National College of Ireland

Bachelors of Science (Honours) in Computing 4

Data Analytics

2021/2022

Jordan-Lee Graham

19103310

X19103310@student.ncirl.ie

Predicting Airline Customer Review Scores Based on Sentiment Analysis of Reviews

Technical Report

Contents

Executive Summary

The purpose of this research was to contribute to further understanding of text classification sentiment analysis for use in the implementation of prediction models. Much of the available literature focuses on lexicon-based approaches, which aim to match words in text with words from ‘positive’ or ‘negative’ word dictionaries and apply a binary classification of negative or positive sentiment. In this study, a statistical approach to text classification was taken by using the ‘*Term Frequency – Index Document Frequency*’ (TF-IDF) algorithm to weight words based on their number of occurrences in a review in comparison to all review documents. The TF-IDF features of *100* and *300* were supplied to the most consistent supervised learning model from relevant literature – a ‘*Support Vector Machine*’ (SVM). The model was tested with a variety of hyper parameters in combination with the linear and radial basis function (RBF) kernels. The non-parametric kernel RBF performed best in classifying values in the range *1-10*, with an accuracy of *44%*. Methods were implemented to improve the model to determine the best possible accuracy the multiclass classification model could achieve in comparison to similar studies performing binary classification. The result was a performance of *83%* accuracy on the multiclass classification of ‘*negative*’, ‘*neutral*’ and ‘*positive*’. A precision of *77%* was achieved on the under represented ‘*neutral*’ label due to data imbalances. The best hyper-parameters identified for the best results on each attempt of classification are recorded. A recommendation is drawn for the use of more TF-IDF features alongside a non-parametric kernel for best results for the solution to multiclass classification of sentiment.

1.0 Introduction

1.1. Background

This project was undertaken in order to apply further understanding of sentiment analysis and in particular statistically based approaches in relation to classifying customer feedback. There is much research available on the comparison of different classification-based models where the primary outcome desired is a positive or negative classification of customer review text. The primary goal of this project was to investigate these techniques further by applying them to customer feedback in relation to the aviation industry and use further available data such as score values for different amenities of the customers journey to investigate if these scores could be concluded based on analysis of review text. Through performing this research, a comparison of classification-based approaches can be made to further research into this area.

1.2. Aims

The aim of this research is to analyse airline customer reviews of flights by applying and comparing different approaches commonly used towards sentiment analysis. Another aim for this project is the collection of data for the purpose of adjusting the models used.

Much of the current research focuses on lexicon-based classification solutions by where review text is compared with a predefined dictionary such as the '*WordNet*' library to check for the occurrence of words associated with negative or positive text. This project aims to explore supervised methods and statistical approaches such as TF-IDF. This aim will be achieved by setting the objective of using a numerical representation of review text the study aims to predict customers overall score provided. Highlighted in the report are studies performed by Brooke et al. (Brooke, 2009) and Ye et al. (Qiang Ye, 2009) which focused on binary classification of review sentiment. The study aims to reach similar or better results of these studies using multiclass classification using a Support Vector Machine supervised learning method with a statistical approach to classifying the data. The objective for this will be to continuously apply methods and techniques in order to improve the model's performance and to highlight the necessary steps taken to achieve the end results. Data will be retrieved and reviewed. Data sets will be pre-processed to construct data appropriate for feeding to the SVM. The review text will be applied a numerical value based on the TF-IDF algorithm. The TF-IDF vectors will be provided alongside the overall score for training the SVM. The SVM will then be tested on unseen data. The results will be evaluated and if required the necessary steps to improve the model will be implemented such as hyper-parameter tuning, addition or removal of features etc.

1.3. Technology

The primary technology used to achieve the aims set out for this project is the Python programming language used within the web based integrated development environment (IDE) '*Jupyter Notebook*'. Both of these tools are accessed through the '*Anaconda*', Python distribution platform designed specifically towards data science projects making the management of packages and integrations straight forward and simplistic.

Python was the preferred choice for programming language after some initial tests using R. Both of these languages offer similar packages and libraries as well as R Markdown being a viable substitute as a notebook. The preference for Python over R was decided based on how the languages handled large amounts of data at once. The R language presented problems by relying on stronger PC specifications such as extra cores in order to perform some tasks which ran smoothly in Python without the added code of specifying space usage, making Python the preferred choice.

Jupyter Notebook was chosen specifically for its ability to incorporate markup text which is useful for presentation throughout the code, using markup to highlight and explain next steps in the process and results.

Python also offered a range of libraries purpose built for analysing and visualizing data. The '*Pandas*' and '*NumPy*' libraries were used to handle and convert data read from comma separated value (CSV) files into tabled data. For replicating lexicon approaches and pre-processing of data, the '*Natural Language Tool Kit*' (NLTK) library was used with the libraries built in dictionaries for stop words being used as well as the previously mentioned '*WordNet*' library. For visualizations of graphs and plots the 'itertools', 'collections' and 'matplotlib' libraries were used. The '*Sckikit-Learn*' package was used to import the '*TFidfVectorizer*' library to prepare the data for use in a '*Support Vector Machine*' (SVM) model in which the library '*SVM*' was also imported from the '*sklearn*' package. The '*train_test_split*' library for separating the processed dataset into training and test splits for the model and 'metrics' library for providing metrics of the SVM models performance which were also, imported from the '*Ski Kit Learn*' package.

1.4. Structure

The project is introduced with an explanation of why the project was chosen as well as an outline of the aims of the project and the objectives set out to achieve those aims. The tools and technology required for completing the objectives is highlighted.

The data used for the project is mentioned along with the details of the source of the data, size and method of storage. Attributes of the data is discussed and the findings of any exploratory analysis performed.

The methodology is discussed in reference to the methodologies and techniques applied to the data in order to manipulate and process the data. The purposes of these processes are justified and highlighted. The libraries and packages required for these processes are also outlined so that the work can be reproduced on the same data.

The analysis section of the report focuses on the analytical techniques used to achieve the end results. In particular the data mining process model, a statistical approach using TF-IDF and support vector machine (SVM) are justified with a discussion on possible alternatives. The evaluation techniques used to evaluate the SVM are also highlighted.

The results highlight the findings of each implementation of the techniques and methodologies performed in the methodology and analysis sections. A comparison with relevant literature and findings of this studies results are compared with the results of similar studies using the same techniques.

2.0 Data

This section discusses the data used, the source of the data, the purpose of the data as well as some attributed of the data such as memory cost, size etc.

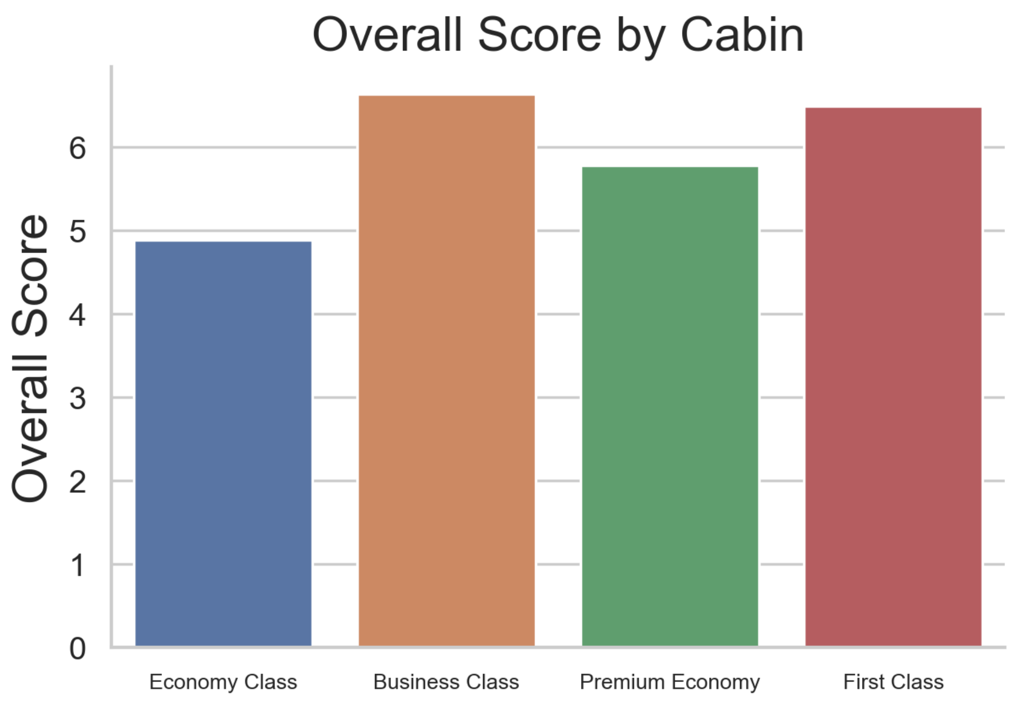
The data used in this project was extracted as an Excel file from Kaggle, a machine learning and data science community used to publish projects and share data sets collected. Kaggle user '*EFEHAN*' published the data set after web scraping the data from '*Skytrax*' as part of their '*Big Data Analytics*' programme at MEF University. The user states how the data was used to gain insights into the airline industry and invited others to implement machine learning models and particularly prediction models on the data (Danisman, 2019). The Skytrax website is an airline review and rating site which serves as an air travel guide. Users of the site can sign up and rate their journeys with particular airlines and rank certain amenities. The reviews are published to the site for other users to use as recommendation.

The data comes in the form of an Excel (.xlsx) file costing *24.24* megabytes of memory. The file contains *132,000* rows total and consists of customer reviews ranging from *2006* to *2019*. The data set was saved locally and then produced to a data frame in the Jupyter

Notebook environment using the '*Pandas*' library to create a '*pandas data frame*'. The fields and feature descriptions are produced in Table 1.

As part of the exploratory analysis the 'seaborn' package was used to create a bar graph showing the overall score provided by passengers allocated to different cabins onboard the airplane. The graph produced in Figure 1. Shows how passengers travelling '*business class*' tend to score higher overall. This is interesting as it would be expected 'first class' passengers would rate higher although this may be a result of greater expectation for a first-class ticket in comparison to business class tickets.

Figure 1. Overall Score Based on Cabin



2.1 Business Understanding

Table 1.

Column Name	Description	Type
airline	The name of the airline flown	String / text
overall	The overall score provided by the customer	Integer (1-10)
author	The author / username of reviewer	String / text
review_date	The date the review was published	Date
customer_review	The text review of the trip	String / text
aircraft	The type of aircraft flown	String / text
traveller_type	The seating class the customer flown in	String / text
cabin	The cabin class the customer flown in	String / text
route	The departure and arrival city	String / text

date_flown	The date of the flight	Date
seat_comfort	A score of seat comfort from 1-10	Integer (1-10)
cabin_service	A score for cabin service from 1-10	Integer (1-10)
food_bev	A score for food and beverages on board from 1-10	Integer (1-10)
entertainment	A score for on board entertainment from 1-10	Integer (1-10)
ground_service	A score of the ground service provided by the airline from 1-10	Integer (1-10)
value_for_money	A score between 1-10 ranking the value for money of the flight	Integer (1-10)
recommended	A recommendation from the customer for the flight flown	Text / binary value (Yes / No)

2.2 Data Understanding

In order to gain further understanding of the data various techniques were used to provide information regarding the size and shape of the data as well as exploration of some of the contents of the data.

The first step undertaken in this process was to use the ‘Pandas’ library to create a Pandas ‘data frame’ of the data file, outlined in Section 2.0, saved locally. The benefit of using the data frame, is that it provides the opportunity to take the semi structured data file and make it more accessible to libraries and functions from within the Python code file as opposed to attempting various operations on the locally saved file. Data frames can also be altered and changed to produce different variations of the data without the need for altering the original data file. By producing the initial rows of the data frame in table format, the observation is that the original data file contained a blank row between every row of data.

Checking the ‘length’ value for a particular column in the data frame reveals *131,895* rows. By removing fully blank rows, setting a threshold of *17* empty rows, this value is halved to a dataset containing *65,947* rows of possible, usable data. In order to avoid missing values, further removal of ‘NaN’ values was performed, requiring a threshold of *16* data values present for the row to be eligible, thus removing any rows of the data missing at least 1 value. Through this process, the dataset is further reduced to contain *28,497* rows of usable data. This information is confirmed by checking the number of rows once the removal of ‘NaN’ values is complete to satisfaction. A usable row of data in this context is being defined as a row of data which consists of a customer review text as well as the overall score provided by the customer.

3.0 Methodology

The primary data mining methodologies used are outlined in this section. The methods for pre-processing the data in order to transform the data into usable information for a machine learning model are explained. The processes involved include a manual review of the data set which helped to identify the first major issues with the data such as blank rows and missing values. Further more common techniques such as converting text to lowercase, removal of ‘stop words’ and removal of punctuation.

3.1 Data Preparation

Through the analysis performed in the data understanding stage, the first issue identified within the data was the occurrence of blank or fully ‘*NaN*’ rows of data, appearing in between full, usable rows of data. The initial step undertaken was to set a threshold of *17* full columns of data for the row to be eligible, this removed all of the completely blank rows and halved the data set from *131,895* to *65,947* rows. The next step of the process was to remove any rows which contained at least 1 missing data value, this was achieved by setting the threshold requirement value to *16*. Through this process, any row of data which was missing at least *1* value would be removed from the data. This step reduced the dataset further to consist of *28,497* rows. An alternative process to achieve similar results could be to reduce the data frame to consist of only the customer review text and overall score and implement the same operation. By taking this alternative option it may ensure that some rows which may be missing a value in the ‘airline’ column, although contain customer review text and an overall score value, are not removed. In the context of this research, the goal was to keep as many ‘full’ rows of data as possible for contention in various models available.

The removal of blank rows reduced the dataset from *131,895* rows to *28,497*. The resulting number of rows from this process was considered satisfactory as related works in this area mostly implement modelling on much smaller datasets for ease of use. A similar study performed by Garcia et al. performed lexicon-based classification techniques on customer review text of restaurants and hotels. The study used outlines the use of *1,000* reviews of restaurants and separately *994* reviews of hotels (Aitor Garcia, 2012). Another study performed by Gräbner et al. which similarly used ‘Trip Advisor’ reviews for the sentiment analysis of hotel reviews, used a total of *1,000* reviews from the website. This study balanced out what was outlined as a “*skewed dataset*” by only using a sample of *200* reviews from each star ranking, of *1-5* (Dietmar Grabner, 2012).

Once a full block of complete usable data has been established, further implementations of pre-processing techniques can be applied. The aims for this phase of the development are to turn the usable data which is human readable, into chunks of information which can be interpreted and understood by a machine learning model. The objective is to tidy the data into its most basic form as well as handle any components of the data which may cause problems if read by the machine learning model. Another key aspect of this stage is to remove any information which may not provide any kind of predictive value to models used.

In order to begin the process of transforming the data into its most basic form, the next step performed was to convert all of the text from the ‘*customer review*’ column from a mixture of uppercase and lowercase, to plainly lowercase characters. This is done in preparation for techniques used later in the process which involve actions such as counting a frequency of a word when vectorizing the texts words into numerical values (see Section 4.0). The purpose of this preparation is to ensure that words aren't miscounted or identified as separate words based on text casing, for example "*Great*" and "*great*" being counted as their own unique word when in actuality these words count should be added together. To avoid the longer process of counting each word and checking for the same spelling and then counting the total, the more efficient process is to transform all of the strings in the text to lowercase.

Another common method used and implemented in the pre-processing stage was the removal of punctuation marks. Punctuation is used throughout written text to allow the author to provide clarity in their writing and for the reader to understand which elements of the text requires a pause or stop. It is stated in literature that punctuation marks inside text can have an effect machine learning models – particularly on models which rely on the process of implementing a word to vector approach, as in this study. Although, it is proven the removal of punctuation does not improve the precision of the model and is simply performed to facilitate ease of use (Amit Purushottam Pimpalkar, 2020). This means the removal of punctuation differs from a technique such as converting text to lowercase characters as this technique is shown to improve accuracy by preventing the same words being considered different. The removal of punctuation serves the purpose of preventing misreads in the text although does not affect the accuracy of the model (Amit Purushottam Pimpalkar, 2020).

The removal of '*stop words*' was the next process undertaken. 'Stop words' refers to frequent words commonly used to bridge the gap between nouns, verbs and adjectives. Words such as '*a*', '*will*' and '*are*' fit into this category. These words in the customers written review will not provide much - if any, context. Therefore, it is considered that stop words will not provide any predictive value to the models. The removal of stop words was performed using the '*Natural Language Tool Kit*' (NLTK) libraries package '*stopwords*'. This package consists of an English language 'corpus' or dictionary, of the most common stop words - *179* in total, used in written English. The partly processed customer reviews are looped through and reconstructed based on the condition the word checked does not match a word within the 'stop word' dictionary thus, creating a new review consisting of the text without any of the pre-defined stop words.

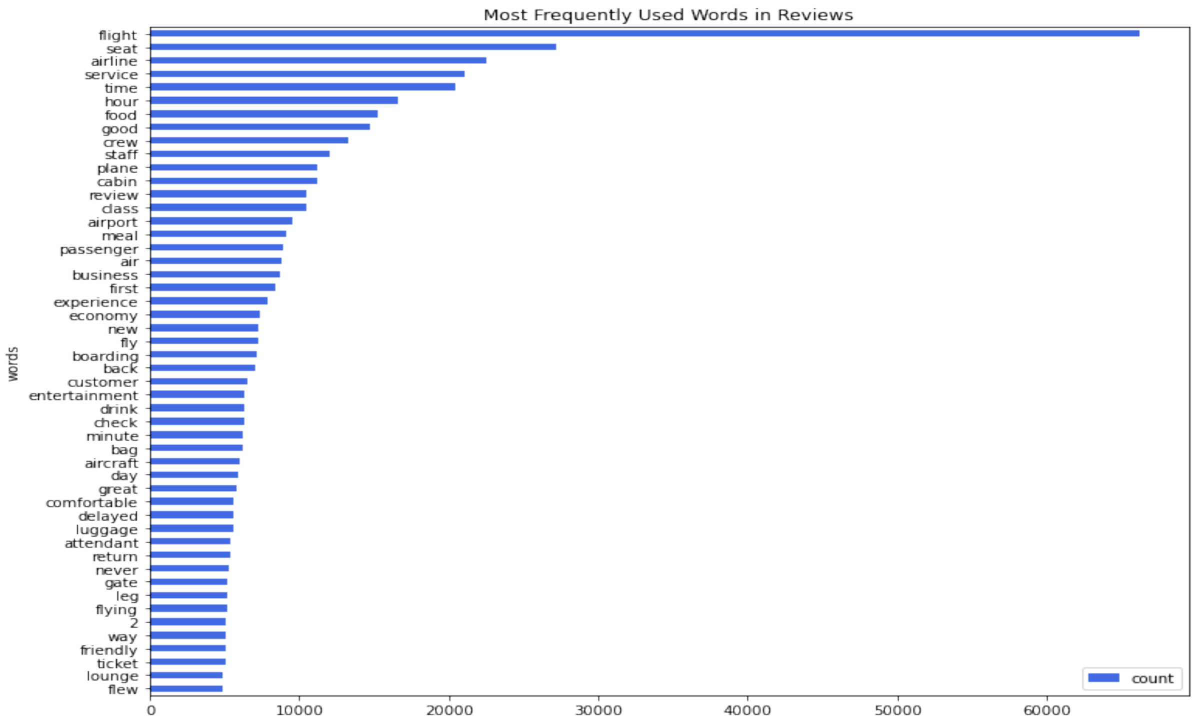
As mentioned in Section 4.0, the primary purpose of choosing a process model framework such as CRISP-DM allows for the decision to iterate back to the 'data understanding' (Section 2.2) stage and explore the data further from this point in the pre-processing stage. In order to explore the data further, a frequency count is then performed to provide an overview of the most common words still present in the reviews after the removal of the most common stop words. This is performed so that words which may be uniquely common to the subject area of airline customer reviews although, not frequent enough within everyday written English to be included in a dictionary of common English stop words, can be handled. This process is achieved by taking all of the customer reviews which just had stop words removed and putting the words into one large review and then counting them. The top *50* results were filtered and printed to provide an overview of the most common words in the most recently processed reviews, in order to manually check for any words which may be frequent although, do not provide any predictive value (see Table 2.).

Table 2. First Frequency Count – sample of top 15 results.

Word	Count
Flight	56,082
Service	19,984
Verified	19,436
Time	17,409
Food	15,188
Good	14,696
Seat	14,256
trip	13,587
Seats	12,928
Crew	12,703
Staff	11,935
Airline	11,475
One	11,452
Would	11,364
Airlines	11,036

Words such as "*trip*" and "*verified*" were amongst the most common words identified in this frequency count. Each customer review contained the phrase "*trip verified*" at the beginning if the website Trip Advisor had manually reviewed the review to ensure authenticity. A new list of extended stop words was then created containing the most frequent words from this frequency count which were deemed not to provide any predictive value. The same process was recycled of removing these extended stop words from the previous iteration of stop word removal, along with a frequency count of words in the most recently processed reviews (see Figure 2.).

Figure 2. Final Frequency Count



The final step in the pre-processing strategy was implementing '*lemmatization*'. This is the process of taking the words from the documents and reducing them to their base meaning i.e., '*changes*', '*changing*' and '*changed*' are reduced to the root '*lemma*' word which is '*change*'. The '*NLTK*' library '*WordNet*' function '*WordNetLemmatizer*' was used in conjunction to a loop which read each word in a document (review) and returned the lemmatized version of the word. Another '*NLTK*' function used in this process was the '*WhitespaceTokenizer*' which is used to identify spaces and/or shifts to the next line in text. This function supports the lemmatization process in identifying separate tokens of words.

4.0 Analysis

The analysis process began with a decision on an appropriate data mining process model which could be used as a framework / lifecycle to follow and iteratively progress through the various stages and techniques required to effectively analyse the data. Effectively, following a data mining process provides a clear 'road map' of stages required to achieve the aims outlined in Section 1.2.

The process models in consideration for this analysis include CRISP-DM – '*cross industry standard process for data mining*', KDD – '*knowledge discovery in databases*' and SEMMA – '*sample, explore, modify, model and assess*'. These process models are three of the most commonly used data mining frameworks. KDD was the original process model created in *1989* however, the SEMMA and CRISP-DM methodologies shortly followed in the year *1996* in order to create variety in the selection of a process model. The latter models (SEMMA and CRISP-DM) incorporated their own alterations and updates on the original process model (KDD) to better suit modern data mining techniques. The CRISP-DM framework appears more thorough with the inclusion of an additional step in comparison to SEMMA and KDD's five steps. In comparison, the CRISP-DM process model provides a more comprehensible iterative flow between the different phases involved in the lifecycle as well as a clearer definition of what is expected from each phase (Ana Azevedo, 2008).

4.1 Modelling

The first step in the modelling process was to identify relevant models typically used in this area of research. In literature, the most common classifier algorithms implemented in problems which require text classification are the 'Naïve Bayes' classifier, '*K-Nearest Neighbour*' (KNN) and '*Support Vector Machine*' (SVM). In similar studies performed the 'SVM' classification algorithm produced the best performance. In particular a study performed by Lifu Chen at al. (Lifu Chen, 2005) outlined the SVM models better performance in terms of accuracy of the classification of reviews written in Chinese in comparison to Naïve Bayes and KNN, with an accuracy of *91.15%* (W. Zheng, 2009). A study performed on a dataset by Brooke et al. which had a SVM perform to an accuracy of *85.1%* on *2,000* English written movie reviews on the classification of positive / negative features (Brooke, 2009).

A '*Support Vector Machine*' (SVM) works by separating data into categories on different hyperplanes. The goal of an SVM is to increase the margin between different category labels among the hyperplanes so that an appropriate fit for a hyperplane can be identified based off of the data points provided. SVM falls into the category of '*statistical learning*' which differs from '*lexicon based*' approaches commonly used in literature in the field of text classification (Barakat AlBadani, 2022). A lexicon-based approach may also be implemented with an SVM. A common approach in text classification literature and in particular sentiment analysis, is to take a corpus of negative and positive words and/or phrases with attached weighting and polarity scores and apply these scores when the words are found in the text document. There is less publication on the use of statistical based learning approaches in conjunction with SVM, so for this study the chosen method of categorisation was to use a '*term frequency – inverse document frequency*' (TF-IDF) algorithm.

TF-IDF is a way to quantify the importance of a specific word in a list of words. The weight of importance grows with the number of times the word appears in a review however, is offset by the word's total frequency in all reviews. This approach is used in a study of sentiment analysis on hotel reviews performed by Shi et al. (Han-Xiao Shi, 2011) (Amit Purushottam Pimpalkar, 2020). (See eq. (1). (Amit Purushottam Pimpalkar, 2020)

$$TF(t) = \frac{(Number\ of\ times\ term\ t\ appears\ in\ review)}{(Total\ number\ of\ terms\ in\ all\ reviews)} \quad (1)$$

This algorithm eq. (1) and supporting functions were implemented using the '*Sklearn feature extraction*' package with the '*TfidVectorizer*' library. In the data exploration performed (see Section 3.1), a word count of the most frequent words and their occurrences revealed that a majority of the top values begin to fall under *10,000* occurrences after the first *25* results. A minimum features value was chosen of *100* as *5-10* thousand occurrences is still quite a high amount for the bottom half of the top *50* results. The processes were also performed using an increased number of maximum features of *300* to test the performance when the value of features increased. From this process, a data frame consisting of each review as a series of TF-IDF scores replacing each word is created. The '*ReviewScore*' column was then added to this data frame with the intention of using this value as the target value. The purpose of using the '*ReviewScore*' variable in this instance is to analyse the performance of the SVM on predicting the overall score of the review based off of the TF-IDF scores. Using the '*sklearn.model_selection*' package with '*train_test_split*' library, the new data frame containing the words as vectors and the 'ReviewScore' is then separated into training and test splits (see Figure 3.).

Figure 3. New Data Frame – sample of final 9 features with 'Review Score' – 100 features data frame

91	92	93	94	95	96	97	98	99	ReviewScore
0.0	0.065245	0.116811	0.0	0.0	0.0	0.103997	0.0	0.0	7.0
0.0	0.0	0.0	0.0	0.377869	0.0	0.0	0.0	0.0	2.0
0.0	0.0	0.133802	0.0	0.247387	0.0	0.0	0.0	0.0	3.0
0.0	0.082371	0.0	0.0	0.0	0.146161	0.0	0.0	0.0	10.0
0.0	0.402861	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

A *70%* training and *30%* testing split is chosen using a 'random state' of *109*, in order to aid in reproducibility. This random state variable was chosen at random considering the results of a check on different values (see Table 3.). During the process of splitting the data, the target or predicted variable is defined as the 'ReviewScore' variable added to the data frame in the previous steps. The *70/30* split is chosen based off of the literature provided by Zi-qiang Wang et al. (Z. Wang, 2006) in which the optimal metrics are outlined for the most optimal implementation of a SVM on text classification problems.

Table 3. Random State Values with Results

Random State Value	Linear Kernel Accuracy	RBF Kernel Accuracy
0	37	39
109	38	40
1,000	38	40
12,000	38	40
20,000	37	38
29,000	38	40
30,000	39	40
100,000	38	40

(a) The SVM is implemented from the '*sklearn' packages library*'– '*svm*'. An initial run of the SVM is created by defining the classifier using the 'svc' function '.SVC' setting a parameter of a '*linear*' kernel. The model is then trained on the testing sets of data. A prediction response for the testing set is created using the '*.predict()*' function. This process was implemented using both *100* TF-IDF vector features and *300* TF-IDF vector features.

(b) The same process applied in Section 4.1(a) was then applied once again. This time, the classifier is defined with the parameter of using a '*Radial Basis Function*' (RBF) kernel as opposed to a linear kernel. The purpose of this selection was to provide a comparison to the linear kernel's linearity assumption between the TF-IDF values and the 'ReviewScore' target variable, by introducing a nonlinear kernel - RBF. The choice of RBF over others such as '*sigmoid*' is due to the RBF kernel being a "*reasonable first choice*" and common choice among many text classification problems (Vasileios Apostolidis-Afentoulis, 2015).

(c) In order to identify the most optimal range of hyperparameters, a loop was created to iterate through a cross validation of various hyperparameters such as '*C*' value, '*gamma*' value and type of kernel. This cross validation was achieved through the use of the '*sklearn.model_selection*' packages, library – '*GridSearchCV*'. This process was implemented testing both linear and RBF kernels. This technique works by defining a dictionary of '*k-fold*' validation values to use in order to test the model's best performance without underfitting or overfitting. The dictionary is defined in a '*parameter grid*' which states the '*C*' values to be tested, '*gamma*' values to be tested and type of kernel to use. This cross-validation testing was performed on both linear and RBF kernels. The '*C*' value is the regularisation value which evaluates the accuracy of the prediction taking into account the decision boundary. The decision boundary is a margin of estimation allowed for a value to be considered '*accurate*' when training i.e., a low C value is considering a larger margin between the actual result and the predicted result to be accurate making decision making easier for the model. If a gamma value is too high then it is assumed there is a risk of overfitting the data. A low gamma value assumes that the model is not taking into account variance in the data (Mei-Ling Huang, 2021). 5-fold cross-validation on 25 candidates was performed totalling 125 fits for both linear and RBF kernels (*see Table 4.*).

Table 4. Parameter Grid Dictionary – Linear and RBF Kernels

C Value	0.1	1	10	100	1000
gamma value	1	0.1	0.01	0.001	0.0001

(d) After the results of the method implemented Section 4.1 (c) outlined in Section 5.0, a process to improve the model was implemented. Predicting values of the range *1-10* yielded best results of *40%* when using a maximum of *100* TF-IDF features and with an assumed drop off in accuracy due to the imbalances in the data. A process to improve this classification accuracy score was implemented by creating a feature for grouping the review scores into specific labels. A condition was implemented using Python to assign the label '*1*' to review scores from *1-4* representing negative reviews, label '*2*' to review scores *5-6* representing neutral reviews and label '*3*' to review scores of *7-10* representing positive reviews. A new data frame was produced containing the TF-IDF vector features along with the review score and '*sentiment score*' – referring to the newly created labels. This new data frame was then split into a *70%* training, *30%* testing split of data with the target variable now being defined as the '*sentiment score*' variable. The '*review score*' column was removed from the data in order to provide the SVM with only the TF-IDF values from the customer reviews to predict the target variable of 'sentiment score'. This method provided a more realistic or expected depiction of the SVM performance when producing overall scores from simply using the text-based customer review.

4.2 Evaluation

To evaluate the SVM models implemented in Section 4.1 (a) and Section 4.1 (b), the library '*metrics*', from the '*sklearn*' package and '*classification_report*', '*confusion_matrix*' libraries from the '*sklearn.metrics*' package. A classification report is created using the '*y_test*' values from the testing split of data and the '*y_pred*' predicted values from the training data split. This classification report produces a table displaying the precision, recall, f1-score and support values as well as accuracy, macro average and weighted average. The results of these classification report can be viewed in Section 5.0 Tables 14. and 15. For runs consisting of *100* TF-IDF features and Tables Using the '*metrics.accuracy_score*' function an average accuracy value is produced along with functions '*metrics.precision_score*' and '*metrics.recall_score*' to produce the average precision and recall scores which can be seen in Section 5.0 Tables 10. and 11.

Evaluating the processes of cross-validation using '*GridSearchCV*' outlined in Section 4.1 (c) is a task which requires more time. The time to run this process of cross-validation was > *24 hours* on the system information outlined in Table 5. To save time on this process the results of each fit from the hyper parameter tuning were saved locally to Excel and a data frame was produced using the Excel file. The limitation caused by saving the results to Excel is that the '*GridSearchCV*' functions such as classification report cannot be produced although, the same results can be gathered through basic operations on a Pandas data frame of the results. In order to evaluate the best parameters from the grid search, plots can be created in order to identify the best values with the lowest run time (see Figures 4. and 5. for linear runs, Figures 6. and 7. for RBF runs.).

Table 5. System Information

Memory	8.00 GB
Speed	2.50 GHz
Processor	AMD Ryzen 3 2200U
Environment	Python 3.6 (ipykernel) – Jupyter Notebook

A similar process to the previous iterations of the SVM model evaluations was used to evaluate the performance of the implementations outlined in Section 4.1 (d) – using the newly created '*sentiment score*' as the target variable. The functions from the '*metrics*' library such as '*.accuracy_score()*' and '*classification_report()*' were used to provide an insight to the models performance. The resulting classification report for 4.1 (d) is reproduced in Table 10. The method outlined was also evaluated using 'GridSearchCV' to perform 5-fold cross validation for 9 candidates, totalling 45 fits. The number of candidates or '*C*' and '*gamma*' values was reduced to produce faster results than previous cross validation attempts which resulted in *> 24 hour* runs.

Table 6. Parameter Grid Dictionary for 4.1 (d)

C Value	0.1	10	1000
gamma value	1	0.1	0.01
Kernel	Linear		

5.0 Results

The results in Table 8. display a classification report from the linear kernel SVM run using '*GridSearchCV*' with a maximum value of 100 TF-IDF vector features, which reached an accuracy of *38%*. The precision values for predicting review scores of *4.0* and *6.0* are *0.00*. The precision value is calculated based off of the formula in eq. (2) and refers to the number of predictions correct over the number of actual values for that target variable (Z. Wang, 2006). Recall is similar to precision although, precision takes into account a total of all positives, recall is the number of correct predictions over the number of actual positives and negatives (see eq. 3) (Z. Wang, 2006). The classification report outlines a weighted average precision of *27%* and a weighted average recall of *38%*. Weighted averages were used in order to produce a prediction based on proportion as the predicted values are imbalanced. This can be identified in the '*support*' column of the classification report showing the number of instances of each review score value. With values such as *8*, *9*, and *10* each having over *1,000* occurrences and values such as *4*, *5* and *6* only totalling *1,145* occurrences together. It may be reasonable to assume the lack of precision on predicting these values is a result of the low number of these values in the data.

When the number of TF-IDF vector features is increased from 100 to 300, the results of the accuracy improve slightly to 41%. The precision percentages also increase across all predicted values especially in the previously mentioned values of '*4.0*' and '*6.0*' increasing to an average of *20%* between them from *0%* when using *100* features. The precision becomes much more evenly distributed when more features are included for the SVM model to base predictions off. An adverse effect to adding more features is running the risk of overfitting the data, that is not the case in this study as it is shown to increase the results of the model. The classification report for the model using 300 features is outlined in Table 9.

The results in Table 10. Outline the classification report produced from the single run RBF kernel SVM. Using the RBF kernel produced a slight increase in performance with an accuracy reaching *40%* - a *2%* improvement from the linear kernel model. The weighted average precision also increases to *34%* and then weighted average recall increases to *40%*. The precision accuracy is overall much more evenly distributed amongst the RBF predicted values. As a linear kernel is parametric relying on the assumption of a linear relationship between the predictor variables and target variable and the RBF kernel is non-parametric meaning it does not rely on any assumptions, this may lead to the more evenly distributed precision amongst the target variable and the overall better accuracy. However, the average score for the linear kernel produced a result of *37%* to the RBF kernels average score of *34%*. The linear kernel may have performed better more consistently although, the RBF kernel produced the highest overall score of *40%* to the linear kernels *38%*.

Displayed in Table 11. is a reproduced classification report returned from the analysis of the SVM RBF kernel single run using 300 TF-IDF vectorised features. As expected, based on the results of the linear kernel single run outlined in Table 9. using the same number of features - the results improved. This time the RBF kernel reached an accuracy of *44%* and a weighted precision of *41%*. An interesting value appearing from the inclusion of more features is the precision score of *57%* on the predictor value of '*4.0*' which has had a precision of *0%* in a *100* feature, single run of a linear kernel SVM model. This is also interesting due to the low 'recall' score of *06%*.

The '*GridSearchCV*' best parameter search results were plotted for both kernels - linear and RBF. The plot in Figures 4, 5, 6 and 7 measure the accuracy score both the C value and gamma parameters used in cross validation. These results were achieved using system specifications outlined in Table 5. Table 7 highlights the best parameters found for the best accuracy. The plots can be cross referenced to see both best the performing C and gamma values from a single plot although, the creation of a second plot was used in each instance to highlight where the specific parameters fall in relation to one another.

Table 7. Best Parameter Cross-Validation – Linear & RBF

Kernel	C Value	Gamma Value
Linear	1.0	0.0001
RBF	1.0	1.0

$$Precision\ Score = \frac{Total\ True\ Positives}{Total\ Actual\ Positives} \quad (2)$$

$$Recall\ Score = \frac{Total\ True\ Positives}{Total\ Actual\ Positives + total\ false\ negatives} \quad (3)$$

Table 8. Linear Kernel Single Run Classification Report – 100 TF-IDF Features

	precision	recall	F1-score	Support
1.0	0.47	0.94	0.63	2027
2.0	0.11	0.01	0.01	789
3.0	0.09	0.01	0.01	607
4.0	0.00	0.00	0.00	380
5.0	0.14	0.06	0.08	377
6.0	0.00	0.00	0.00	388
7.0	0.16	0.01	0.03	635
8.0	0.26	0.37	0.30	1018
9.0	0.31	0.32	0.32	1205
10.0	0.36	0.47	0.41	1123
Accuracy			0.38	8549
Macro avg	0.19	0.22	0.18	8549
Weighted avg	0.27	0.38	0.29	8549

Table 9. Linear Kernel Single Run Classification Report – 300 TF-IDF Features

	precision	recall	F1-score	Support
1.0	0.52	0.93	0.67	2027
2.0	0.23	0.02	0.04	789
3.0	0.15	0.05	0.08	607
4.0	0.22	0.03	0.06	380
5.0	0.17	0.09	0.12	377
6.0	0.18	0.04	0.07	388
7.0	0.24	0.14	0.17	635
8.0	0.30	0.38	0.33	1018
9.0	0.36	0.35	0.35	1205
10.0	0.44	0.53	0.48	1123
Accuracy			0.41	8549
Macro avg	0.28	0.26	0.24	8549
Weighted avg	0.34	0.41	0.34	8549

Table 10. RBF Kernel Single Run Classification Report – 100 TF-IDF Features

	precision	recall	F1-score	Support
1.0	0.50	0.92	0.65	2027
2.0	0.22	0.07	0.10	789
3.0	0.23	0.05	0.08	607
4.0	0.33	0.02	0.03	380
5.0	0.25	0.08	0.13	377
6.0	0.21	0.02	0.03	388
7.0	0.27	0.10	0.15	635
8.0	0.26	0.37	0.31	1018
9.0	0.34	0.36	0.35	1205
10.0	0.40	0.48	0.43	1123
Accuracy			0.40	8549
Macro avg	0.30	0.25	0.23	8549
Weighted avg	0.34	0.40	0.33	8549

Table 11. RBF Kernel Single Run Classification Report – 300 TF-IDF Features

	precision	recall	F1-score	Support
1.0	0.53	0.93	0.68	2027
2.0	0.39	0.09	0.15	789
3.0	0.29	0.10	0.15	607
4.0	0.57	0.06	0.11	380
5.0	0.29	0.13	0.18	377
6.0	0.35	0.06	0.10	388
7.0	0.31	0.17	0.22	635
8.0	0.32	0.43	0.37	1018
9.0	0.40	0.40	0.40	1205
10.0	0.45	0.56	0.50	1123
Accuracy			0.44	8549
Macro avg	0.39	0.29	0.29	8549
Weighted avg	0.41	0.44	0.38	8549

Figure 4. Linear Kernel – Best ‘C’ Value

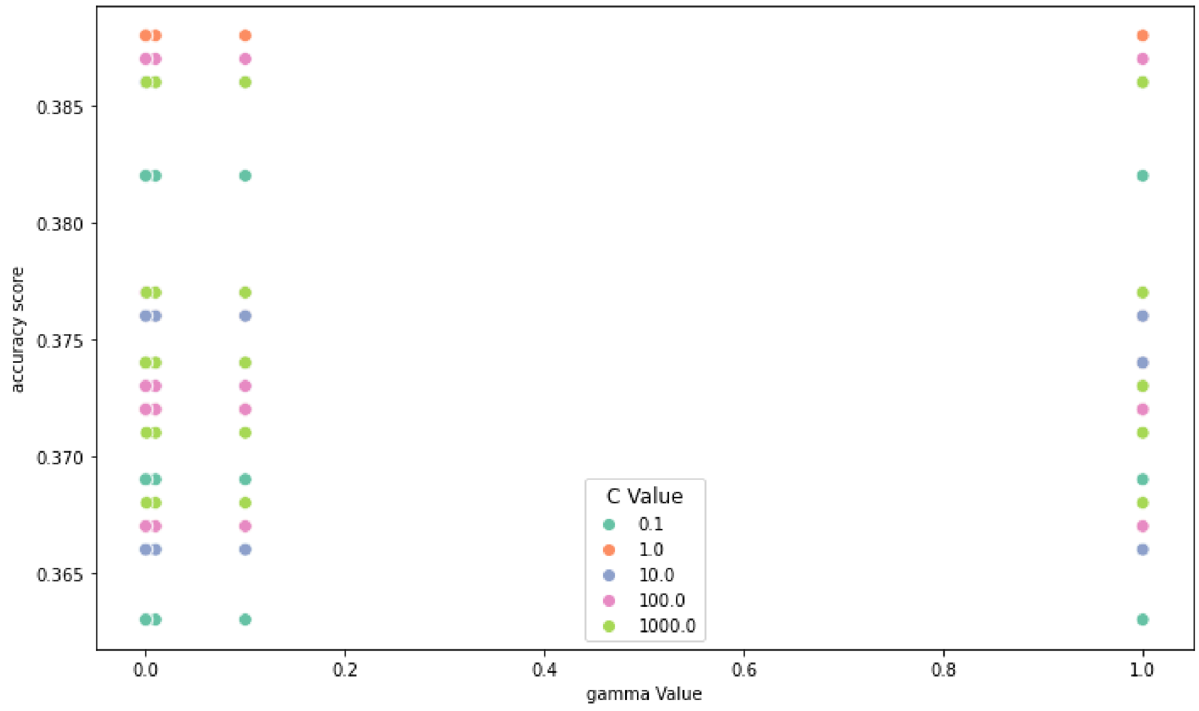


Figure 5. Linear Kernel – Best ‘gamma’ Value

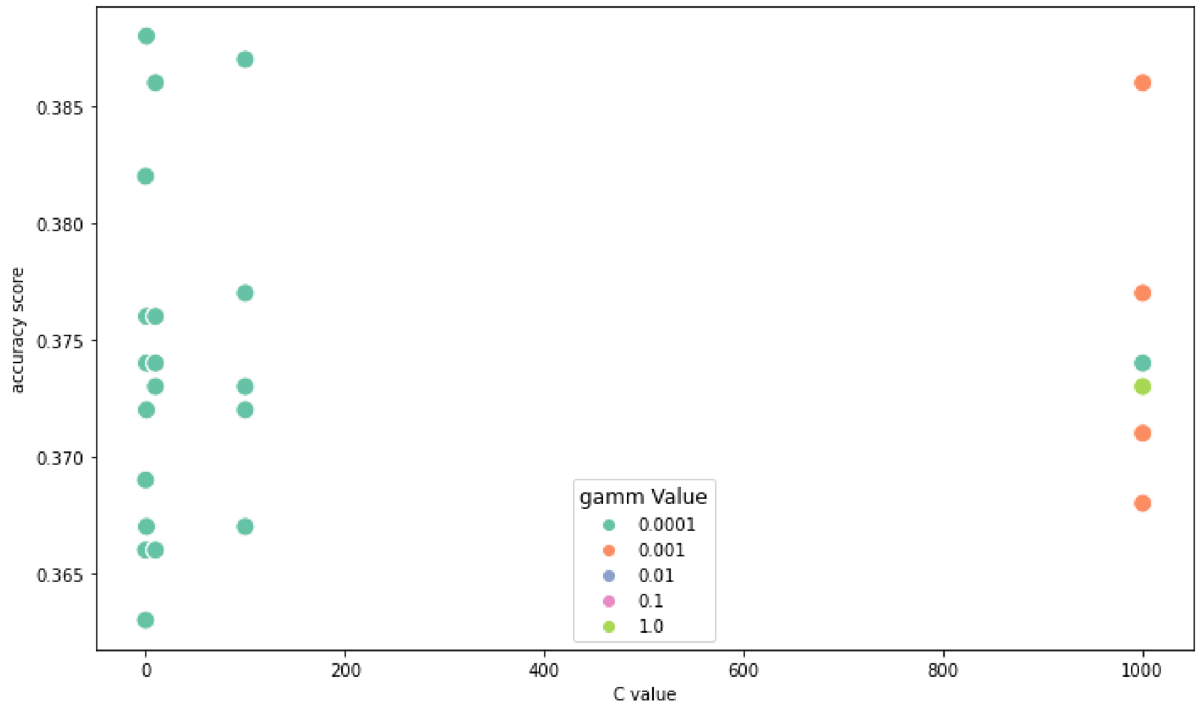


Figure 6. RBF Kernel – Best ‘C’ Value

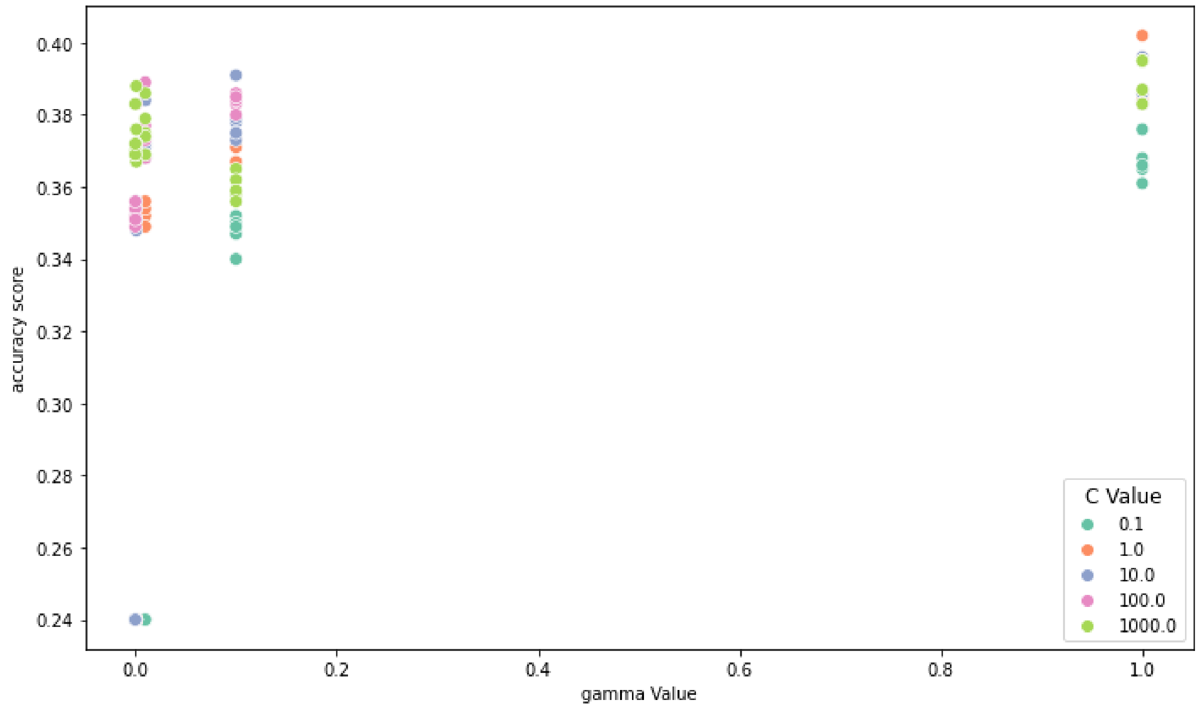
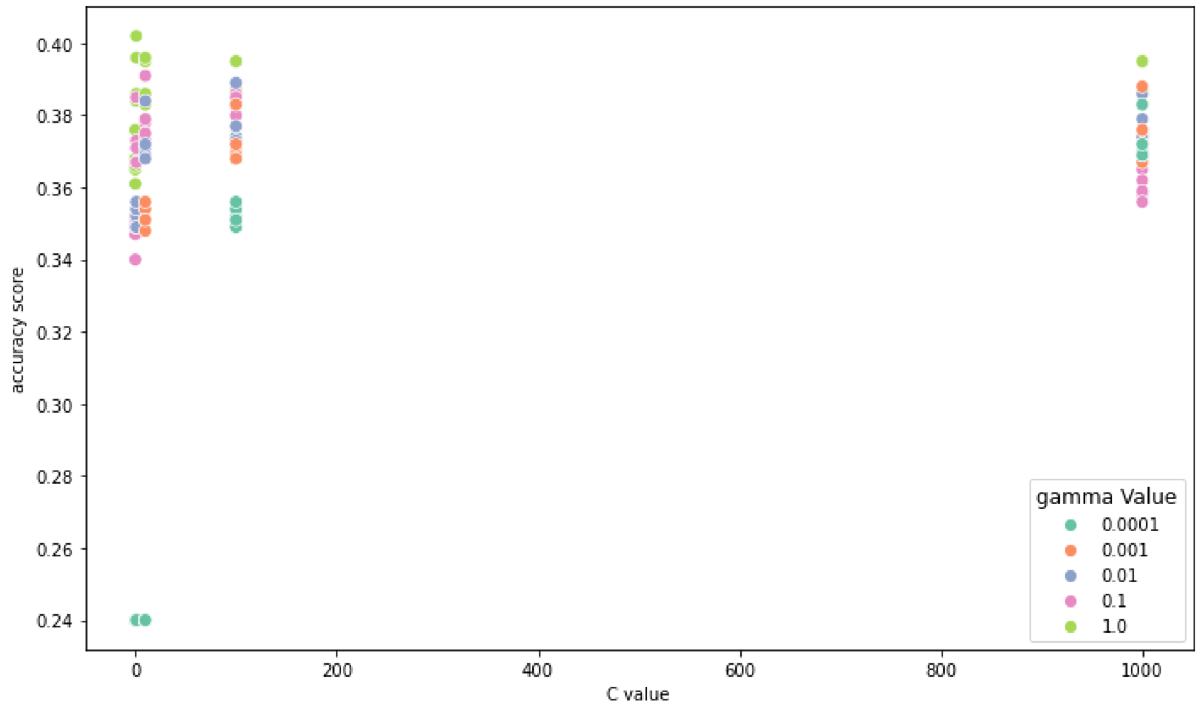


Figure 7. RBF Kernel – Best ‘gamma’ Value



Implementing the SVM with the labels for ‘*sentiment score*’ outlined in Section 4.1 (d) yielded more realistic results in an *80%* accuracy using a ‘linear’ kernel with *300* TF-IDF vector features and *83%* accuracy using an RBF kernel with 300 TF-IDF vector features. A major impact on the performance of this model using a linear kernel is the model’s performance of precision and recall on predicting the target label ‘*2*’ used to represent the neutral reviews scored *5* and *6*.

The assumption made from this is that the data is still unbalanced in that there are less labels of '*2*' when compared to the labels '*1*' and '*3*'. Making it more difficult for the model to classify these labels. A distribution of the number of occurrences of each label is highlighted in Figure 6. The results of the hyper parameter tuning performed using '*GridSearchCV*' resulted in best parameters outlined in Table 13 for *100* TF-IDF features. Table 15 displays the results of a classification report from the SVM model using *300* TF-IDF vectorised features with an RBF kernel. The non-parametric kernel proves to handle the imbalances in the data well by producing a precision score of *77%* compared to the linear kernel SVM runs result of *0%* on label '*2*' for neutral reviews which consists of less than *1,000* occurrences compared to label '*1*' and '*3*' which each contain over *3,000* occurrences. The model performs exceptionally well almost matching the results of published literature studies by reaching an accuracy of *83%*. Some of the literature for comparison is a study performed on tourism reviews written in English by Ye et al. (Qiang Ye, 2009) which produced an accuracy of *85.14%* on a binary classification task of positive and negative reviews. A similar study performed by Zheng et al. (W. Zheng, 2009) reproduced the same methods on travel reviews written in Chinese which reached results of *91.15%*. Both studies in this instance used support vector machines. The results of these studies in comparison to this study can be viewed in Table 16.

Table 13. SVM Hyper Parameter Tuning – Best Parameters – 100 features

C	**gamma**	**Kernel**
10	1	linear

Table 14. SVM Classification Report – 'Sentiment Score' target variable; attempt 02 – 100 features

	precision	**recall**	**F1-score**	**Support**
1.0	0.80	0.88	0.84	3803
2.0	0.00	0.00	0.00	765
3.0	0.79	0.87	0.83	3981
Accuracy			0.80	8549
Macro avg	0.53	0.58	0.56	8549
Weighted avg	0.73	0.80	0.76	8549

Table 15. SVM Classification Report – 'Sentiment Score' target variable; attempt 02 – 300 features

	precision	**recall**	**F1-score**	**Support**
1.0	0.84	0.92	0.88	3800
2.0	0.77	0.06	0.10	778
3.0	0.83	0.91	0.87	3972
Accuracy			0.83	8550
Macro avg	0.81	0.63	0.62	8550
Weighted avg	0.83	0.83	0.80	8550

Figure 8. Distribution of 'Sentiment Score' Labels

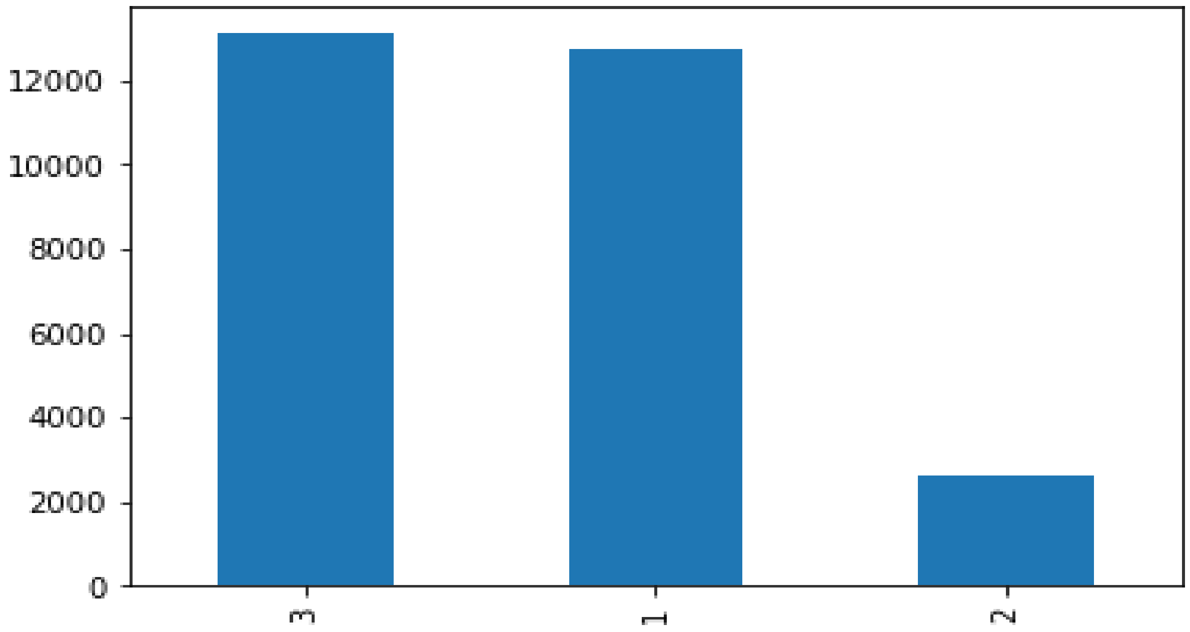


Table 16. Comparison with Literature

	Airline Customer Reviews (this study)	Chinese Travel Reviews (W. Zheng, 2009)	English Travel Reviews (Qiang Ye, 2009)	Movie Reviews (Brooke, 2009)
Accuracy	83%	91.15%	85.14%	85.1%
Data Size	8,550	775	1,191	2,000
Classification Type	Multiclass	Binary	Binary	Binary
Data Balance	Unbalanced	Balanced	Balanced	Balanced

6.0 Conclusions

This chapter will conclude the research by summarising the key results and discussing the advantages and disadvantages. It will also highlight the main limitations of the analysis performed. The study aimed to investigate a machine learnings models' ability to correctly classify customer sentiment and overall score based on the customers text reviews. A key result from this study is the performance of *83%* accuracy for multiclass classification when taking into consideration the data imbalances and size of the data. In comparison to similar studies using similar approaches the performance of the model implemented in this study compares well with the performance of models implemented on binary classification problems. This research uses a larger quantity of data in comparison to similar studies. By numbers, there appears to be an increase in model performance as the size of the data decreases. An advantage to this study is the key results highlighting the performance of a SVM model used in multiclass classification in comparison with binary classification solutions using the same methods. Another advantage to this study is the research involved in identifying and highlighting the best performing parameters for cross validation to contribute to the improvement of a SVM model's performance. A disadvantage to this study is its inability to evenly split the classification across *negative*, *neutral* and *positive* labels. There was a drop off in performance for the precision attribute for '*neutral*' labels by *6%* and *7%*. It could be argued by balancing out the data across these labels so each label is evenly represented, it may improve the precision on this label and overall improve the accuracy of the model. If the model were to improve to the same value or beyond in comparison to similar studies on binary classification problems - it may well present excellent findings. In some instances, the extended run times involved with hyper-parameter tuning meant that the Jupyter Notebook kernel would reset requiring the results to be saved locally. This meant it became more difficult to plot different functions available to the parameter grid dictionary such as graphs of the '*receiver operating characteristic curve*' (ROC) curve and 'area under the ROC curve' (AUC). These plots help to visualize the classification models performance with parameters for true positive rate and false positive rate. Another limitation to this study is that the data is extracted from one data source instead of multiple sources which may help to reduce any biases in the data set. The power of the SVM is highlighted in its performance for all instances implemented. It can be argued that the SVM would be expected to massively outperform human estimates of these scores. In particular the result of *44%* when predicting between *10* values. It could be argued a human may be expected to have *10%* accuracy by correctly estimating *10/100* attempts. The SVM massively outperforms this estimate by coming close to *50%*.

7.0 Further Development or Research

This section outlines the direction this project could take with additional time and resources. With additional time on this project an approach to attempt to match or outperform the results of the relevant literature would be explored. By balancing out the data to contain even numbers of negative, neutral and positive labels – a full comparison could be made with binary classification studies referenced which use the same techniques. The additional resources of a more powerful system may provide the opportunity to implement larger cross validation fits providing a more complete insight towards the best performing parameters as well as possibly identifying better performing parameters. This study implemented on a more powerful system may also aid in quicker cross validation allowing for the results of these process to be targeted with their built-in functions such as producing rates of true positives against true negatives. From much of the literature reviewed throughout this study, more powerful models have been identified with the use of deep learning algorithms such as '*Bidirectional Encoder Representations from Transformers*' (BERT). This methodology is a statistical approach similar to the TF-IDF algorithm implemented in this study. BERT differs from TF-IDF in that it weights words, represented in numerical form, based on their connections together. This algorithm makes it easier for models to recognise positive and negative phrases as opposed to only words and has shown to increase the performance of models such as SVM.

8.0 References

Aitor Garcia, S. G. a. M. T. L., 2012. 'A Lexicon Based Sentiment Analysis Retrieval System for Tourism Domain'. *e-Review of Tourism Research (eRTR),* 10(2 (ENTER 2012 Idea Echange)), pp. 35-38.

Amit Purushottam Pimpalkar, R. J. R. R., 2020. 'Influence of Pre-processing Strategies on the Performance of ML Classifiers Exploiting TF-IDF and BOW Features'. *ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal,* 9(2), pp. 50-58.

Ana Azevedo, M. F. S., 2008. *KDD, SEMMA and CRISP-DM: a parallel overview,* Instituto Politécnico do Porto. Instituto Superior de Contabilidade e Administração do Porto: http://www.iadisportal.org/digital-library/kdd-semma-and-crisp-dm-a-parallel-overview.

Barakat AlBadani, R. S. J. D., 2022. A Novel Machine Learning Approach for Sentiment Analysis on Twitter Incorporating the Universal Language Model Fine-Tuning and SVM. *Applied System Innovation,* 5(1)(13).

Brooke, J., 2009. *A Semantic Approach to Automatic Text Sentiment Analysis. M.A. thesis,* Burnaby, B.C., Canada: Simon Fraser University.

Danisman, E., 2019. *Skytrax Airline Reviews.* [Online]
Available at: https://www.kaggle.com/datasets/efehandanisman/skytrax-airline-reviews
[Accessed 14 May 2022].

Dietmar Grabner, M. Z. G. F. a. M. F., 2012. *'Classification of Customer Reviews based on Sentiment Analysis'.* Springer, Helsingborg, Sweden, 19th Conference on Information and Communicaion Technologies in Tourism (ENTER).

Han-Xiao Shi, X.-J. L., 2011. "A sentiment analysis model for hotel reviews based on supervised learning,". *International Conference on Machine Learning and Cybernetics,* Issue 10.1109/ICMLC.2011.6016866, pp. 950-954.

Lifu Chen, N. Z. D. L., 2005. "Study on machine learning based automatic text categorization model". *New Technology of Library and Information Service,* 21(10), pp. 23-27.

Mei-Ling Huang, Y.-Z. L., 2021. Use of Machine Learning and Deep Learning to Predict the Outcomes of Major League Baseball Matches. *Applied Sciences,* 10(11), p. 4499.

Qiang Ye, Z. Z. R. L., 2009. 'Sentiment classification of online reviews to travel destinations by supervised machine learning approaches'. *Expert Systems with Applications,* 36(3), pp. 6527-6535.

Vasileios Apostolidis-Afentoulis, K.-I. L., 2015. *SVM Classification with Linear and RBF kernels,* Thessaloniki, Greece: Academia.

W. Zheng, Q. Y., 2009. "Sentiment Classification of Chinese Traveler Reviews by Support Vector Machine Algorithm". *2009 Third International Symposium on Intelligent Information Technology Application,* Issue 10.1109/IITA.2009.457, pp. 335-338.

Z. Wang, X. S. D. Z. a. X. L., 2006. "An Optimal SVM-Based Text Classification Algorithm,". *2006 International Conference on Machine Learning and Cybernetics,* Issue doi: 0.1109/ICMLC.2006.258708, pp. 1378-1381.

9.0 Appendices

9.1 Project Proposal

Objectives

The ambitions for this project are to develop and create an accurate program which takes prepared datasets containing customer reviews from their journeys on airlines and specific flight routes and analyses the information in order to give an accurate predicted classification of a customer's review based on sentiment analyses of the text within the customer review.

By pre-processing and preparing the dataset which contains information such as the customers overall score (within a range of 1-10), a text review uploaded online and various other scores for amenities such as service provided, food and comfort; the aim is to create an accurate predictive model which can return an overall score and/or classification based on the contents of the customers text review.

The aim by the end of the project is to test multiple models used for sentiment analyses and compare which model gives the best overall accuracy in predicting the overall score based on the text review.

Background

This project was considered due to a big interest in the aviation industry. It is an industry of interest in and undertaking projects based on the industry will help with gaining experience and understanding of the different factors to take into consideration when working with data based on the aviation industry. This learning and any gained knowledge will benefit myself as a student learning more advanced computing and analytical concepts as well as my future self as a potential applicant to analytical roles within the aviation industry. Through this project there is an opportunity to gain experience working with the data.

To meet the requirements set out in Section 10.0 the aim is to research the different approaches towards conducting sentiment analysis. By learning the different approaches such as data gathering, data pre-processing and implementing models for sentiment analysis on the data, the goal is to begin to narrow down the tools and techniques required for such a project and begin to split the project work into smaller steps which need to be completed before any model testing can begin. One of the main goals early on in the project is to have the data prepared and ready so the focus can be on using the data on different models and experimenting with different techniques.

State of the Art

Performing sentiment analysis on text is no new feat. It is common to perform sentiment analysis on text such as social media posts and online reviews for products. Mainly, the goal of this sort of work is to have a model working and determining if the sentiment of the text provided is either negative or positive.

Where the project stands out is due to the prediction of an overall score based on the customer reviews text sentiment. Rather than just discovering if the text is negative or positive, the aim is to use the text in correlation with an overall score in order to predict the overall score of other reviews based on the text. The dataset also provides scores for '*comfort*', '*service*' etc. which are areas that could ideally incorporate into the analysis to find out if there is any correlation between particular areas with a high score and an overall high score. To analyse the reviews to determine if a '*negative*' sentiment was found although not a negative overall score, which areas contributed to the overall score being higher. Information such as this being discovered may be of interest to companies within the aviation industry such as airlines.

Data

For this project the data required is medium to large dataset which contains information including airline customer reviews. The dataset identified contains detailed information on airline customer reviews including information such as customer names, flight route, the review text entered by the customer and an overall score for the flight. Also included are scores for the different areas of interest for the flight such as comfort, service, food. No complementary datasets are required for this work as the analysis will be carried out mainly on the information provided by this dataset.

However, an option of using complementary datasets such as a '*flight route*' dataset could be optional if there is a wish to include further analysis such as identifying any reviews which mention '*turbulence*' and linking them to specific flight routes. This is not something the project aims to achieve but rather, an optional part of the project which may be chosen for inclusion if the base work goes to plan.

For accessing and compiling the data, the plan is to download the dataset from Kaggle in the form of an Excel (xlsx) file. Then, performing the data pre-processing on this dataset using '*Jupyter Notebook*'.

The dataset does contain arbitrary data such as the customer's name which does not provide any significance for the research so columns such as this may be extracted using Python. For the project, it is planned to use Python to read and pre-process the dataset. It will also make use of Python libraries such as '*Pandas*', '*NumPy*', '*NLTK*' (Natural Language Tool Kit)

List of potential datasets:

Airline Review Dataset*:*
https://www.kaggle.com/divyansh22/airline-reviews-eda-and-preprocessing-pt-1

Potential Complementary Dataset:
https://www.kaggle.com/open-flights/flight-route-database

Methodology & Analysis

This project will follow an agile Kanban approach to the development of the technology. The reasoning for using Kanban is because this approach is better suited to a single person project as opposed to other methodologies.

Looking at an example of agile methodology, the main focus of this method is to delegate components of the project into sprints of 2-4 weeks of work which need to be completed. The concern with an approach like this is that if one sprint falls behind, it may have a knock-on effect delaying the work further and thus leaving the project with a large backlog of work to be completed near the end stages or various deadlines set out. An example of such a deadline may be that of the midway presentation, if the project was to be two or three sprints behind by this point, it would mean a lot of work missing from what was originally planned by this point.

With the Kanban approach, the plan is to break the project up into individual tasks such as gathering weather data. A separate task would then be to clean this data to only the variables required. The final task might be to add this data to the system. Through following this approach there will be a more defined description of exactly what has been done so far and what still needs to be completed.

The main idea is to create many separate user stories for how the technology could be used, these user stories are then broken down into components which are required to be completed in order to have that specific functionality operating as wished. Once complete these components are moved into a "*testing*" stage where they will be tested for bugs. Once some functionality is working in this manner, it will be a matter of repeating the process for implanting the next user story with some modifications.

In terms of approach to research, the objective is to do some qualitative research in order to gather the evidence and reasoning behind particular contributing factors to high review scores. Once this approach is complete, hopefully it will provide an idea of what sort of numbers and variables required to identify in order to demonstrate '*positive*' reviews in facts. From here, moving onto quantitative research and seeing if the data gathered matches up to any of these figures found throughout the qualitative research.

Taking this approach to research the plan is to initially find the evidence and cause of a positive review in figures and to compare it to data currently available or gathered from predictive tools. From this, to then using Python and libraries such as Pandas, NLTK etc to identify reviews from the dataset with these characteristics.

Technical Details

The first technical development to be carried out is the gathering of the data. For this, some review datasets from online data sites such as Kaggle were downloaded. The datasets were then manually reviewed to provide an idea of the data contained within the datasets and the size.

Once an appropriate data set had been identified, work began on learning some of the pre-processing techniques required for conducting sentiment analysis on reviews. One of the first items which stood out was to remove any pre-existing sorting within the dataset. From reviewing the data, the dataset was sorted in list of airline name, an objective to remove this sorting was noted as the goal was to separate and split the data into a training/validation/test split.

A training/validation/test split is one of the techniques used for working with models on datasets, the purpose of this is so that there is data for training the model on, data for testing and data for validating the models results. The split chosen was that *80%* of the current dataset would become training data. *10%* of the data would become validation data and the final *10%* of the data would become testing data. Other splits include a *60% 20% 20%* as well as a *70% / 30%* split.

A concern was that *10%* of the test or validation split may become all reviews on one particular airline due to the pre-existing sorting. To navigate this, there would be a need to shuffle the rows of the dataset in order to remove this sorting. This would be done using the ‘*Pandas*’ library with Python.

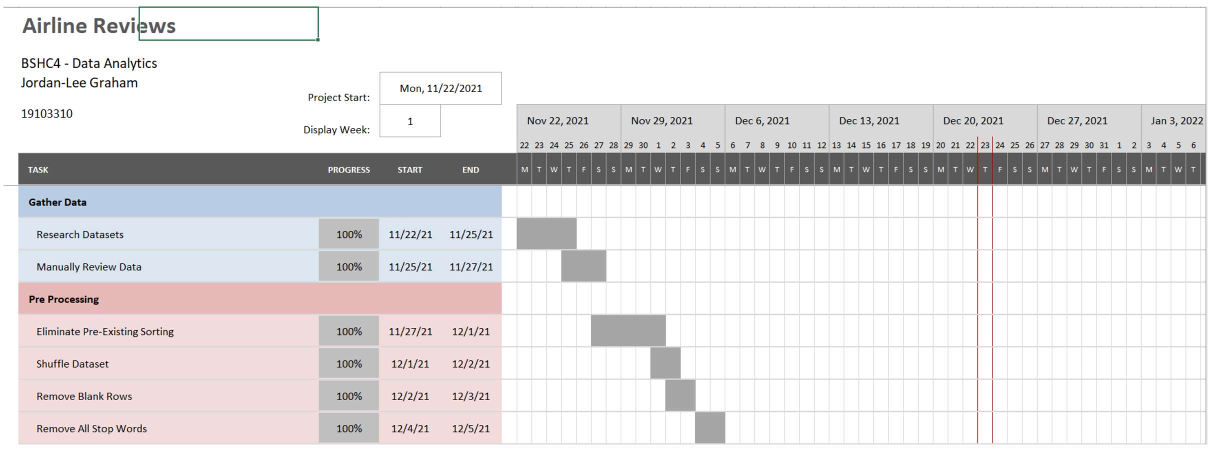
There would then be a need to separate the shuffled Excel file into the training/validation/test split and save the data to separate files in a safe place.

Once this was complete, the next step was to remove ‘blank’ rows of data from the split files which showed every second row as an empty row. This was easily sorted using Excels built in functions although it may be an option to write a Python script to automate the process in future.

The next step in the plan was to use the Natural Language Tool Kit (NLTK) library and use its built-in functions in order to extract ‘*stop words*’ from the dataset. This includes words which are not relevant to the customers review such as ‘*the*’ ‘*a*’ ‘*and*’ etc.

After the pre-processing is complete, the goal is to have working datasets ready for use. With the pre-processed datasets ready, the next step is to then attempt to run the data from the training split file of airline reviews through some sentiment analysis models such as Naïve Bayes or a ‘*Support Vector Machine*’ (SVM) in order to get feedback of negative or positive sentiment from the reviews.

Project Plan



Airline Reviews

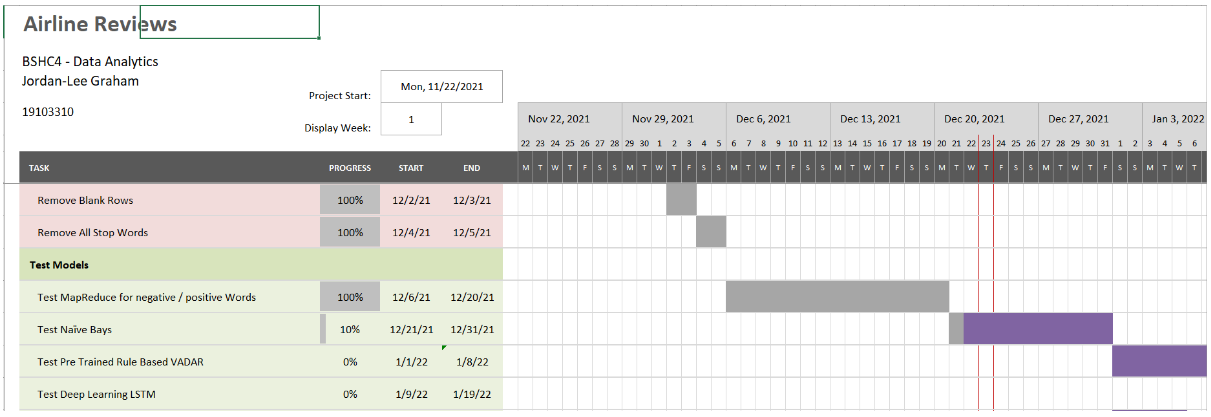
BSHC4 - Data Analytics
Jordan-Lee Graham
19103310

Project Start: Mon, 11/22/2021
Display Week: 1

TASK	PROGRESS	START	END
Gather Data			
Research Datasets	100%	11/22/21	11/25/21
Manually Review Data	100%	11/25/21	11/27/21
Pre Processing			
Eliminate Pre-Existing Sorting	100%	11/27/21	12/1/21
Shuffle Dataset	100%	12/1/21	12/2/21
Remove Blank Rows	100%	12/2/21	12/3/21
Remove All Stop Words	100%	12/4/21	12/5/21

The first phase of planning for this project will be the initial research and data gathering. There will be a need to identify datasets which contain text of a customer’s review for an airline which can be extracted from the dataset and used for implementing a sentiment analysis model on. I also wish to manually review any datasets found to ensure the quantity of data available as well as the fields available within the dataset which may be of use or not of use.

The second phase of this project will involve the pre-processing of the dataset identified. These are the steps performed in order to get the data into ‘working’ condition i.e., processing through a model. For this stage there will be work on eliminating any pre-existing sorting within the dataset such as alphabetical sorting of names, scores etc. To do this, a shuffle of the rows from the dataset using Python with Pythons built in Panda's library, by converting the data to a data frame, using a panda's function to shuffle the rows of data and saving the new data frame to a new excel file.



Airline Reviews

BSHC4 - Data Analytics
Jordan-Lee Graham

19103310

Project Start: Mon, 11/22/2021

Display Week: 1

Nov 22, 2021 | Nov 29, 2021 | Dec 6, 2021 | Dec 13, 2021 | Dec 20, 2021 | Dec 27, 2021 | Jan 3, 2022

TASK	PROGRESS	START	END
Remove Blank Rows	100%	12/2/21	12/3/21
Remove All Stop Words	100%	12/4/21	12/5/21
Test Models			
Test MapReduce for negative / positive Words	100%	12/6/21	12/20/21
Test Naïve Bays	10%	12/21/21	12/31/21
Test Pre Trained Rule Based VADAR	0%	1/1/22	1/8/22
Test Deep Learning LSTM	0%	1/9/22	1/19/22

9.2 Reflective Journals

October

The month of October involved coming up with a project idea which I felt could be explored and used as a basis for my final year project. After some thought and research, I looked into the various roles involved in the different industries I found interesting. One of the main areas I find particularly interesting was the aviation industry. Due to the highly technical nature of this industry I decided to look into the various roles required in order for this industry to function so well every day, all over the world. One of the main roles which caught my interest was that of a flight dispatcher, who was a specially trained person to analyze data in real time and use their knowledge of aviation, weather, physics and flight costs in order to determine the most efficient, safe and cost-effective flight paths for their airline's aircraft throughout the day. This sparked my interest due to the human requirement of analyzing data and I figured this would be an excellent area of study.

I had a meeting with Frances Sheridan of The National College of Ireland in order to speak about the project and gauge her thoughts on weather this would be an appropriate focus area. One of the main concerns Frances had in our meeting was that I was attempting to create an artificial Intelligence air traffic controlling system. This is an honest mistake and it raised the concern I will have to detail exactly what the role and function of a flight dispatcher clearly as to avoid this misunderstanding for others who may be looking at the project and the idea who may not be as familiar with the aviation industry. Detailing and explaining the project well and concisely became one of my first main priorities. Luckily this concern was risen early as after my meeting with Frances I was given the green light to pitch this project idea as part of my project video pitch which was due later in the month.

For my project pitch I described what my project does – "*My project idea is to virtually recreate the role of a flight dispatcher and attempt to improve upon the decision making and risk assessment process involved in this role. Professionals in this occupation currently use multiple sources of information such as flight lists for the day, weather radars, terrain maps and fuel consumption calculators in order to determine the most efficient and cost-effective route, as well as the safest possible route to establish a flight plan for the pilots of the aircraft. My aim through this project is to gather this information from various real time data API's and present it in one system as opposed to multiple radars. I plan to use the information gathered to predict ahead potential hazards the flight may encounter along particular routes which would aid the decision-making process as well as cut time spent analyzing the data for the dispatcher.".*

I described why the project would be challenging – "*This project is challenging because it will require knowledge in aviation, meteorology and cartography in order to program it correctly to recognize the different scenarios a flight may encounter. For example, we could have turbulence predicted at a certain point in the flight because an aircraft is flying over an elevated terrain which would flag on the system as "mechanical" turbulence which would be different from "thermal" turbulence which would be experienced when flying over somewhere the ground temperature is quite hot. These things will have to be identified and considered in order to properly prepare a risk assessment for a flight.".*

Then I described who the project was targeted for, why it should be attempted and what made it different – "*The project is for aviation enthusiasts as well as the aviation industry. It is an industry I would personally like to work in someday so I feel this project could benefit myself also in the future when applying to development jobs in this industry. I believe this should be attempted because I cannot find anything similar which is currently used in industry. The main protocol at the moment is for flight dispatchers to use multiple radars to analyze the information themselves. I believe if the project can effectively flag potential issues on potential flight routes it could be improved upon and further studied as part of a master's project in the future. Using different streams of knowledge such as meteorology, cartography and aviation in order to create a risk assessment. I am also attempting to improve upon a very important industry and a high stress role within this industry. I will also be combining the use of dynamic data such as weather forecasts with static data such as terrain maps.".*

Shortly after the submission of our project video pitch, we had received a list containing our academic supervisors for the project. My academic supervisor was William Clifford. I reached out via email to introduce myself and query weather our project ideas had been accepted or not. William replied soon after and informed me that himself and another supervisor had viewed my video and that my idea had been accepted with some additional changes to be made. William informed me I would find out during the week when we would have these modifications passed onto the students.

November

At the beginning of November, we received feedback on our initial project proposal documents. The takeaway from the feedback was that the reviewers were overall satisfied with my initial project idea. A main concern of the review was that the scope of the project needed to be reduced in order to improve the possibility of completing the project in the time allowed.

My project idea shifted from a focus of overall flight planning, to focusing purely on the analyzation of data for potential turbulence. For this I would need to find data which shows evidence of turbulence being reported during specific weather conditions. Searching online for potential datasets containing this information, I discovered it was not data recorded or available easily online. One example I did find however, was an API provided by Boeing the aircraft manufacturer. This API consisted of weather conditions and turbulence reports although, it required special granted permission from the company in order to use in applications. I sent an application through the company's website to enquire about use of the API however, I am still yet to receive a response.

During the meantime, I began work on gathering a list of resources of possible datasets and APIs which could possibly be used for the project. A list was compromised of weather-related API's including historical, current and potential future weather conditions, a land elevation API, real time flight data and status reports, a dataset of aircraft characteristics and some research articles on turbulence in aircraft.

After discussing these resources with my project supervisor William Clifford, we came to the conclusion that the data required for this type of project was too difficult to access or source, to commit to the project. The premise was that I would use already recorded data on conditions which lead to turbulence in aircraft and comparing them with real time information provided by flight routes and live weather information. Using clustering analysis, I would attempt to plot new information against previously recorded information in order to determine potential turbulent conditions happening in real time. I began work on building a prototype application which took live flight data and weather conditions and returned a report on the weather conditions for the flights take-off and landing however, without exact geo-locational data of aircraft available and the lack of turbulence data available, it became apparent the project idea was not currently viable, or clearly possible to complete in the timeframe.

With this in mind, we shifted focus and began discussing other potential ideas to base a project around in the aviation industry. A dataset presented to me by my project supervisor William Clifford containing thousands of records of airlines and journey reviews by passengers, became a topic of interest. A suggested idea by my project supervisor William Clifford to combine these passenger reviews with a dataset of flight route history in order to discover if positive reviews left by passengers correspond with the route flown or airline.

I really like the idea of learning and performing sentiment analysis and I am happy with the project still revolving around the airline industry so this is definitely a project I wish to undertake. Over the next weeks, leading into mid-December, I plan to begin researching sentiment analysis and the different techniques involved in cleaning the datasets.

December

At the end of November, I had made the decision with the clearance of my project supervisor William Clifford, to switch my project idea to focus on new datasets with a different goal in mind. It was decided the new project would entail the use of a dataset found on Kaggle containing information on customer reviews of many different airlines. The dataset contained numbered scores for different areas surrounding the flight experience such as seat comfort, service and a yes or no field as to whether the customer would recommend that particular airline. Now, I would focus on creating a sentiment analysis which could read the airline review entered by the customer and predict the potential scores given by the customer.

Through recommendations and guidance from my project supervisor I began researching the processes required in order to perform sentiment analysis on a dataset. It was clear the first step was to begin the data pre-processing stage of the project. My goal between the beginning of December to mid-December was to fully pre-process the data and have it prepared for use in a model in order to showcase this in the mid-point presentation.

Once this was clear, I began work on different strategies required for the pre-processing phase such as removing any pre-existing sorting from the dataset. The worry with pre-existing formatting was that the data would also need to be separated into different splits known as a ‘training validation, test split’ so I would have enough data to train models with, validate any results and a final split for testing the final model. Without removing any pre-existing formatting, it would be possible the entire ‘test’ split of the data which would equate to 10% of the entire dataset, could potentially only contain data on one specific airline or overall score meaning the results would be skewed or completely different, leading to an inaccurate result. To achieve this, I used Python to open the dataset and transform all contents of the excel file into a ‘data frame’ format using the Pandas library, which would save all the data as a table. Then, still using Python with Pandas and the data frame produced, I used a Pandas function to ‘jumble’ the rows of the data frame to remove any formatting. Once complete, I saved the new jumbled data frame to an excel file.

I validated the rows had been switched by opening this excel file however, noticed a series of blank rows had been saved to the excel file. Excel provides an easy workaround for this issue by allowing me to select all blank rows in the file and remove them. Once this was complete, I now had a dataset ordered randomly containing roughly 60,000 rows of airline reviews and scores which was then separated into a split of 80% for training, 10% for validation and 10% for testing. Noted, I had researched different validation splits and learned a split of ‘60%, 20%, 20%’ was also sometimes used however, I decided on the split I used due to wanting a larger portion of the split for training purposes.

After these steps I began working on the next phase of the data pre-processing and began work on removing ‘stop’ words from the data. To achieve this, I started a new Python file and used the Natural Language Toolkit (NLTK) library and its available corpus to remove 'stop' words such as – 'and', 'a', 'I', 'the'. These are words which provide no added detail or information to the dataset so it is important to remove this from the data so that we can save and read from only descriptive words or names provided in the reviews. I achieved this following recommended example programs provided by my project supervisor William Clifford. The result of this Python file was some reviews read and the stop words removed, with a JSON response of all nouns and adjectives discovered within the dataset.

The next phase was to submit our mid-point presentation showing the work we had complete to this point so I did not get the opportunity to implement this data to a model. I did however, research potential models which are commonly used in sentiment analysis and some of the models I aim to implement over the coming weeks – leading into January – include VADAR Lexicon and rule-based sentiment analysis for the scoring of the different areas contained in the reviews. After this, I will also attempt to implement a Naïve Bays sentiment analysis on the data set I have saved in order to determine a 'positive' or 'negative' sentiment analysis.

February

The month of February we had begun our semester 2 modules, one of which included the 'Data and Web Mining' module for the data analytics stream. This module introduced some new concepts we had not yet learned and provided me with lots of new ideas of features which could be added to the project in order to enhance the work.

One of these main insights was the use of the R programming language in relation to data pre-processing and cleaning. I really enjoyed learning R and its simplicity and considered a possible approach towards using it in my project along with Python code.

Another one of the concepts which I learned about was the idea of clustering. A possible feature for my project would be to create clusters of different amenity scores such as seat comfort, food and beverages and entertainment and color them based on whether or not they were part of a positive or negative overall scoring review. By doing this I would be able to evaluate if different amenities contributed to positive or negative reviews.

During this month I also restarted my weekly meetings with my project supervisor William Clifford after the midpoint break in the year. One of the things brought forward from the midpoint submission was the possible addition of another source of data such as an API or website which could be web scraped depending on permissions available.

Through my first meeting with my supervisor William Clifford, we had created the goal of first focusing on predicting an overall score rather than different review scores for different areas of amenities i.e. cabin service, ground service. We had also outlined I would immediately begin working on implementing a basic model first and deciding on the approach of using a word to vector and passing this through a support vector machine.

We had also discussed the possibility of working on more advanced models down the line such as a recurring neural network. if I could achieve the task of completing these straighter forward models as early as possible.

I then began work on creating a term frequency – inverse document frequency (TF-IDF) approach in 'Jupyter Notebook' using Python.

After encountering many issues with accessing Python libraries from Jupyter Notebook and finding the process of storing my files through Anaconda, I decided to implement a new approach for the overall project. I created an R Markdown script which would allow me to combine both R and Python programming languages and allow for access to one another's libraries. Through R Markdown I would be able to create a HTML web document which included my code along with the output and any graphs created through the code. I like the simplicity of this rather than creating multiple separate files for each part of the project i.e., preprocessing, visualizing.

I worked on adding all of my previous work in Python to the R Markdown file and preprocessing all the data in R. After a discussion with my project supervisor, I decided on creating a word to vector classifier and pushing the resulting numerical representation of the words through a support vector machine (SVM).

Hoping to complete this in a short amount of time I currently have my sights set on attempting to implement a deep neural network such as an Long Short Term Memory (LSTM) approach in the future.

March

Throughout February until the beginning of March I had began experimenting with the use of R for implementing machine learning models within my project. After some time using R, it became apparent R requires quite a powerful setup to handle data with thousands of rows when running machine learning models.

Due to this I switched back to using Python in order to implement the models and began quickly becoming more familiar with how the process works between processing the raw data and separating it into testing / training splits. The first model I implemented on my processed airline reviews was a term frequency – index document frequency (TF-IDF) vectorizer which applied a numerical value to each word in the customer review. This produced an output of an array of words represented in numerical form. The array was passed to a support vector machine (SVM) model with the target value being the 'overall score' given by the customer in their airline review. This meant the model was attempting to predict a score from 1-10 based on the vectorized words array representing the customer's review.

The SVM model produced an accuracy of 41% which is quite good considering the range of values it has to account for and target prediction variable. I am happy with this result currently as it comes close to 50% correct when having to predict between 10 values.

Next, I will focus on parameter tuning as well as building on the final submission documentation. A big factor I want to begin focusing on now in the final month is the research portion of the project and being able to clearly identify good values from the results and compare them with the research already done.

A very useful aspect of this month was the use of a Kanban board suggested to me by my project supervisor. By separating the functional coding aspect of the project work and the documentation in to-do -> completed boxes, I have been able to keep better track of my progress of which parts of the project need to be focused on.

______Jordan Lee Graham______________

Signature