**AIRLINE REVIEWS AND RATINGS**

Brandon Liu, Max Li, Aidan Frantz, Yuchen Cai

CSSE/MA415 Machine Learning

17 May 2024

**ABSTRACT**

The airline has been a common way to travel, and people’s recommendations play a large role in which airline they pick. In this paper, we will apply various models to predict whether or not a trip was recommended on a dataset of existing reviews. We applied a diverse set of classifiers including K-Nearest Neighbors (KNN), Random Forest, Gradient Boosting, and Support Vector Machines (SVM) to evaluate their performance in predicting airline reviews. We measured the accuracy of each model value and the analysis of faulty predictions. During the preprocessing stage of our project, we utilized word processing which made use of a bag-of-words algorithm. The Bag-of-Words algorithm allowed our team to convert textual data into boolean features that associated specific words with negative or positive reviews. Training and testing different models with, and without, the extra features created by the team’s word-processing method outputted different results. The results of these models gave our project the potential to automate sentiment analysis for airline reviews, providing insights into customer satisfaction and areas of service improvement.

# 1. INTRODUCTION

Airline Customer Satisfaction is a critical factor in assessing the success of any airline company. With the rise of online review platforms, customers can now share their experiences online and these reviews provide a great way for other customers to select airlines based on their budgets. Our interests arise from our past flight experience and the need to occasionally take the flight. Therefore, we decided to focus on the airline reviews to extract the most important features that matter most to customer satisfaction to help us better understand and select flights in the future. Additionally, this project can help airlines understand what factors drive positive or negative reviews, allowing them to improve specific areas to better serve their customers. Ultimately, better customer satisfaction leads to increased loyalty, positive word-of-mouth, and higher revenues. However, this problem is challenging due to the high-dimensional nature of the data involved. The dataset contains several word-based features which made us put effort into data pre-processing.

Our proposed solution applies machine learning algorithms to capture complicated patterns and relationships within customer feedback data. By training models on large datasets of airline reviews and ratings, we aim to develop a machine learning model that can accurately predict if future flights can satisfy customers based on historical customer ratings and text reviews.

One of the most interesting things about our solution is the incorporation of traditional natural language processing techniques to extract insights from the unstructured text data in the customer reviews. Specifically, we experimented with the word-processing approach where each word in the review text was treated as a separate feature, set to True or False depending on whether that word appeared in the original review text. Then, we trained a Random Forest Classifier using just these word-based features extracted from the reviews and were able to achieve 84.5% accuracy in predicting whether a review recommended the airline or not.

The inputs to our model are a dataset containing 3290 records with 14 different features, including categorical data, numerical ratings, and most importantly, the unstructured text content of the customer review itself. The output is a binary classification - whether the model predicts the review as recommending the airline or not recommending. We have framed this as a supervised classification learning problem, training our models on the provided labeled dataset of 3227 unique user reviews. Regarding simplifying assumptions, we chose to drop the "Users Review" feature column initially, instead extracting a new set of word-based features from that text data through our word-processing technique, as described earlier. We also performed standard data preprocessing like handling missing values and one-hot encoding of categorical variables.

# 2. LITERATURE REVIEW

In a very similar project that worked on Airline Flight Delay Predictions, they also used very similar models to the ones that we used. They used the same algorithms that we used: Decision Tree, Random Forest, Gradient Boosted Tree, KNN, and SVM. The model that came out to have the best accuracy was the Decision tree with an accuracy of 0.9778. [1]

In another study about aircraft passenger satisfaction [3], the random forest came out to be one of the best algorithms to be used on a feature subset. They received an accuracy of 0.963; however, the study found that the passenger satisfaction survey in the data set was not sufficient.

For the study regarding Airline Data Analysis [2], it did not have very much inherent use for our project. It did not include intensive use of machine learning but it provided good insights into the direction we took for our project. It also served as a decent baseline, as the RandomForest classifier was able to see similar results as the AI model they used in their study.

The research study about sentiment analysis of customer feedback and reviews for airline services[4] provided us with good insights into how to utilize the text features inside the dataset by applying a bag-of-words method of associating the words for negative reviews. The article also goes into the same algorithms our project used including KNN and Random Forest.

# 3. PROCESS

## 3.1 Data Source

Our dataset is found on the Kaggle website[5] and was initially extracted from the online airline review. The dataset contains 14 features and one boolean prediction target on whether the reviewer would recommend the airline based on experience. Thus, our project focuses on a binary classification task involving two distinct classes. Specifically, it contains 3290 data records, with 6 categorical descriptions and 8 numerical ratings ranging from 1 to 5, with the possibility of missing values. Additionally, there are 1274 reviews categorized into recommendations and 2016 reviews that are not recommended, with about 39% and 61% for the corresponding portion. With the target composed of these two tags, the baseline accuracy for this dataset is 61%.

## 3.2 Preprocessing

User reviews are text in sentences that the user typed to evaluate or comment on the flight experience. With almost uniquely identifiable user reviews, either keeping them with appropriate processing or dropping this feature became the main decision being made during data preprocessing. By keeping the user review, the model would be able to extract significantly meaningful words for prediction, while the risk of the absence of language processing might cause unexpected interpretation. The standard way is to drop the user reviews and decide whether one-hot encodes other categorical features or drops as well. The empty entries presented in the numerical features are filled either with a number with a valid value or an out-of-range value.

After one-hot encoding all the categorical features after dropping the user reviews feature, there are 1972 features including numerical features.

### For some of our work, we also used a preprocessed version of the Route feature. The approach to processing Route was to split words on the word “to”, then split on the word “via”, fix the ordering, and use the lists of countries as new features.

## 3.3 Feature extraction or engineering

To process the users' reviews, we used an approach that combined bag-of-words and one-hot-encoding. We first split the reviews into spaces that way they would be lists of words. Then, we dropped all characters that weren’t alphanumeric. Afterward, we iterated over the lists of strings, converting all of the words to lowercase, and then, we used all of the reviews to create a set of words, which was used to make new features for all of the words. Finally, we iterated through each review, setting features to True for each word that the review contained, and then dropped the review feature.

## 

## 3.4 Classifiers

**Linear Regression**

The linear regression is that, for certain features and the target, the linear regressors are used to compute the relationship between that individual feature and the target. Together with bootstrapping, the features that are not significant would be indicated and could then be dropped. Linear regression is mainly used to find significant features or feature engineering.

**Random Forest**

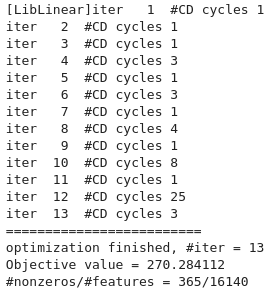
By randomly selecting the features to build multiple decision trees, the random forest would vote for the result based on the mode. It can overcome the flaw of one single decision tree, where the variance value for the random forest model would be high.

**Support Vector Machine**

By boosting the features in the extra-dimensional space, the hyperplane will be used to separate the records with different tags. The SVM is not well interpretable and understandable while involving multi-dimensions, but the result will be mapped back to the feature space. Due to the complex algorithm involved, it would require some effort to avoid underfitting or overfitting by tuning hyperparameters.

**L1 Logistic Regression:**

L1 logistic regression was a useful way to reduce the number of features. For the merged dataset involving the one-hot-encoded category columns and the review word columns, L1 logistic regression was able to remove roughly 98% of the features, as shown in **Figure 1**.

****

**Figure 1: Demonstration of L1 logistic regression on the full dataset (words and all other features)**

**PCA**

PCA is a useful method to further reduce the number of features, using linear algebra to capture as much of the variability as possible in as few features as possible. In several runs using the full dataset already reduced from L1 logistic regression, PCA was able to further capture more than 70% of the variability of the features in less than a third of the original number of columns.

**K-Nearest-Neighbors**

KNN classifier is the technique of using the mode of the nearest K neighbors to predict the class. The training effort for KNN is almost nothing. With the increase in dimension, the curse of dimensionality would occur, where the faraway distance between each pair of points would inaccurately underfit the existing set.

**Decision Tree**

The decision tree is the core component of a Random Forest and Gradient Boosting classifiers. By selecting the most significant features to split the training set and slowly making pure nodes, the tree can easily achieve low bias on the set. However, reducing variance requires multiple trees to minimize the influence of individual data.

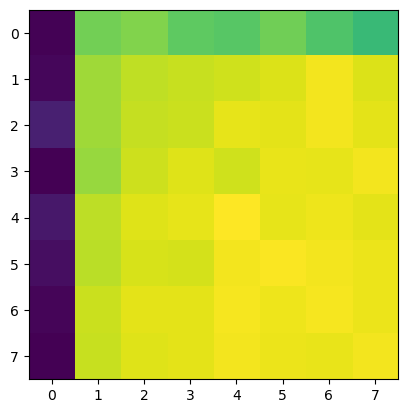
# 4. EXPERIMENTAL SETUP AND RESULTS

**Linear Regressor (without word processing)**

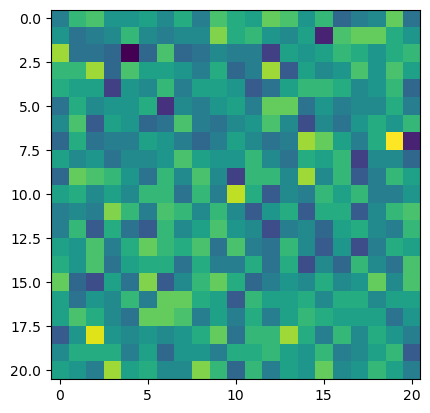
For linear regression, the parameter that is shown to be most significant for the result is seat type of First Class, with a negative relationship to “Recommend”. This is also found as a word processing result and will be discussed in Section 5, where this word would closely relate to unsatisfactory results. With bootstrap with merely a 50% confidence interval, the 6 values that would affect the result with high confidence are all those of numerical values.

**Random Forest (without word processing)**

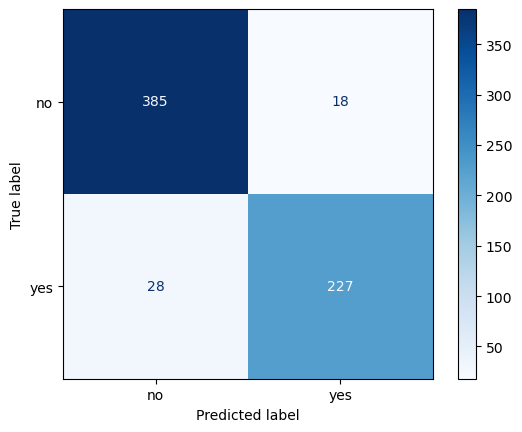
For the random forest classifier, the selection of the parameters for the n\_estimator and the max\_depth would affect the result on a large scale. With too small of a scale for the hyperparameter selection, there is little evidence of the relationship between accuracy and those hyperparameters.



**Figure 2**: random forest heatmap, with the x-axis being the max depth and the y-axis being the number of estimators. The values for both scales are 2,5,10,20,25,50,100.



**Figure 3**: Fine hyperparameter tuning accuracy heatmap, centered at number of estimators of 50, max depth of 100, and the scale for each grid is 1. The fine structural randomness is achieved by the property of random forest, and hence unable to reveal the relationship between hyperparameters and accuracy in this range.

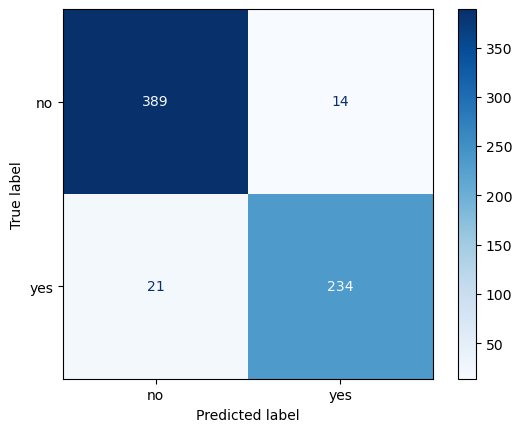


### **Figure 4**: confusion matrix for random forest, with an accuracy of 94.07%.

**SVM (without word processing)**

For the support vector machine classifier, while using all the nearly 2000 features with one-hot encoded processing, the optimum accuracy with this classifier is 93.92%.

However, with only training on those 8 numerical values, the best accuracy is boosted to 94.68%.



### **Figure 5**: confusion matrix for SVM, with an accuracy of 94.68%.

Furthermore, what numerical value those empty grids are filled with would also affect the accuracy. When filling the value with an existing value in the original dataset, selected from 1 to 5, those data will post influence towards records with valid ratings during training. For SVM, the ideal strategy would be using the non-integer mean value or out-of-valid range value. It would not be hard for SVM to find a hyperplane to ignore the feature with value 0, or the not given value somehow related to the satisfactory.

**KNN (without word processing)**

We tested the KNN classification model on our project to see the different kinds of accuracy reading we could receive from this classification model.

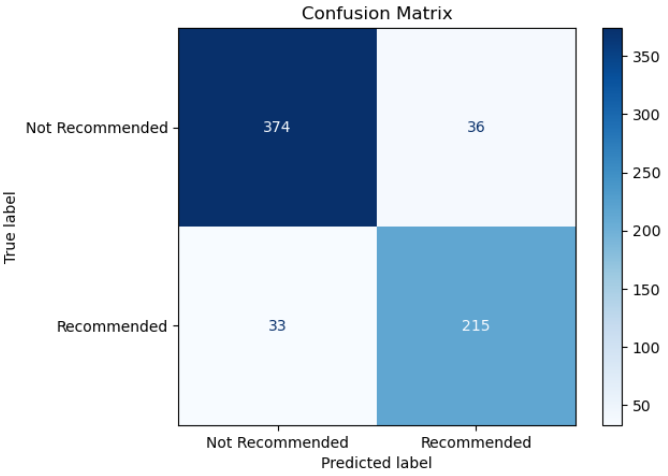
### **Table 1**: Accuracy using KNN classification

| **k** | **Accuracy** |
| --- | --- |
| 1 | 88.1% |
| 4 | 91.9% |
| 6 | 92.1% |

By optimizing the hyperparameter for the KNN classifier model, we found that using a k value of 6, allowed our model to receive a 92.1% accuracy reading.

**Decision Tree (without word processing)**

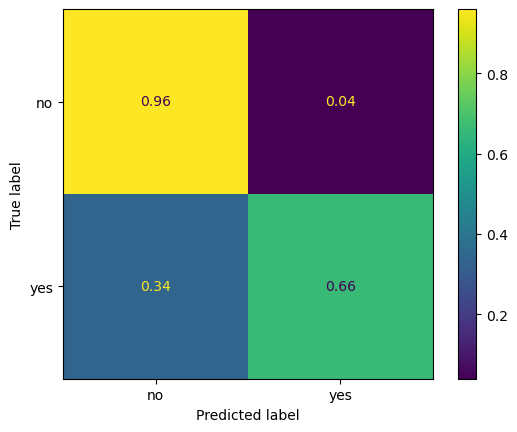
Utilizing the Decision Tree model as well. we were able to receive an accuracy reading of 90%

**Figure 6**: confusion matrix for Decision Tree

In addition to getting a 90% accuracy reading, the model also received a recall of 0.91, precision of 0.92, and f1-score of 0.92.

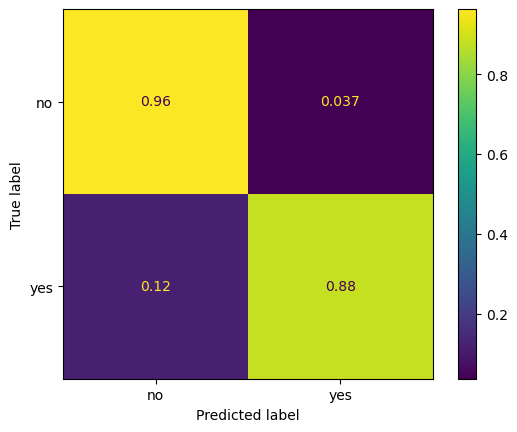
**Random Forest Classifier with just words**

To prove the usefulness of using words as features, we tried making predictions using a Random Forest classifier that only used the new word features. It achieved about 85% accuracy, which is a good deal over our baseline, and shows that there is some correlation between the word features and recommendations. Notably, the predictor performed very well for no but performed only slightly better than 50/50 for yes, showing that the words in the reviews are better for estimating negativity than positivity. This model, as well as any following models involving the words, were based on an 80/20 train/test split.



**Figure 7:** Random Forest Classifier results on just words (normalized over rows)

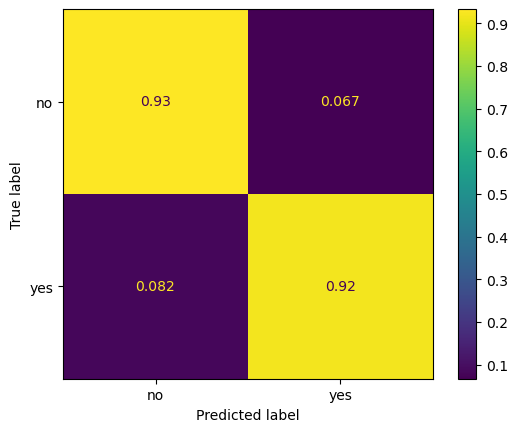
**Random Forest Classifier on L1 reduced full dataset**

****

**Figure 8:** Random Forest Classifier results on the full dataset (including words) after L1 LR feature reduction (normalized over rows).

The RandomForestClassifier used scikit-learn default parameters, except for n\_estimators=525. The top row is normalized over 403 values, and the bottom row is normalized over 255 values.

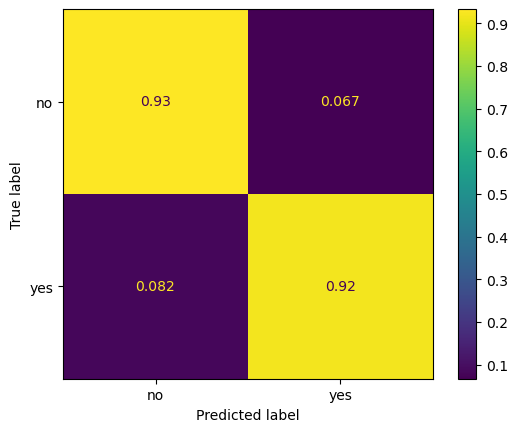
**Gradient Boosting Classifier on L1 reduced full dataset**

****

**Figure 9:** Gradient Boosting Classifier results on the full dataset (including words) after L1 LR feature reduction (normalized over rows).

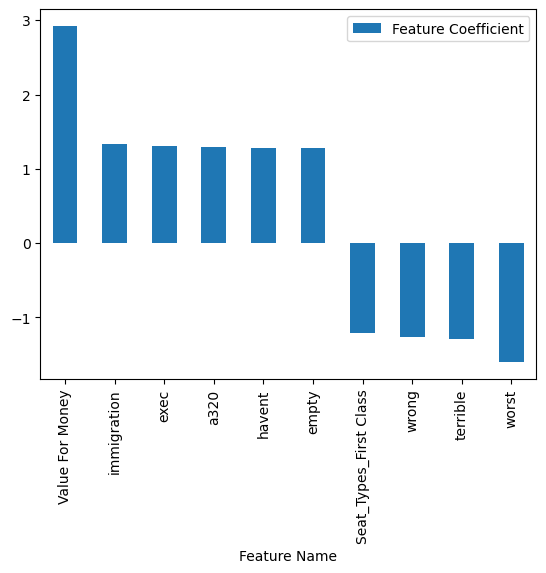
The GradientBoostingClassifier used scikit-learn default parameters except for loss='log\_loss', learning\_rate=0.78, and n\_estimators=935. The top row is normalized over 403 values, and the bottom row is normalized over 255 values.

**L2 Logistic Regression on L1 reduced full dataset**

****

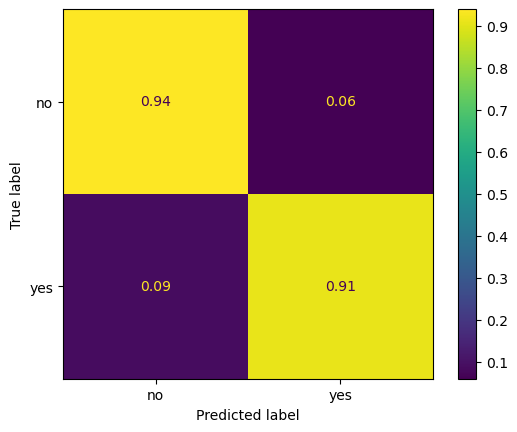
**Figure 10:** L2Logistic Regression results on the full dataset (including words) after L1 LR feature reduction (normalized over rows).

LogisticRegression used scikit-learn default parameters. The top row is normalized over 403 values, and the bottom row is normalized over 255 values.



**Figure 11:** L2 Logistic regression top 10 most important features based on absolute value.

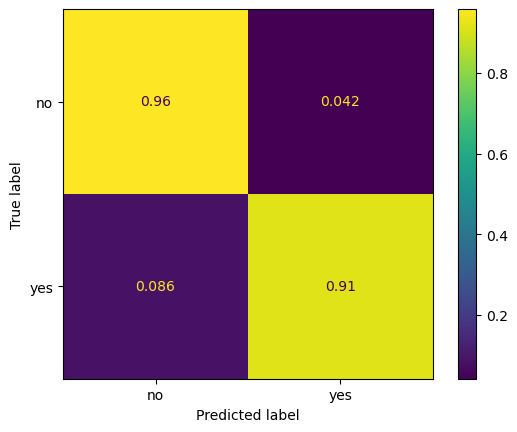
**KNN Classifier on L1 and PCA reduced full dataset**

****

**Figure 12:** KNN Classifier results on the full dataset (including words) after L1 LR feature reduction and PCA feature reduction (normalized over rows).

The KNNClassifier used scikit-learn default parameters except for n\_neighbors=14, weights=’distance’, and p=2. The top row is normalized over 403 values, and the bottom row is normalized over 255 values.

**SVC Classifier on L1 and PCA reduced the full dataset**

****

**Figure 13:** SVC Classifier results on the full dataset (including words) after L1 LR feature reduction and PCA feature reduction (normalized over rows).

The SVC used scikit-learn default parameters except for C=2. The top row is normalized over 403 values, and the bottom row is normalized over 255 values.

# 5. DISCUSSION

The analysis of our selected dataset showed to be effective for many machine learning methods. Many classifiers have an accuracy of over 90%.

The numerical values are helpful for those models without user reviews and have shown to be much more effective than the model using all the numerical and categorical features except user review.

On the contrary, the users’ rating is occasionally opposite to that shown by the “Recommend”. This is the irreducible error for using those numerical features, while the positive relationship is built during training.

For most of the confusion matrices, their similarities are that there are more false negatives than false positives. Given the fact that there are more negatives than positives for our records on Recommended, this tendency to predict false would likely yield better predictions, especially for those uncertain cases.

The L2 logistic regression feature importance results were included because they are by far the most interpretable of all the models, and the L2 logistic regression saw decent overall performance. Value for Money is the most important positive feature, which makes sense because people who think they are getting a good deal will generally be happier with their experience. People who sat in First Class and who included words like “terrible” and “worst” generally didn’t recommend their flights. Generally, this indicates that people want to have a more positive experience during their flights and that people likely don’t think first class is worth it as an experience.

Generally, the full dataset models saw accuracy between 90 and 95%, often around 92 to 93%. This is somewhat disappointing, as the models without the words were able to achieve results almost as good, often around 90 to 92%, which means that doing the extra work for the bag-of-words processing might not be worth it. However, one thing worth trying that might be especially good for algorithms like L2 logistic regression would be feature engineering so that features like “good” and “not good” could be analyzed separately, which might lead to a further accuracy boost. However, that would likely need to be done before feature reduction and thus would end with a dataset with possibly millions of features, which would take too long for our computers to analyze.

A record that the L2 logistic regression model classified incorrectly for example would be review 1013, which was a trip from London to Budapest. The reason why is immediately obvious, it had a value for money of 5, which is the most important feature in the model (as shown in **Figure 11**), so it immediately classified it as ‘yes’ instead of the correct answer, which is ‘no’. This is further evidenced by looking at review 1391, which was a trip from Malaga to Gatwick, had a value for money of 5, and was also classified, this time correctly, as ‘yes’. Unfortunately, deviations such as review 1013 are natural and form a part of the irreducible error of our dataset, and as such can’t be corrected, placing a hard cap on the testing accuracy of our work.

# 6. CONCLUSIONS AND FUTURE WORK

Some short-term possible future work would involve the combination of different categorical features. It would be able to extract the relationship between those features, with the risk of exponentially increasing program runtime. However, this form of feature engineering could greatly increase the accuracy.

Some long-term possible future work would probably involve the use of more advanced machine learning models, or potentially even deep learning models, to glean even more information from the reviews. Especially being able to consider sentence structure, capitalization, and non-traditional characters like emojis would go a long way towards making it possible to glean more information from the text, versus the relatively simple methods we used here.

Airline companies can identify many potentially helpful features from predictive analysis, but one feature was consistently the most important, “Value For Money.” This supports the basic economic idea that what people want most is a good deal, so if a client were to offer a poor airline service, people are far more willing to tolerate it if it’s cheap and if a client offers a more expensive airline service people expect better services.

**7. KEY CHALLENGES AND LESSONS LEARNED**

The most significant challenge is the runtime due to the large data size. Hence, the feature processing and selection would be necessary to boost the running speed of the program within rational time.

One of the lessons we learned is that even with the absence of an AI model, parsing the sentence and treating each word as a feature would give satisfactory results, and would be more subjective for satisfaction from a passenger perspective. The phrase “not good” can be handled with a model that would directly combine multiple features. However, the relationship of words within a sentence cannot be analyzed without a more comprehensive process.

Word process with one-hot encoding would be feasible for prediction. From this perspective, each word in the sentence would be a feature for all data records.

Another lesson learned from this project is that the increase in features does not necessarily mean we are using significant features. The exponentially increased one-hot encoded features also affect the running time significantly, potentially causing overfitting issues.

We also learned that many of the scikit-learn models initially pick an algorithm that is not optimal. For example, the L1 logistic regression model will initially use the ‘lbfgs’ solver and will either take a long time for L1 operations (we tried waiting and it ran for several minutes with no results) or just fail, due to incompatibility. Switching to the ‘liblinear’ library however allowed it to run in less than 10 seconds, which is a significant improvement. Many models, from KNNClassifiers to SVCs have different algorithms, and if it’s taking a long time to execute it’s a good idea to try and change the algorithm.

**8. REFERENCES**

[1] Tang Y . Airline Flight Delay Prediction Using Machine Learning Models. ACM Digital Library, pp. 151-154, October 2021. <https://dl.acm.org/doi/pdf/10.1145/3497701.3497725>

[2] Madhavilatha N., Shireesha B., Anilkumar C. Airline Data Analysis, 2019. IJSRCSEIT, Volume 5, Issue 1. ISSN: 2456-3307. DOI : 10.32628/CSEIT19514. <https://ijsrcseit.com/paper/CSEIT19514.pdf>

[3] Jiang X, Zhang Y, Li Y, Zhang B. Forecast and analysis of aircraft passenger satisfaction based on RF-RFE-LR model. Sci Rep. 2022 Jul 1;12(1):11174. doi: 10.1038/s41598-022-14566-3. PMID: 35778429; PMCID: PMC9247921. July 2022. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9247921/>

[4] PATEL, A., OZA, P., & AGRAWAL, S. Sentiment Analysis of Customer Feedback and Reviews for Airline Services using Language Representation Model. Procedia Computer Science, (2023). 218, 2459–2467. https://doi.org/10.1016/j.procs.2023.01.221

[5] ANANDSHAW2001. Airlines Reviews and Rating. https://www.kaggle.com/datasets/anandshaw2001/airlines-reviews-and-rating/data