with a sensitivity of 0.68 and a high specificity of 0.90.

The model used a linear kernel and a C-classification type. The model was able to provide some classification for the neutral class, but it still had some limitations in accurately distinguishing between the different sentiment classes.

These results suggest that the SVM model is well-suited for predicting the sentiment of airline tweets and can be useful for analyzing customer feedback and improving airline services. However, further work can be done to improve the model’s performance for detecting neutral sentiment, which can be a challenging class to predict accurately.

**Multinomial Logistic Regression :** We used multionomial logistic regression instead of normal one because outcome variable (in this case, the sentiment category) has more than two categories – three sentiment categories: negative, neutral, and positive.

The model achieved an overall accuracy of 0.7516, indicating that it correctly predicted the sentiment category for approximately 75.16% of the instances. The 95% confidence interval for the accuracy is (0.7353, 0.7673).

The p-value is less than 0, indicating that the model performs significantly better in predicting the most frequent sentiment category. Looking at the statistics by class, the model performed better at predicting negative and positive sentiment than neutral sentiment. The sensitivity of the model was highest for negative sentiment. This suggests that the model effectively captured patterns and features associated with negative sentiment expressions.

**Overall, the new insights gained from the analysis were that:**

* The frequency distribution of words in the dataset, which was visualized in a bar plot . This revealed the most common words used in the dataset and provided a general understanding of the language used by Twitter users when expressing their sentiment towards airlines. Words such as unit , usairway, amercianair, southwestair, jetblu , thank hour, cancel help etc were the most common words with a frequency more than 500.
* The most important variables in predicting sentiment were “thank”, “negative\_reason”, “negative\_words”, “positive\_words”, and “neutral\_words”. These variables could be used to improve the accuracy of sentiment analysis models.
* One insight from the KNN model is that normalization using centering and scaling techniques can improve the performance of distance-based algorithms like KNN. Another insight is that hyperparameter tuning using grid search and cross-validation can help to find the optimal value for the number of neighbors (‘k’) in the KNN algorithm.
* Different techniques had different strengths and weaknesses in predicting sentiment. The random forest and SVM models were generally more accurate and effective than the decision tree and KNN models.
* The models still had some limitations in accurately distinguishing between the different sentiment classes, especially in the case of the neutral class. This suggests that there is still room for improvement in sentiment analysis techniques.
* **How did this help answer your questions? If your questions changed, why?**

The EDA and machine learning models used in this project were aimed at answering the business questions related to sentiment analysis of tweets about airlines. These models helped us identify which airlines consistently perform better or worse than others and find areas that airlines can improve upon.

The models’ performance, including the accuracy, confusion matrix, sensitivity, and specificity, helped evaluate the effectiveness of the models in classifying sentiment in the dataset. This information can be useful for airlines to understand their customers’ perceptions and address any issues that may affect customer satisfaction.

No our business questions did not change but there are some questions that would have been interesting to explore– such as analyzing how sentiment in tweets about airlines differs based on demographic groups or region and examining the impact of sentiment on social media on customer loyalty, and brand reputation for airlines.

For further analysis, it may be beneficial to incorporate additional variables such as customer satisfaction ratings on social media, and demographic details (age, gender, occupation, etc) in the dataset. Data sources such as customer profiles and surveys could also be used. This would enable a more targeted analysis of sentiment customer behavior and preferences, which could help airlines prioritize their efforts accordingly.

**CONCLUSION**

In conclusion, sentiment analysis is a valuable tool for social media companies to monitor and understand how their brand or products are being perceived by customers and the public. The analysis of the Twitter US Airline Sentiment dataset has provided insights into the sentiment distribution of different airlines, the most common negative reasons for tweets, and the most frequently used words in tweets related to airlines. These insights can help airlines improve their services, radiss customer issues, and enhance the customer experience. Additionally, analyzing the volume of tweets over time and across different airlines can provide valuable information for businesses to identify patterns and trends and determine the best times to engage with their followers. Overall, sentiment analysis can help social media companies make data-driven decisions about how to improve customer satisfaction and engagement, ultimately leading to increased customer loyalty and improved business outcomes.

* The dataset contains 14,640 instances of tweets about US airlines.
* Among these tweets, 62.7% were negative, 21.2% were neutral, and 16.1% were positive.
* United Airlines had the highest number of total tweets, with 3,032 tweets, followed by US Airways and American Airlines.
* However, United Airlines also had the highest number of negative tweets, with 2,633 negative tweets.
* The most common negative reason for tweets was “Customer Service Issue,” with 2,904 tweets, followed by “Late Flight” with 1,660 tweets.
* The word “flight” was the most frequently used word in tweets related to airlines, with a count of 5,904, followed by “united” with a count of 3,263.
* The Ridge chart showed that the average text length for negative tweets was 118 characters, compared to 96 characters for neutral tweets and 98 characters for positive tweets.
* After EDA, we performed the text pre-processing of the data using the TF-IDF model approach. This included cleaning, formatting, and transforming raw text data into a format that is suitable for analysis. We also calculated and visualized the most frequent words used in the tweets having a frequency of more than 500.
* Before starting with the model building, we split the data into training and testing sets with 80:20 ratio and calculated the baseline accuracy for both the training and test sets.

We started with Decision Tree model which achieved an accuracy of 67%, however, it failed to provide any classification for the neutral sentiment class. The F1 score for the decision tree model is 79%.

* We then applied the Random Forest model to predict the airline sentiment with 20 trees which achieved an accuracy of 75%. The F1-score for the Random first model is 84%. Also, the model correctly predicted more true negatives (i.e. 1588 Negative tweets) as compared to true positives (i.e. 277 Positive tweets) and least for neutral tweets (only 266 true positives).
* Further, we applied the distance-based algorithm model KNN (K- Nearest Neighbour) with k=9 as the optimal number of neighbors for the model which resulted in an accuracy of 62%. The F1-score for the KNN model is 72%.

Also, the model correctly predicted more true negatives (i.e. 1206 Negative tweets) as compared to true positives (i.e. 254 Positive tweets) but did better for neutral tweets 367 true positives).

* Further, we implemented the ensemble learning model – Support Vector Machine (SVM) which achieved an accuracy of 77% indicating a good performance in predicting the tweet sentiment. The F1-score for the SVM model was the highest around 86%.
* Lastly , we implemented the multinomial logistic regression model which demonstrated an overall accuracy of 75.16%. The model performed better in predicting negative and positive sentiments compared to neutral sentiment. The sensitivity analysis revealed that the model had the highest accuracy in identifying instances with negative sentiment.The F1-score for the Multinomial logistic radientn model was the around 85%.
* To summarize , comparing all the models based on accuracy, the SVM model achieved the highest accuracy of 76.8%, followed by Multinomial Logistic Regression (75.16%), Random Forest (75.05%) , KNN (62.56%), and Decision Tree (67.02%).

In terms of F1-score, which considers both precision and recall, the SVM model outperformed the other models with an F1-score of 0.8565., Multinomial Logistic Regression had an F1-score of 0.8477, Random Forest had an F1-score of 0.8416 ,KNN had an F1-score of 0.7251, and Decision Tree had an F1-score of 0.7907.

* Overall, the SVM model demonstrated the highest performance in terms of both accuracy and F1-score, making it the top-performing model in this sentiment analysis task. It effectively captured the complex patterns in the data and provided reliable predictions. Random Forest and Multinomial Logistic Regression also performed reasonably well, while KNN and Decision Tree showed slightly lower performance.

| **APPENDIX** |
| --- |

| # Load Packages & Libraries library(tidyverse) library(caret) library(highcharter) library(RcolorBrewer) library(ggcorrplot) library(corrplot) library(psych) library(dplyr) library(ggplot2) library(gtools) library(ggfortify) library(Ggally) library(readr) library(readxl) library(knitr) library(modelr) library(scales) library(sqldf) library(car) library(ggpubr) library(grid) library(gridExtra) library(lattice) library(hrbrthemes) library(ISLR) library(caret) library(pROC) library(psych) library(olsrr) library(naniar) library(DataExplorer) library(formattable) library(glmnet) library(Metrics) library(Mlmetrics) library(Ggally) require(grid) library(lubridate) library(gganimate) library(babynames) library(hrbrthemes) library(gifski) library(plotly) library(leaflet) library(radi) library(tidytext) require(grid) library(treemapify) library(ggridges) library(tidyr) library(plotly) library(viridis) library(hrbrthemes) library(gganimate) library(wordcloud) library(magrittr)  # Importing the dataset  Tweets.data <- read.csv(“Tweets.csv”,header=TRUE,sep=”,”)  # Understanding the dataset str(Tweets.data) summary(Tweets.data) glimpse(Tweets.data) headTail(Tweets.data)  # Data Pre-Processing & Data Cleaning  # Checking data is clean? colSums(is.na(Tweets.data)) # check & Returns the number of missing values in each column sum(is.na(Tweets.data)) # Counts missing values in entire data frame colSums(Tweets.data==0) #Using colSums function to find the total number of Zero records in each column plot\_missing(Tweets.data, title=”Missing Data Profile”,geom\_label\_args = list(“size” = 2, “label.padding” = unit(0.1, “lines”)))  # Dropping tweet\_id column (numeric) drop1<-c(“tweet\_id”) Tweets.data<-Tweets.data[,!(names(Tweets.data) %in% drop1)]  # Replacing NA values with the mean of the column Tweets.data$negativereason\_confidence[is.na(Tweets.data$negativereason\_confidence)]<-round (mean(Tweets.data$negativereason\_confidence,na.rm=TRUE),2)  # Checking for duplicated rows and removing them duplicated(Tweets.data) anyDuplicated(Tweets.data) Tweets.data<-Tweets.data[!duplicated(Tweets.data), ]  # Seperating date and time into two columns Tweets.data<-Tweets.data %>%  separate(tweet\_created, c(“date”, “time”), “ “)  # Hour from time Tweets.data$hour <- hour(hms(Tweets.data$time))  # Replacing hours to Morning, Afternoon,Evening and Night # 0-5 night # 6-11 morning # 12-17 afternoon # 18-24 evening Tweets.data <- Tweets.data %>%   mutate(hour = ifelse(hour %in% c(0:5), “Night”,  ifelse(hour %in% c(6:11), “Morning”,  ifelse(hour %in% c(12:17), “Afternoon”, “Evening”))))  # Creating day column Tweets.data$day<-Tweets.data$date Tweets.data$day[Tweets.data$date == ‘2015-02-16’] <- ‘Monday’ Tweets.data$day[Tweets.data$date == ‘2015-02-17’] <- ‘Tuesday’ Tweets.data$day[Tweets.data$date == ‘2015-02-18’] <- ‘Wednesday’ Tweets.data$day[Tweets.data$date == ‘2015-02-19’] <- ‘Thursday’ Tweets.data$day[Tweets.data$date == ‘2015-02-20’] <- ‘Friday’ Tweets.data$day[Tweets.data$date == ‘2015-02-21’] <- ‘Saturday’ Tweets.data$day[Tweets.data$date == ‘2015-02-22’] <- ‘Sunday’ Tweets.data$day[Tweets.data$date == ‘2015-02-23’] <- ‘Monday’ Tweets.data$day[Tweets.data$date == ‘2015-02-24’] <- ‘Tuesday’  # Relocating the columns Tweets.data<-Tweets.data %>% relocate(day,.after = date) Tweets.data<-Tweets.data %>% relocate(hour,.after = time)  # Changing the datatypes Tweets.data$airline\_sentiment<-as.factor(Tweets.data$airline\_sentiment) Tweets.data$airline<-as.factor(Tweets.data$airline) Tweets.data$day<-as.factor(Tweets.data$day) Tweets.data$hour<-as.factor(Tweets.data$hour) str(Tweets.data)  # Descriptive Statistics for entire dataset formattable(describe(Tweets.data),   caption = “Descriptive statistics summary of the Twitter US Airline Sentiment Dataset”)  # Histogram Distribution  # Airline Sentiment Confidence ggplot(Tweets.data, aes(x=airline\_sentiment\_confidence)) +   geom\_histogram(color=”black”, fill=”orange”, position=”identity”)+  labs(title=”Airline Sentiment Confidence Distribution”,x=”Airline Sentiment Confidence”, y = “Count”)  # Negative Reason Confidence ggplot(Tweets.data, aes(x=negativereason\_confidence)) +   geom\_histogram(color=”black”, fill=”orange”, position=”identity”)+  labs(title=”Negative Reason Confidence Distribution”,x=”Negative Reason Confidence”, y = “Count”)  # Retweet Count ggplot(Tweets.data, aes(x=retweet\_count)) +   geom\_histogram(color=”black”, fill=”orange”, position=”identity”)+  labs(title=”Retweet Count Distribution”,x=”Retweet Count”, y = “Count”)  # Correlation Matrix corr<-Tweets.data %>% select(airline\_sentiment\_confidence,negativereason\_confidence,retweet\_count) corr\_matrix <- cor(corr) corr\_matrix  ggcorrplot(corr\_matrix, hc.order = TRUE,   type = “lower”,   lab = TRUE,   lab\_size = 3.5,   colors = c(“#6D9EC1”, “white”, “#E46726”),   title = “Correlation Plot of Selected Variables”)  # Data Profiling Report create\_report(Tweets.data)  # Data Visualizations # Graph 1 – Sentiment analysis by airline df1<-Tweets.data %>% group\_by(airline, airline\_sentiment) %>% summarize(count=n())  In2 <- hchart(df1, ‘column’,  hcaes(x = ‘airline’, y = ‘count’, group = ‘airline\_sentiment’),  stacking = “normal”) %>%  hc\_colors(c(“#CD0000”, “#EEE8CD”, “#698B69”)) %>%  hc\_xAxis(title = “Airline”) %>%  hc\_yAxis(title = “Count”) %>%  hc\_title(text = “Sentiment Analysis by Airline”) In2  # Graph 2 interactive – Tweet Count by Day and Hour df2 <- Tweets.data %>% group\_by(day,hour) %>% summarize(count=n())  In2 <- hchart(df2, ‘column’,  hcaes(x = ‘day’, y = ‘count’, group = ‘hour’)) %>%  hc\_colors(c(“#B0E2FF”, “#FF8247”, “#FDE725”, “#00008B”)) %>%  hc\_xAxis(title = “Day”) %>%  hc\_yAxis(title = “Count”) %>%  hc\_title(text = “Tweet Count by Day and Hour”) In2  # Graph 3 interactive – Negative Reason Counts by Airline df3 <- Tweets.data %>%  filter(!is.na(negativereason) & negativereason != “”) %>%  group\_by(negativereason, airline) %>%  summarize(count = n()) %>%  filter(count > 0) %>%  arrange(desc(count))  In3 <- hchart(df3, ‘bar’, hcaes(x = negativereason, y = count, group = airline)) %>%  hc\_colors(c(“#1874CD”, “#B22222”, “#FFD700”, “#00BFFF”,”#8B8682”,”#FF0000”)) %>%  hc\_xAxis(title = “Negative Reasons”) %>%  hc\_yAxis(title = “Count”) %>%  hc\_title(text = “Negative Reason Counts by Airline”) In3  # Graph 4 – Timeline of Daily Cumulative Tweets  dailyTweetscum <- Tweets.data %>% group\_by(date) %>% dplyr::summarise(count = n()) %>%  mutate(cuml = cumsum(count))  p1<- ggplot(data=dailyTweetscum , aes(x = date, y = cuml)) +  geom\_point(size=2.0 ,color = “skyblue”)+  theme\_classic()+  ggtitle(‘Daily Cumulative Tweets’) +  scale\_y\_continuous(labels = scales::comma, breaks = scales::pretty\_breaks(n=7)) +  theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1))+  theme(panel.grid.major = element\_blank(), panel.grid.minor = element\_blank(),  text=element\_text(size=12, family=”Comic Sans MS”, color= “black”))  ggplotly(p1)  # Graph 5 – Density graph of Negative tweets  negativeTweets <- Tweets.data %>% filter(airline\_sentiment==”negative”) negativeTweets <- negativeTweets %>% group\_by(airline , date) %>% dplyr::summarise(count = n())   p2 <- ggplot(negativeTweets, aes(x = count, fill = airline)) +  geom\_density(alpha = 0.5) +  ggtitle(“Distribution of Negative Tweets by Airline”) +  theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust = 1)) p2  # Graph 6 – Timeline of Daily Tweets for Each Airline dailyTweets <- Tweets.data %>% group\_by(airline,date) %>% dplyr::summarise(count = n()) dailyTweets$date <- as.Date(dailyTweets$date)  p3 <-dailyTweets %>% ggplot(aes(x = date, y = count,   group = airline,  color = airline)) +  theme\_bw()+  geom\_line() +  geom\_point() +  ggtitle(“Daily Tweets per Airline”) +  theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1)) +  transition\_reveal(date)  animate(p3, renderer = gifski\_renderer()) rstudioapi::viewer(“http://localhost:24784/session/filecd4843c75819.gif")  anim\_save(filename = “Animated.gif”)  # Graph 7 – Location Wise Tweets – Circular Bar chart location <- Tweets.data %>% group\_by(tweet\_location) %>%  dplyr::summarise(count=n()) %>% arrange(desc(count)) %>% filter(!is.na(tweet\_location) & tweet\_location != “”) %>% top\_n(10) location  ggplot(location, aes(tweet\_location, count, fill = tweet\_location)) +  geom\_col(position = “dodge”) +  coord\_polar() +  geom\_text(aes(label = count), vjust = -0.5, size = 3) +  scale\_fill\_viridis\_d() +  ggtitle(“Circular Bar chart- Top 10 Location Wise Tweets”) +  xlab(“Location”) +  theme(axis.text.x = element\_text(hjust = 1))  # Graph 8 – Treemap of Negative reasons  data = Tweets.data %>% group\_by(negativereason) %>% summarise(count = n())%>% filter(!is.na(negativereason) & negativereason != “”)  ggplot(data, aes(area = count, fill = negativereason, label = negativereason)) +  geom\_treemap() +  scale\_fill\_brewer(name = “Negative Reasons”,palette = “Set3”) +  geom\_treemap\_text(family = “Comic Sans MS”, colour = “black”, place = “centre”, reflow = T) +  theme\_void() +  ggtitle(“Treemap – Negative reason”) +  theme(plot.title = element\_text(hjust = 0.5, size = 18, family = “Comic Sans MS”, color = “black”)) +  guides(fill = FALSE) +  coord\_fixed(ratio = 0.75)  # Graph 9 – Pie charts with Distribution of Sentiments by Airline sentiment\_summary <- Tweets.data %>%  group\_by(airline, airline\_sentiment) %>%  summarise(n = n()) %>%  mutate(percent = n / sum(n) \* 100) pie\_charts <- ggplot(sentiment\_summary, aes(x = “”, y = percent, fill = airline\_sentiment)) +  geom\_bar(stat = “identity”, width = 1, color = “white”) +  coord\_polar(theta = “y”) +  facet\_wrap(~ airline, nrow = 2) +  theme\_void() +  labs(x = NULL, y = NULL, fill = NULL, title = “Distribution of Sentiments by Airline”) +  theme(plot.title = element\_text(hjust = 0.5, size = 18, family = “Comic Sans MS”, color = “black”),  strip.text = element\_text(size = 16, family = “Comic Sans MS”, color = “black”),  text = element\_text(family = “Comic Sans MS”, color = “black”)) +  scale\_fill\_manual(name = “Sentiment”,  values = c(“#CD0000”, “#EEE8CD”, “#698B69”),  labels = c(“Negative”, “Neutral”,”Positive”)) +  geom\_text(aes(label = paste0(round(percent), “%”)), position = position\_stack(vjust = 0.5))  print(pie\_charts)  # Graph 10 – Wordcloud of words used in tweets Tweets.data %>%  select(text) %>%  unnest\_tokens(word, text) %>%  dplyr::count(word, name = “n”) %>%  arrange(desc(n)) %>%  with(wordcloud(word, n, max.words = 100, colors = brewer.pal(8, “Dark2”)))  # Graph 11 – Top 20 words in Tweets colour <- c(  “#E31A1C”,  “green4”,  “#6A3D9A”,   “#FF7F00”, “gold1”,  “skyblue2”,”dodgerblue2”, “#FB9A99”,  “palegreen2”,  “#CAB2D6”,   “#FDBF6F”,   “khaki2”,  “maroon”, “green1”, “orchid1”,”deeppink1”, “steelblue4”,  “darkturquoise”, “yellow3”,  “brown” )  bar <- Tweets.data %>%  unite(text\_combined, text, sep = “ “) %>%  select(text\_combined) %>%  unnest\_tokens(word, text\_combined) %>%  filter(!grepl(“^[0-9]+$”, word)) %>%  dplyr::count(word, sort = TRUE) %>%  top\_n(20) %>%  ggplot(aes(x = word, y = n)) +  geom\_col(color = “black”, fill = colour) +  coord\_flip() +  labs(x = “Words”, y = “Frequency”, title = “Top 20 Most Frequent Words in Tweets”) +  theme\_bw() +  theme(text = element\_text(family = “Comic Sans MS”, color = “black”))  ggplotly(bar)  # Graph 12 - Ridge chart – Tweet Text Length by Sentiment Tweets.data$text\_length <- nchar(Tweets.data$text) Tweets.data %>%  ggplot(aes(x = text\_length, y = airline\_sentiment, fill = airline\_sentiment)) +  geom\_density\_ridges(scale = 3, rel\_min\_height = 0.01,alpha=0.5) +  scale\_fill\_manual(name = “Sentiment”,labels = c(“Negative”, “Neutral”,”Positive”),values = c(“#CD0000”, “#EEE8CD”, “#698B69”)) +  labs(x = “Text Length”, y = “Sentiment”, title = “Tweet Text Length by Sentiment”) +  theme\_minimal() +  theme(text = element\_text(family = “Comic Sans MS”, color = “black”))  # Text Pre-processing  # Convert Tweets data to UTF-8 encoding  tweets\_utf8 <- iconv(Tweets.data$text, to = “UTF-8”)  # Create a Corpus object from the text data  corpus\_text <- Corpus(VectorSource(tweets\_utf8))  # Inspect the first five documents in the Corpus object  inspect(corpus\_text[1:5])  # Convert all text to lowercase  corpus\_lc <- tm\_map(corpus\_text, tolower)  # Remove all punctuation from the text  corpus\_np <- tm\_map(corpus\_lc, removePunctuation)  # Remove all numbers from the text  corpus\_nn <- tm\_map(corpus\_np, removeNumbers)  # Remove common English words from the text  corpus\_sw <- tm\_map(corpus\_nn, removeWords, stopwords(‘english’))  # Remove additional words such as airline-specific terms from the text  corpus\_asw <- tm\_map(corpus\_sw, removeWords, c(‘flight’,’get’,’plane’,’flights’,’flightl’,’don’t’, ‘day’, ‘im’, ‘cant’, ‘can’, ‘now’, ‘just’,’will’, ‘don’t’, ‘ive’, ‘got’, ‘much’))  # Remove any extra white space from the text  corpus\_ws <- tm\_map(corpus\_asw, stripWhitespace)  # Reduce each word to its base form  corpus\_stem <- tm\_map(corpus\_ws, stemDocument)  # Create a TermDocumentMatrix from the pre-processed text  tdm <- TermDocumentMatrix(corpus\_stem)  # Remove sparse terms from the TermDocumentMatrix  tdm\_clean <- removeSparseTerms(tdm, sparse = 0.999)  # Convert the resulting matrix to a data frame  tdm\_df <- as.data.frame(as.matrix(tdm\_clean))  # Compute the frequency of each word in the Corpus object  word\_freq <- rowSums(tdm\_df)  # Select words with a frequency of more than 500  freq\_words <- subset(word\_freq, word\_freq >= 500)  # Sort the selected words in decreasing order  sorted\_freq\_words <- sort(freq\_words, decreasing = T)  # Create a bar plot of the selected words with their frequency  barplot(sorted\_freq\_words, col = rainbow(38,alpha=0.7), las = 2,  main = “Words with frequency more than 500”,  xlab = “Words”, ylab = “Frequency”)  # Create a DocumentTermMatrix from the pre-processed text  dtm <- DocumentTermMatrix(corpus\_stem)  # Remove sparse terms from the DocumentTermMatrix  dtm\_clean <- removeSparseTerms(dtm, 0.999)  # Convert the resulting matrix to a data frame  dtm\_df <- as.data.frame(as.matrix(dtm\_clean))  # Add the airline sentiment labels as a factor variable to the data frame  dtm\_df$sentiment <- Tweets.data$airline\_sentiment  # Rename any duplicate variables in the training set  colnames(dtm\_df)[duplicated(colnames(dtm\_df))] <- paste0(colnames(dtm\_df)[duplicated(colnames(dtm\_df))], “\_2”)  # Convert the airline sentiment labels to a factor variable  dtm\_df$sentiment <- as.factor(dtm\_df$sentiment)  # Split the data into a training set and a testing set  set.seed(222)  split <- sample(2, nrow(dtm\_df), prob = c(0.8,0.2), replace = TRUE)  train\_set <- dtm\_df[split == 1,]  test\_set <- dtm\_df[split == 2,]  # Compute the baseline accuracy for the training set  train\_set\_baseline\_acc <- prop.table(table(train\_set$sentiment))  # Compute the baseline accuracy for the testing set  test\_set\_baseline\_acc <- prop.table(table(test\_set$sentiment))  # Display the baseline accuracies for both sets  train\_set\_baseline\_acc  test\_set\_baseline\_acc  # Removing Duplicacy  # Display column names of train\_set data frame  names(train\_set)  # Rename any duplicate variables to ensure unique names  colnames(train\_set)[duplicated(colnames(train\_set))] <- paste0(colnames(train\_set)[duplicated(colnames(train\_set))], “\_2”)  names(test\_set) <- names(train\_set)  # Decision tree  # Fit a decision tree classifier using the training set  dt\_classifier <- rpart(sentiment ~ ., data = train\_set, method = “class”)  # Plot the decision tree  rpart.plot(dt\_classifier)  # Display a summary of the decision tree  summary(dt\_classifier)  # Predict the airline sentiment using the decision tree classifier and the testing set  dt\_predict <- predict(dt\_classifier, newdata = test\_set, type = “class”)  # Compute the accuracy of the prediction on the testing set  dt\_accuracy <- accuracy(dt\_predict, test\_set$sentiment)  ################# Random forest #################  rf\_model <- randomForest(sentiment ~ ., data = train\_set, ntree = 20)  print(rf\_model)  #Predict on Train set  rf\_predict1 <- predict(rf\_model, newdata=train\_set, type=”class”)  accuracy(rf\_predict1,train\_set$airline\_sentiment)  # Predict on test data  rf\_predict2 <- predict(rf\_model, newdata=test\_set , type=”class”)  accuracy(rf\_predict2,test\_set$airline\_sentiment)  # Evaluate model performance  conf\_mat <- confusionMatrix(table(rf\_predict2, test\_set$airline\_sentiment))  conf\_mat  ######################## KNN ###################################  # Load required libraries  library(caret)  # Split the data into a training set and a testing set  set.seed(222)  split <- sample(2, nrow(dtm\_df), prob = c(0.8, 0.2), replace = TRUE)  train\_set <- dtm\_df[split == 1, ]  test\_set <- dtm\_df[split == 2, ]  # Prepare the training set and test set without the sentiment column  train\_data <- train\_set[, -ncol(train\_set)]  test\_data <- test\_set[, -ncol(test\_set)]  # Prepare the target variable for the training set  train\_labels <- train\_set$airline\_sentiment  # Normalize the data  normalize <- preProcess(train\_data, method = c(“center”, “scale”))  train\_data\_norm <- predict(normalize, train\_data)  test\_data\_norm <- predict(normalize, test\_data)  # Create a data frame for training data and labels  train\_df <- cbind(train\_data\_norm, airline\_sentiment = train\_labels)  # Define the hyperparameter search grid  k\_grid <- expand.grid(k = seq(5, 30, by = 1))  control <- trainControl(method = “cv”, number = 5, search = “grid”)  # Train the model using the training set with hyperparameter tuning  knn\_model <- train(airline\_sentiment ~ .,  data = train\_df,  method = “knn”,  trControl = control,  tuneGrid = k\_grid,  preProcess = c(“center”, “scale”))  # Print the best hyperparameter values  print(knn\_model$bestTune)  # Predict the test set labels  knn\_predict\_test <- predict(knn\_model, newdata = test\_data\_norm)  # Compute the accuracy for the test set  test\_accuracy <- mean(knn\_predict\_test == test\_set$airline\_sentiment)  print(test\_accuracy)  # Evaluate model performance  conf\_mat1 <- confusionMatrix(knn\_predict\_test, test\_set$airline\_sentiment)  conf\_mat1  ######################### SVM #################################  library(e1071)  svm\_model<- svm(airline\_sentiment ~ .,  data = train\_set,  type = “C-classification”,  kernel = “linear”,  scale = FALSE)  print(svm\_model)  # predict on the testing set  svm\_pred <- predict(svm\_model, newdata = test\_set)  plot(svm\_model, train\_set$airline\_sentiment ~ .)  # evaluate the performance of the model  conf\_mat4 <- confusionMatrix(svm\_pred, test\_set$airline\_sentiment)  conf\_mat4  svm\_f1\_score <- F1\_Score(test\_set$airline\_sentiment, svm\_pred)  svm\_f1\_score  ######################## COMPARISON WORD CLOUD ##########################  # Subset the sentiments  pos\_tweets <- subset(Tweets.data$text, Tweets.data$airline\_sentiment == “positive”)  neg\_tweets <- subset(Tweets.data$text, Tweets.data$airline\_sentiment == “negative”)  neu\_tweets <- subset(Tweets.data$text, Tweets.data$airline\_sentiment == “neutral”)  # Combine and label the sentiment terms  all\_terms <- c(positive = paste(pos\_tweets, collapse = “ “),  negative = paste(neg\_tweets, collapse = “ “),  neutral = paste(neu\_tweets, collapse = “ “))  # Create a corpus and term-document matrix  all\_corpus <- Vcorpus(VectorSource(all\_terms))  all\_tdm <- TermDocumentMatrix(all\_corpus, control = list(removePunctuation = TRUE,  removeNumbers = TRUE,  stemDocument = TRUE,  tolower = TRUE,  stopwords = c(‘flight’, ‘get’, ‘plane’, ‘flights’, ‘flightl’, ‘don’t’, ‘day’, ‘im’, ‘cant’, ‘can’, ‘now’, ‘just’, ‘will’, ‘don’t’, ‘ive’, ‘got’, ‘much’),  stopwords = stopwords(‘english’)))  all\_tdm\_m <- as.matrix(all\_tdm)  colnames(all\_tdm\_m) <- c(“positive”,”negative”, “neutral”)  # Generate the word cloud with labels    comparison.cloud(all\_tdm\_m, max.words = 100, colors = c(“green”, “red”, “blue”),  scale = c(5, 1), title.size = 1.5, title.colors = “black”, rot.per = 0.35,  random.order = FALSE, main = “Sentiment Comparison Word Cloud”)  ########################## MULTINOMIAL LOGISTIC REGRESSION #############################  install.packages(“VGAM”)  library(VGAM)  lg\_classifier <- vglm(airline\_sentiment ~ ., data = train\_set, family = multinomial)  lg\_classifier <- multinom(airline\_sentiment ~., data=train\_set , MaxNWts =11676)  print(lg\_classifier)  #Predict on the testing set#  lg\_pred <- predict(lg\_classifier, newdata = test\_set)  # evaluate the performance of the model  conf\_mat5 <- confusionMatrix(lg\_pred, test\_set$airline\_sentiment)  conf\_mat5  lg\_f1\_score <- F1\_Score(test\_set$airline\_sentiment, lg\_pred)  lg\_f1\_score  conf\_mat5\_plot <- ggplot(conf\_mat5$table, aes(Prediction, Reference, fill = Freq , label=Freq)) +  geom\_tile() + geom\_text(color=”white”) +scale\_fill\_gradientn(colors=my\_colors) +  theme\_minimal() + labs(title = “Confusion Matrix”, x = “Predicted”, y = “Actual”)  conf\_mat5\_plot  ################# Calculate accuracy and F1 score for each model #######################  dt\_accuracy <- confusionMatrix(dt\_predict, test\_set$airline\_sentiment)$overall[‘Accuracy’]  rf\_accuracy <- confusionMatrix(rf\_predict2, test\_set$airline\_sentiment)$overall[‘Accuracy’]  knn\_accuracy <- mean(knn\_predict\_test == test\_set$airline\_sentiment)  svm\_accuracy <- confusionMatrix(svm\_pred, test\_set$airline\_sentiment)$overall[‘Accuracy’]  lg\_accuracy <- confusionMatrix(lg\_pred, test\_set$airline\_sentiment)$overall[‘Accuracy’]  dt\_f1\_score <- F1\_Score(test\_set$airline\_sentiment,dt\_predict)  rf\_f1\_score <- F1\_Score(test\_set$airline\_sentiment,rf\_predict2)  knn\_f1\_score <- F1\_Score(test\_set$airline\_sentiment, knn\_predict\_test)  svm\_f1\_score <- F1\_Score(test\_set$airline\_sentiment, svm\_pred)  lg\_f1\_score <- F1\_Score(test\_set$airline\_sentiment, lg\_pred)  # Create a table of the results  results <- data.frame(  Model = c(“Decision Tree”, “Random Forest”, “KNN” ,”SVM” ,”Logistic”),  Accuracy = c(dt\_accuracy, rf\_accuracy, knn\_accuracy,svm\_accuracy, lg\_accuracy),  F1\_Score = c(dt\_f1\_score, rf\_f1\_score, knn\_f1\_score ,svm\_f1\_score , lg\_f1\_score)  )  print(results)  #################### Model Comparison ############################  # Melt the data frame to long format for plotting  model\_results\_melt <- reshape2::melt(results, id.vars = “Model”)  model\_results\_melt  # Create a line graph  plot\_results <-model\_results\_melt %>% ggplot(aes(x = Model, y = value,  group = variable,  color = variable)) +  theme\_bw()+  geom\_line() +  geom\_point() +  ggtitle(“Model Comparison”) +  theme(axis.text.x = element\_text(angle = 90, vjust = 0.5, hjust=1))  plot\_results  ##################################################################### |
| --- |
|  |