Project Final Report

Twitter Airline Sentiment Analysis

Soudeh Nilforoushan, Ghazaleh Noroozi
snilforo@uwo.ca, gnoroozi@uwo.ca

The University of Western Ontario
Department of Computer Science

1 Abstract:

Social media get more attractive nowadays and public and private ideas have been spread through social media. Twitter is one of the social media which is so popular and it plays a significant role in the marketplace. Developing a program to analyze customers' sentiment can be useful for companies to track users' ideas not only improve their quality but also compete with other companies. The objective here is to analyze how travelers mentioned their feelings on Twitter in February 2015. We compare 7 methods including Decision tree, Random forest, Bagging, Multinomial naive based, Support vector machine, K-nearest neighbor, and neural network using the one-hot encoder. At the end, we compare all these methods[1].

2 Introduction:

Customer feedback is so important for improving the quality of the services for companies. Airline industries try to collect data through some questioners form, but these forms may consist of some incorrect data since it is time-consuming and customers may not take it seriously, also it needs a lot of manpower to analyze these data. However, Twitter is a popular social media and more than 100 million people use Twitter, about one billion tweets, are tweeted daily. So, Twitter is a golden source for the airline industry to get feedback from their customers. Once the data is gathered from Twitter we have to clean the text of customers. In this paper, we go through 7 different methods and we compare different techniques[1]. The paper is organized as follows. In section 3 we talked about how we extract the dataset, in section 4 we explained how we clean and pre-process the data, in section 5 we briefly explained the method that we used, in section 6 we provided the results of our work and we compare them, in section 7 we bring the conclusion and finally we talked about the future work.

3 Data Extraction:

In this project we use the Kaggle dataset, US airline sentiment, this extracted dataset is for the six airline companies: United, US Airways, Southwest, Delta, and Virgin America. Tweets are a combination of neutral, negative, and positive sentiments.

4 Data Reprocessing

we have to clean the text into an understandable format for the machine. Tweets include many leaks and problems so we apply the following techniques to clean the data:

- We removed stop words like "the flight" should be "flight"
- We removed the Hashtags.
- We removed URLs.
- We removed the usernames.
- We removed emoji es.
- We applied lower case on all characters

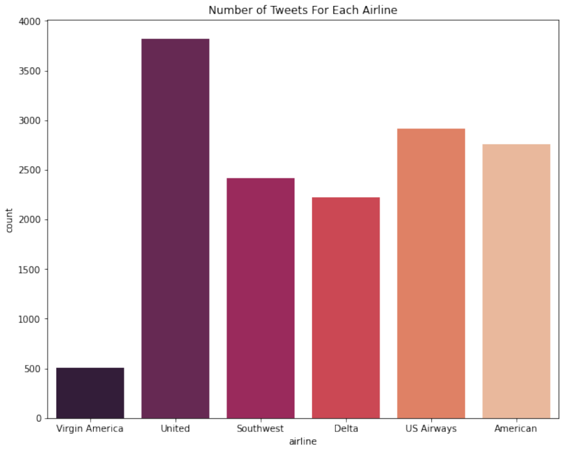


Figure 1: Airlines

- We applied decontraction like ”wasn’t” should be ”was not”.

We show in figure 2, that we divided the data set into different classes and extracted 100 most repeated words from each one of them. Then we used set difference to find the words that are specifically used to convey a certain sentiment. This is a good way to extract positive, negative and neutral words. for example, words ”amazing”, ”awesome”, ”great” are repetitively and only seen in positive context.

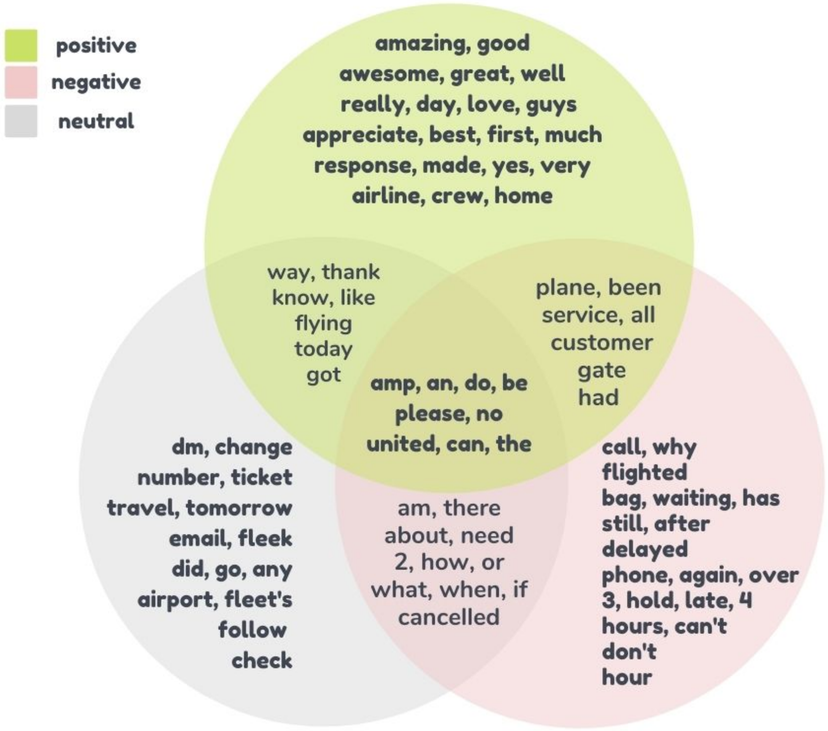


Figure 2: Top 100 Words

Sampling the Data

Sampling is a method that allows us to get information about the population based on the statistics from a subset of the population (sample), without having to investigate every individual, also when we have unsampled data we can use sampling to make it balanced. The data is unbalanced shows in figure **??** and it might cause some bias in our final result so we re-sampled the data shown in figure**??**. We use SMOTE for sampling, SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space, and drawing a new sample at a point along that line.

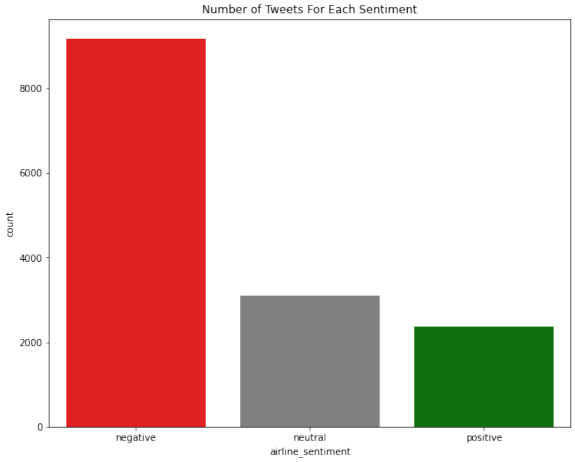


Figure 3: Imbalanced Data

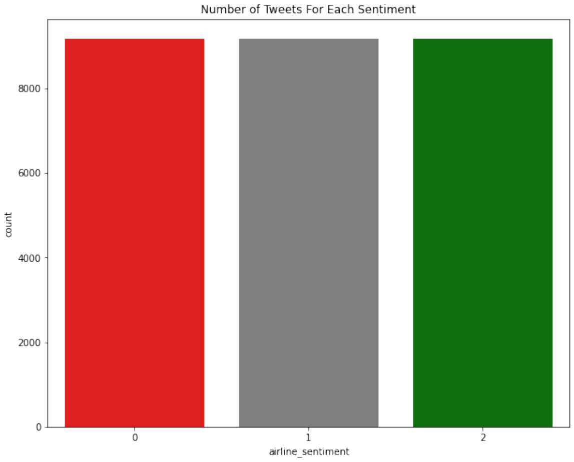


Figure 4: Balanced Data

Splitting the Data

We split the data into training and test sets so that we could determine the effectiveness of our models. We set the test size to 0.2. We also split our training set into training$_{v}alidationandvalidationsets.$

5 Methodology:

We have exploited 7 different machine learning models, including decision tree, random forest, multinomial naive bayes, bagging with multinomial naive bayes as its base classifier, support vector machine, K-nearest neighbor, and a neural network. We apply these models on the data, tune them and analyze their performance.

5.1 Classification Techniques

5.1.1 Decision Tree

The decision tree is a simple tree-based model. Decision trees extract patterns based on simple but effective feature selection in large databases and they're intuitively interpretable. These are the reasons behind their extensive use in analysis and predictive modeling. Constructed models of decision tree models are more interpretable in comparison to other pattern recognition techniques. Apart from the main target that is pattern recognition and classification, decision tree can be used as a data analysis technique. Information gathered by looking at a fitted decision tree can help identify important features and inter-class relationships can be used in future experiments. [2]
On the other hand, they might be too simple and hence will overfit the data very easily. Also, a slight change in the data can make the tree structure unstable. To avoid overfitting the data we can put limits on further branching the tree, keeping the model more simple and general.
We used sklearn library's DecisionTreeClassifier algorithm. It supports two main variables to help avoid overfitting:

- max_depth: The maximum depth of the tree.
- min_samples_leaf: The minimum number of samples required to be at a leaf node.

Without these restrictions, we would have a deep tree that has low bias and high variance, and hence can learn irregular patterns and overfit. [3] We assigned 2 and 55 to these variables respectively. We used the try and error technique on validation data to achieve these numbers. This way it's faster, simpler, more general, and more accurate on the test data.

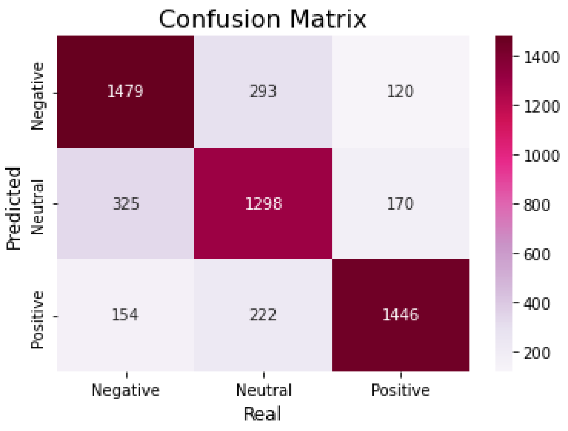


Figure 5: Decision Tree Confusion Matrix

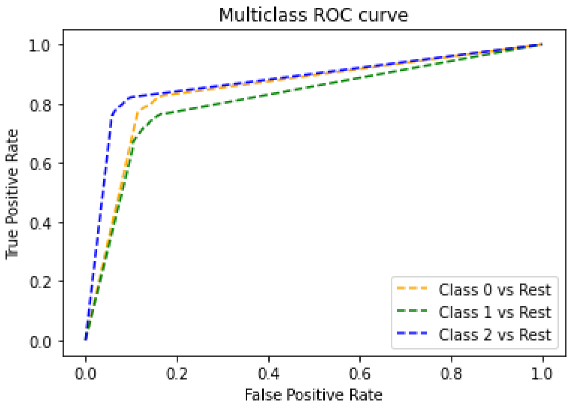


Figure 6: Decision Tree ROC Curve

5.1.2 Random Forest

Random forest is an ensemble decision tree classifier that combines prediction of each tree and outputs them as a prediction of its own. Each decision tree is built by a subset of the training data and for this matter, it is better than just bagged trees. In random forest, trees are decor-related when they are built-in. [3] It uses randomization to improve the predictive and classifying ability of the model. Random forest is efficient and has high prediction accuracy, it doesn't have many parameters to tune and works well for diverse kind of applications. With these being said, a single decision tree is way more interpretable than a random forest.
We used the sklearn library's RandomForestClassifier. It has the same restrictive parameters as the decision tree, max_depth, and min_samples_leaf. Because of the randomization feature, we can use less restrictive values for these variables. You can also specify the number of trees using n_estimators variable. We assigned 20 as the maximum depth and 100 as the number of estimators as they worked well on the validation data.

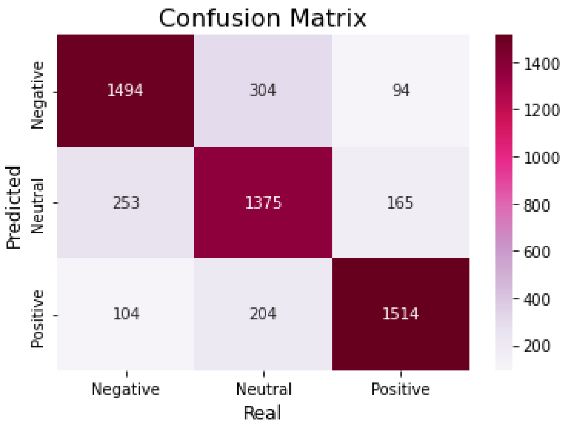


Figure 7: Random Forest Confusion Matrix

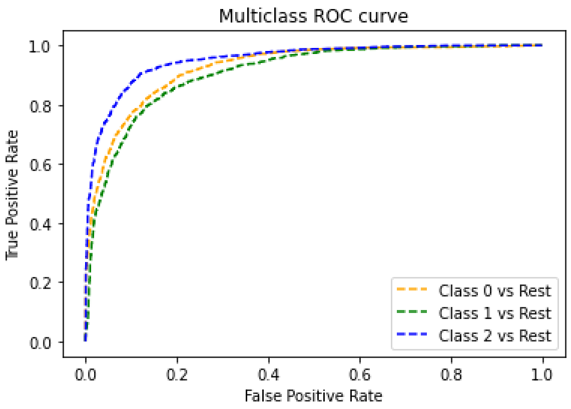


Figure 8: Random Forest ROC Curve

5.1.3 Multinomial Naive Bayes

Naive bayes classifiers are a popular family of classification models, based on the famous bayes probability theorem. A multinomial naive bayes classifier is part of this family and is considered as a baseline dominant modeling approach. It was introduced as it is usually more efficient than the multivariate Bernoulli model, which introduced language modeling in information retrieval. [4]
The naive Bayes family are specifically designed for text processing and analysis applications. They are intuitively easy to understand, simple to implement and complexity-wise efficient. They use small amount of resources. Hence they can be efficiently used for large datasets, unlike kernel-based models like SVM and Neural Networks.
We used sklearn's library MultinomialNB algorithm. The only hyperparameter it has is alpha, the additive smoothing

parameter that helps solving the zero probability issue in Naive Bayes algorithm. It was observed that by increasing this number, the accuracy on validation data gets worst and we decided to use the default value of 1 for this parameter.

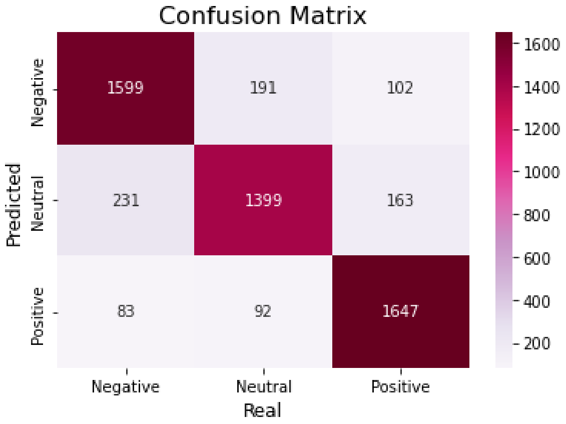


Figure 9: Multinomial NB Confusion Matrix

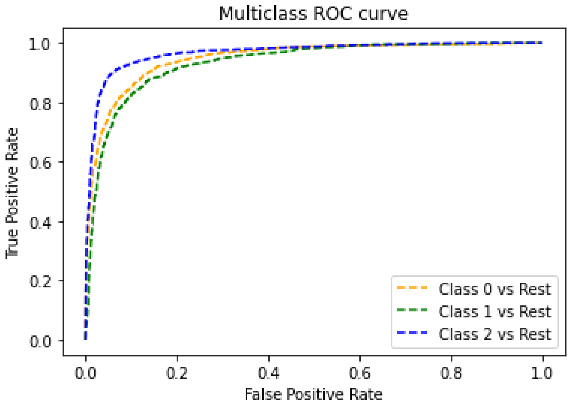


Figure 10: Multinomial NB ROC Curve

5.1.4 Bagging(Bootstrap Aggregating)

Bagging is an effective but computationally intensive model, built to improve the accuracy of unstable models or classifiers by reducing the variance of the predictor. It is very useful for multi-dimensional and large data that finding just one model that works well is almost impossible. [5]
Bagging has used The classifier that we used as the base classifier here is the multinomial naive bayes. We used sklearn library's BaggingClassifier algorithm. It has an important parameter that specifies the base classifier. The other important hyperparameter is n_estimators, which is the number of base estimators to aggregate their outcomes at the end. We assigned 100 to this parameter as it performed relatively good on the validation data.

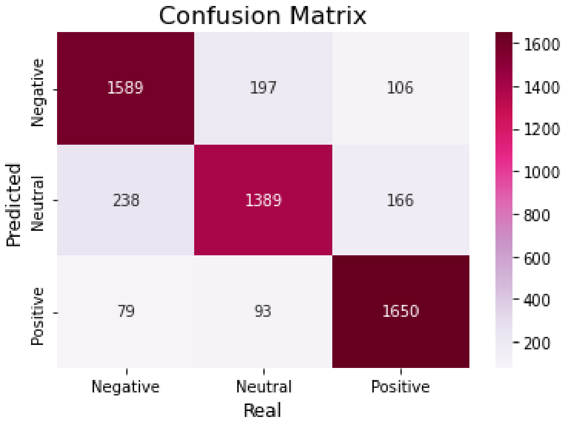


Figure 11: Bagging with MNB Confusion Matrix

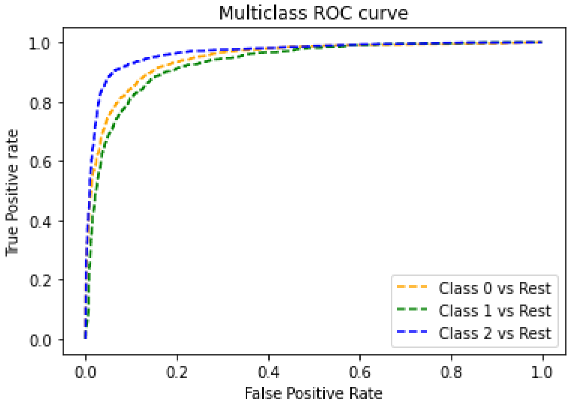


Figure 12: Bagging with MNB ROC Curve

5.1.5 Support Vector Machine(SVM)

One of the last models we applied to the airline tweet data is a support vector machine with the linear kernel. SVM is a very capable classification algorithm that is trying to maximize the classification boundaries. The SVM boasts a strong theoretical underpinning, coupled with remarkable empirical results across a growing spectrum of applications.[6]
we used the sklearn library's SVC algorithm. SVM takes a significantly longer time to train on the data, so we tried regularization to make the model simpler and hence, faster. The regularization variable in this model is C, which is inversely proportionate to the rate of regularization, and by reducing it from the default number C=1, more

regularization is applied and the model would be simpler and faster. But as shown in the table the accuracy drops heavily by doing that and we decided not to add regularization to the loss function.

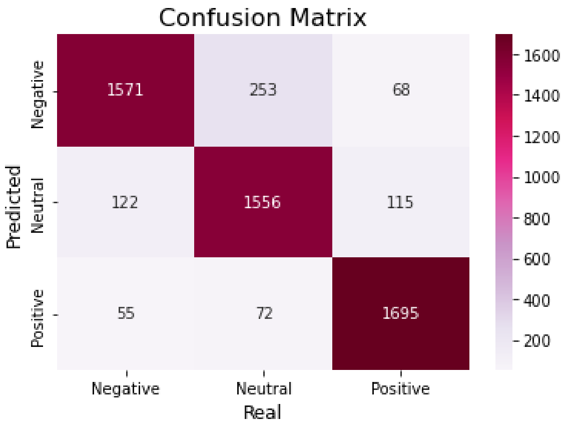


Figure 13: SVM Confusion Matrix

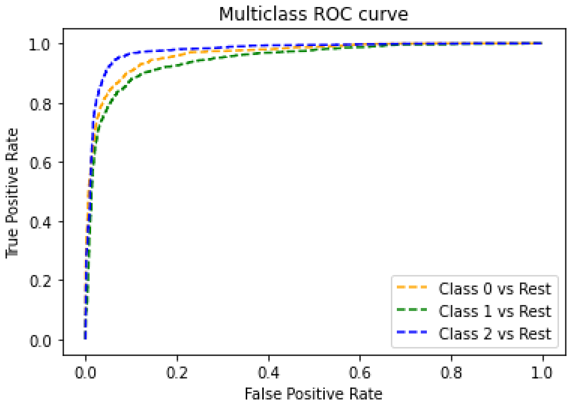


Figure 14: SVM ROC Curve

5.1.6 K Nearest Neighbor(KNN)

The last model we applied to the data is KNN. KNN is a simple, effective, and non-parametric model. It works well on a good distance measure. But it's dependant and biased toward the values of k, the number of data to observe in order to classify each instance. It has a high cost of classifying new instances since all the work is done at classification time, not the training time. [7]

We used sklearn's library KNeighborsClassifier algorithm. The main hyperparameter is n_neighbors, or as we mentioned above k, which is the number of neighbors to use in the queries. As said by [7], a simple way to find the best k is to run the algorithm many times with different k values and choose the one with the best performance. We assigned values of 2 to this variable as it has the best performance on the validation data.

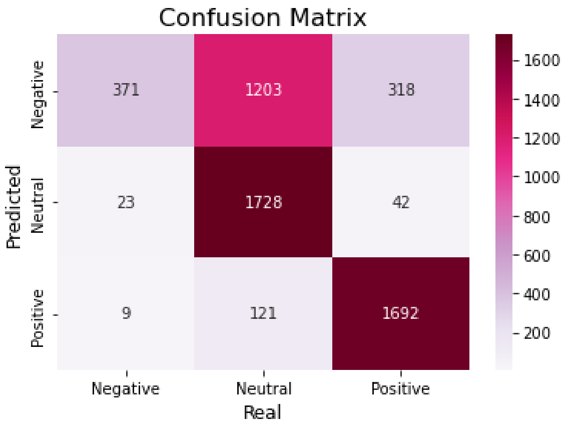


Figure 15: KNN Confusion Matrix

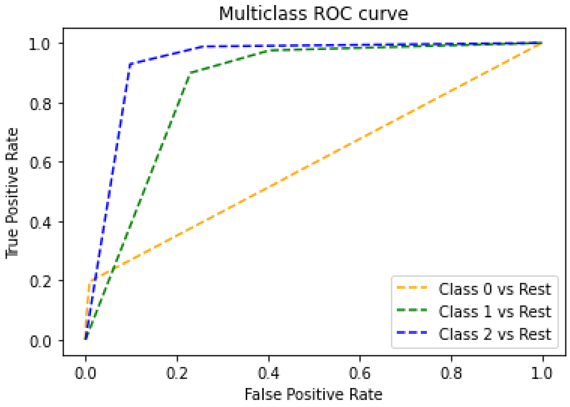


Figure 16: KNN ROC Curve

5.2 Deep learning with neural network

Generally, the neural network idea is to be like the human brain that has some neurons and the goal of a neural network is to learn the model and it is used in the field of machine learning, deep learning, and AI. A neural network wants to mimic the biological human brain, it contains some layers, an input layer, one or more hidden layers, and an output layer. Each node connects to other neurons and has associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network. In the Neural Network, we need to design the text input in order to we can feed the Neural Network. So, we used one hot encoder for this problem. One Hot encoding is a representation of categorical variables as binary vectors. Each integer value is represented as a binary vector that is all zero values except the index of the integer, which is marked with a 1. Our Neural network contains 3 layers, we called this as a baseline model. Then we plot the accuracy and loss of baseline model in figure 17 and 18. As you can see in the loss diagram our validation loss starts to increase after epoch 4 and the training loss continues to decrease, it is normal because our model is training and it means that it could not train well on the new data(validation data). So the reason is that we have over-fitting here in our model, to solve this problem we apply some techniques for over-fitting.

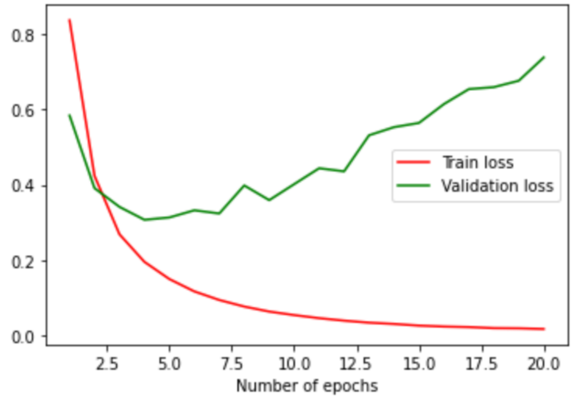


Figure 17: Loss of the baseline model

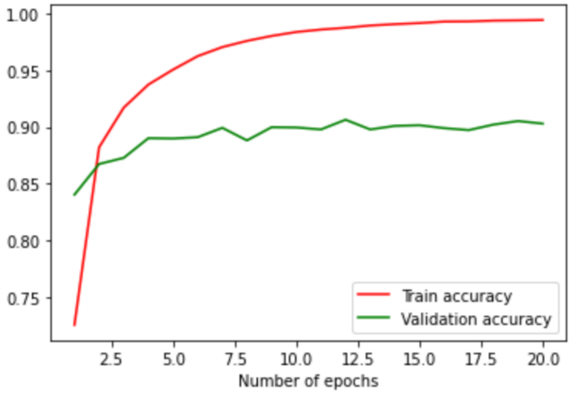


Figure 18: Accuracy of the baseline model

- We reduced the layer of our model and the number of input as well. In figure 21 we compare the loss of baseline model and reduced model for the training set. As you see the reduced model could learn better the new data and the loss of the reduced model starts increasing after epoch 19.

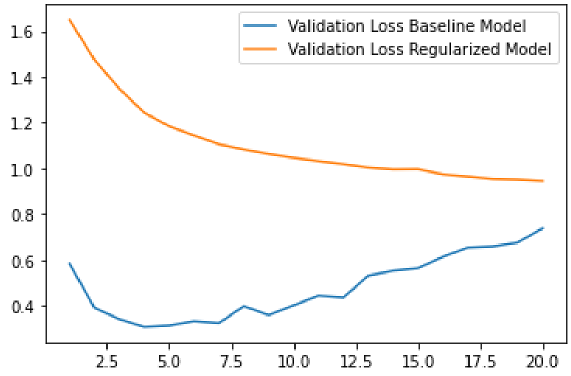


Figure 19: Comparing validation loss of baseline and Losso regularization model

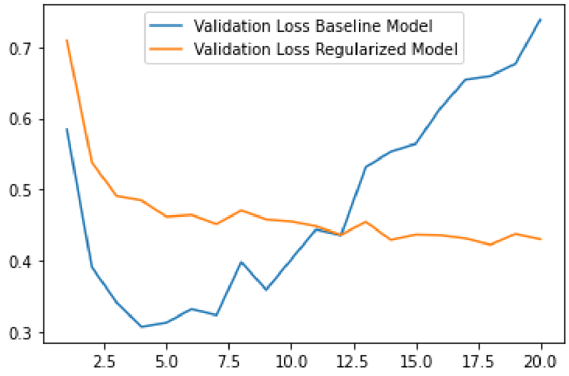


Figure 20: Comparing validation loss of baseline and Ridge regularization model

- We applied two regularization, Losso and Ridge in figure 20 figure 19. Losso validation loss starts increasing after epoch 20 and the Ridge validation loss starts increasing after epoch 20.
- We applied two drop-out layers in our baseline model. By dropping, we mean temporarily removing it from

the network, along with all its incoming and outgoing connections. Figure 22 shows that the validation loss of this model starts increasing after epoch 11 .

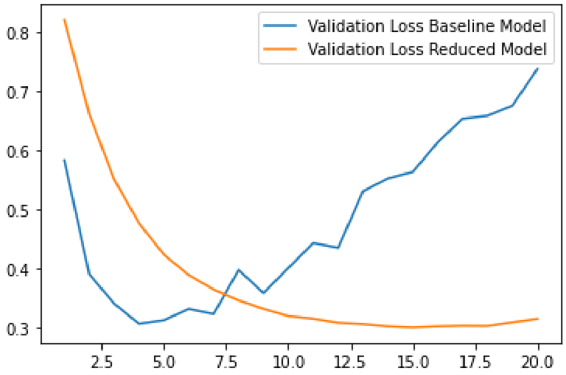


Figure 21: Comparing validation loss of baseline and reduced regularization model

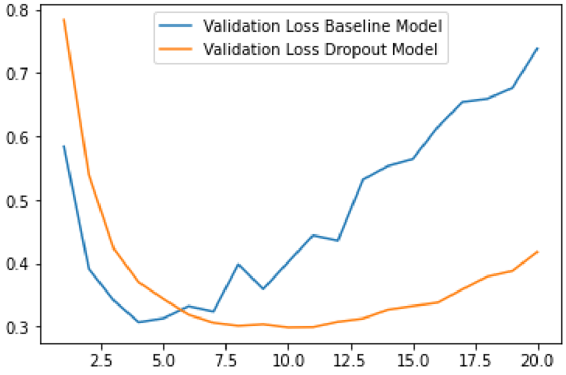


Figure 22: Comparing validation loss of baseline and Drop out regularization model

For testing these models and computing the accuracy we have epoch stop, the epoch stop shows which epoch the model performs best and it shows the result on that epoch that we obtain from analyzing the diagrams in the previous step.

We compare all these methods in figure 23. It seems that reduced model,regularized(ridge) and drop out works better than others in this figure, When we compute their accuracy we see that regularized(ridge) model works best. We provide all of the accuracy in 24

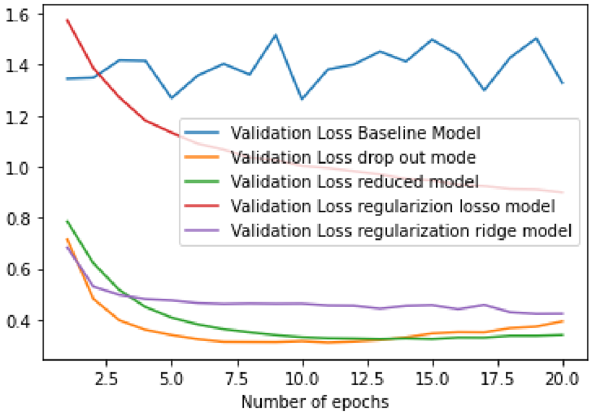
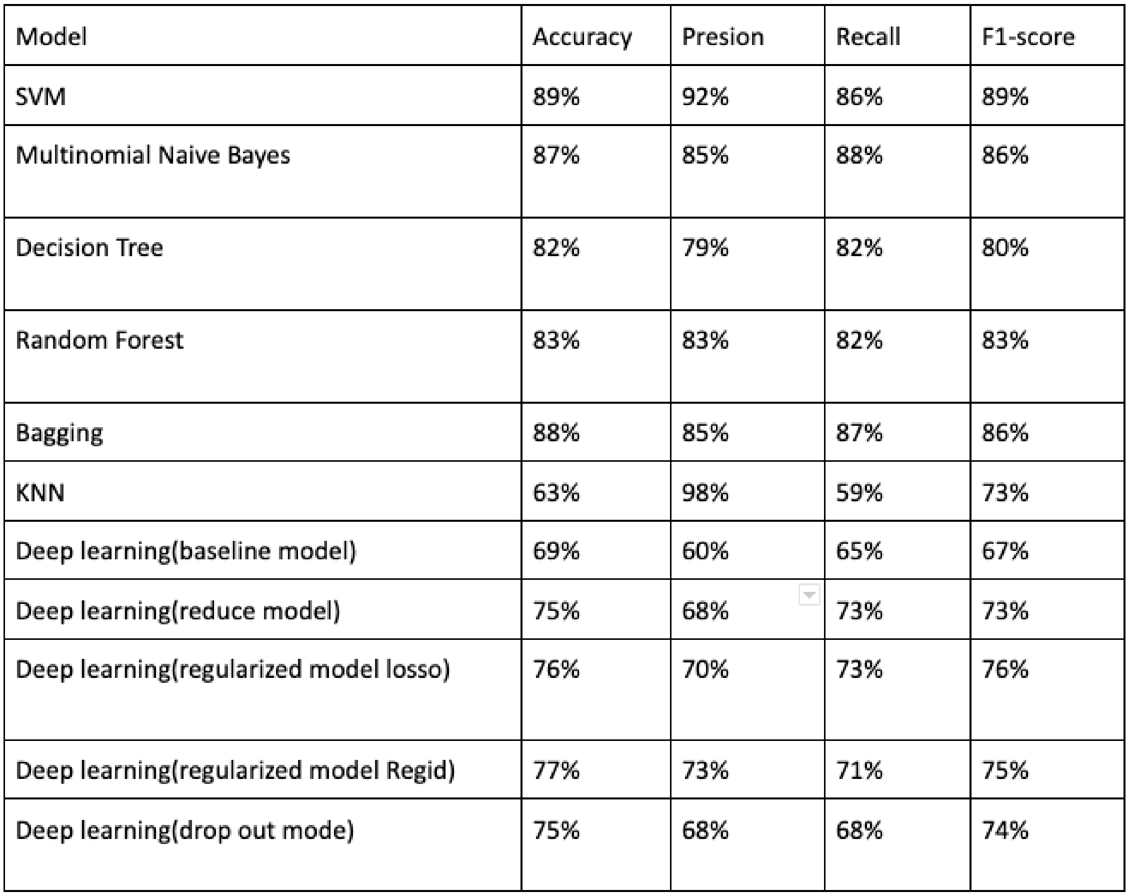


Figure 23: Comparing validation loss of baseline and Drop out regularization model

6 Results

As a result of this project, we compare the accuracy, precision, recall, and f1-score of all of the methods. we got the best result for accuracy for the SVM method and the results for the deep learning techniques were lower than classification techniques.



Model	Accuracy	Presion	Recall	F1-score
SVM	89%	92%	86%	89%
Multinomial Naive Bayes	87%	85%	88%	86%
Decision Tree	82%	79%	82%	80%
Random Forest	83%	83%	82%	83%
Bagging	88%	85%	87%	86%
KNN	63%	98%	59%	73%
Deep learning(baseline model)	69%	60%	65%	67%
Deep learning(reduce model)	75%	68%	73%	73%
Deep learning(regularized model losso)	76%	70%	73%	76%
Deep learning(regularized model Regid)	77%	73%	71%	75%
Deep learning(drop out mode)	75%	68%	68%	74%

Figure 24: Performance Comparison

7 Conclusion:

This paper contributes to the field of data science and sentiment analysis. In this paper, we compared different classification models including decision tree, random forest, multinomial naive bayes, bagging, K-nearest neighbors, and deep neural network. As it is shown in 13, SVM has the best accuracy on test data. We can also look at the 14 where all of the ROC curves for different classes are almost close to a perfect angle and is furthest from the middle line.

Although KNN has the best precision, accuracy-wise it performs the worst out of the models we've implemented. The confusion matrix in 15 shows that KNN can't classify the 'negative' class at all. It classifies most of the 'negative' class as 'neutral'. This is also why we have to consider both precision and recall, or f-1 measure for short. Also, we can see in 16 that the ROC Curve for the 'negative' class is close to a random classification, indicating an unacceptable performance for this specific class.

Other models have an acceptable accuracy on test data and the ROC curves show that as well. The confusion matrices show us that they all collectively perform better for the 'positive' class.

8 Future Work:

We have experimented the Twitter airline sentiment on different methods and one deep learning method. Deep learning is a black box and we cannot exactly figure out the reason for getting the lower result for the deep learning model, so for the next step, we can apply other deep learning techniques including bert to improve its performance of it.

References

[1] Ankita Rane and Anand Kumar. Sentiment classification system of twitter data for us airline service analysis. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, volume 01, pages 769–773, 2018.

[2] Anthony J Myles, Robert N Feudale, Yang Liu, Nathaniel A Woody, and Steven D Brown. An introduction to decision tree modeling. *Journal of Chemometrics: A Journal of the Chemometrics Society*, 18(6):275–285, 2004.

[3] Yashaswini Hegde and SK Padma. Sentiment analysis using random forest ensemble for mobile product reviews in kannada. In *2017 IEEE 7th International Advance Computing Conference (IACC)*, pages 777–782. IEEE, 2017.

[4] Muhammad Abbas, K Ali Memon, A Aleem Jamali, Saleemullah Memon, and Anees Ahmed. Multinomial naive bayes classification model for sentiment analysis. *IJCSNS Int. J. Comput. Sci. Netw. Secur*, 19(3):62–67, 2019.

[5] Peter Bühlmann and Bin Yu. Analyzing bagging. *The annals of Statistics*, 30(4):927–961, 2002.

[6] William S Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.

[7] Gongde Guo, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. Knn model-based approach in classification. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, pages 986–996. Springer, 2003.