**CSIS 3290-001**

**Fundamental of Machine Learning**

**Project 3**

**Yelp consumer experience**

Instructor: Ivan Wong

Group 7

|  |  |
| --- | --- |
| Xuan Truong Ha | 300314475 |
| Le Quang Thy | 300312559 |
| Hao Wei Ho | 300352783 |
| Dong Nie | 300340704 |
| Cheng Yi Chen | 300335751 |

1. **Introduction and Discovery**

This dataset is from the yelp. Yelp is a crowd-sourced local business review and social networking site. Yelp users can submit a review of their products or services using a one to five star rating scale. We would like to predict consumer experience by their comments and star-rating.

We used sentiment analysis to process of determining whether a text is positive or negative. “A sentiment analysis system for text analysis combines natural language processing (NLP) and machine learning techniques to assign weighted sentiment scores to the entities, topics, themes and categories within a sentence or phrase”[1]

In the dataset, we assume that the star-rating greater than 3 menas that a positive experience and vice versa.

This report shows how we processed data, applied different models and ensemble method with more models, moreover,it has the result of prediction in the end.

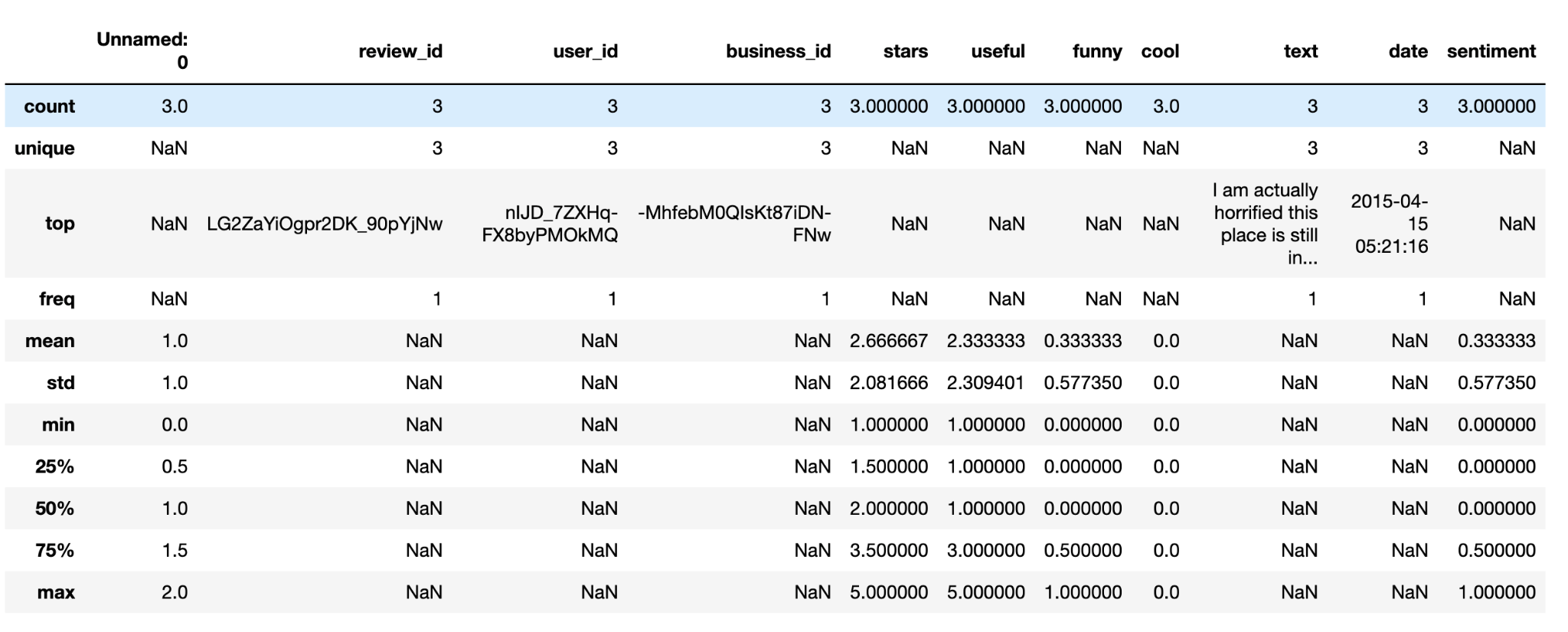
1. **Data Preparation**

We got this dataset from Kaggle contest / yelp-reviews-dataset (<https://www.kaggle.com/code/omkarsabnis/sentiment-analysis-on-the-yelp-reviews-dataset/data>).

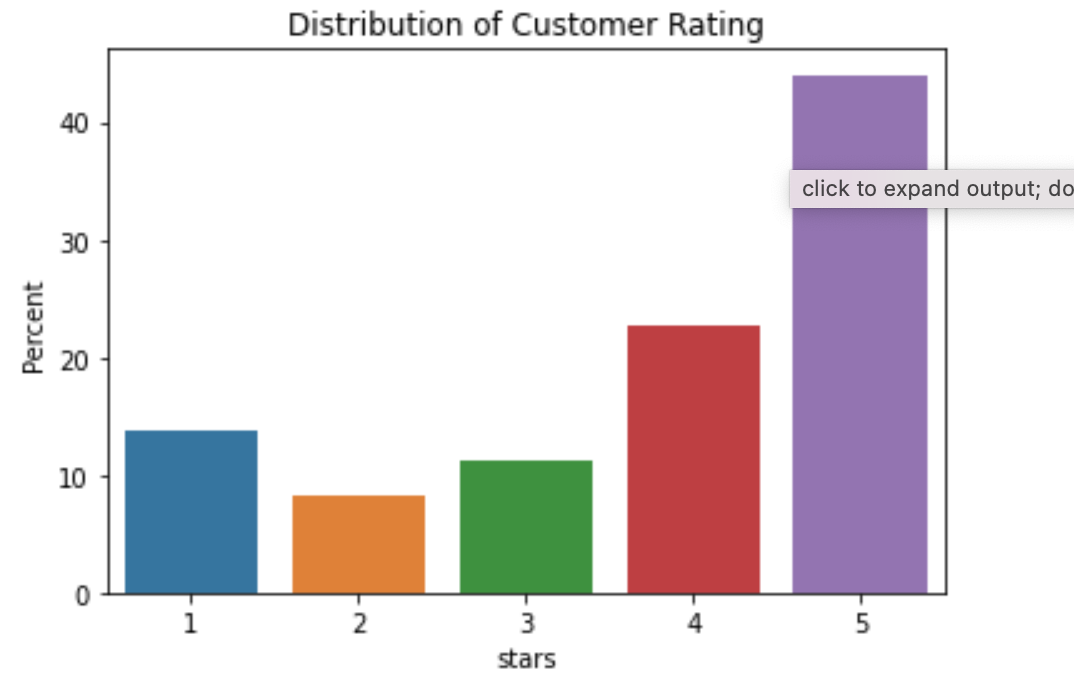
The following are the description of the features we have

* Column 1 - Unique Business ID
* Column 2 - Date of Review
* Column 3 - Review ID
* Column 4 - Stars given by the user
* Column 5 - Review given by the user
* Column 6 - Type of text entered - Review
* Column 7 - Unique User ID
* Column 8 - Cool column: The number of cool votes the review received
* Column 9 - Useful column: The number of useful votes the review received
* Column 10 - Funny Column: The number of funny votes the review received

The following are the summary statistics



The ratio of stars -



1. **Preprocessing**

The goal is to deal with the text data as input and transfer it into numeric representation. To do it, we use the following steps to deal with the text input.

**Step 1** Data preprocess:

To apply the binary classification, we map stars larger than 3 to 1 as positive labels and stars smaller than 3 to 0 as negative labels.

**Step 2** Down sampling:

Since the amount of positive sample and negative sample is imbalance. In this project we sample these two categories and collect 4000 of each as our training data.

**Step 3** Data transformation:

Transform text to tokens using nlp pipeline and the tech we used including tokenize and stemmed.

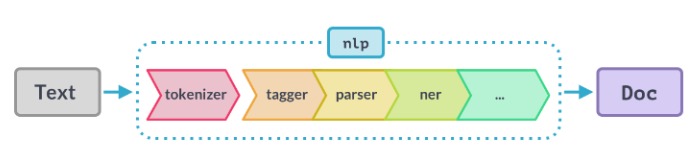


Fig 1. NLP Preprocess to get the token [2]

**Step 4** word representation**:**

We use n-gram with tf-idf to get the representation for our model. To do this, we also applied stop word tech to get rid of the high frequency and not that useful features.

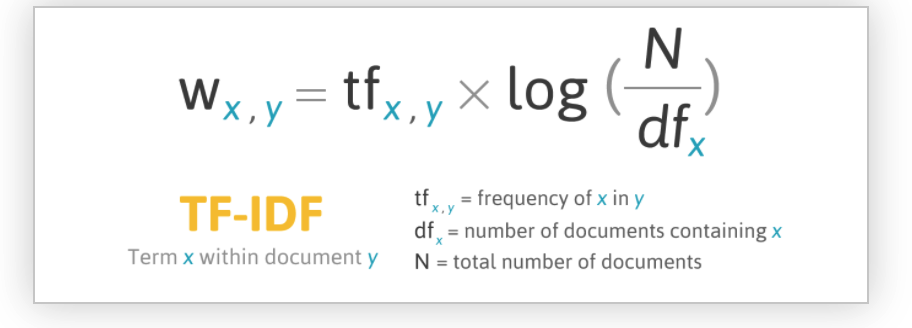


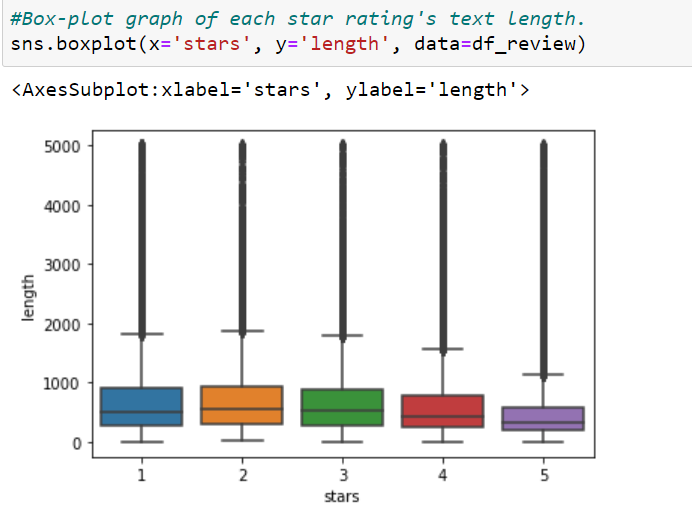
Fig 2. Formula of TF-IDF [3]

**Step 5** - Feature Selection & avoid the effect of sparse data:

We use the top 3000 frequency n-gram as our input to release the effect of sparse data.

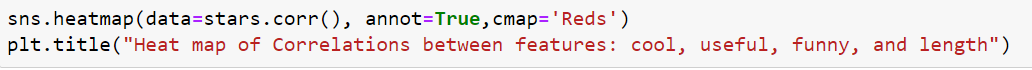
1. **Visualization**

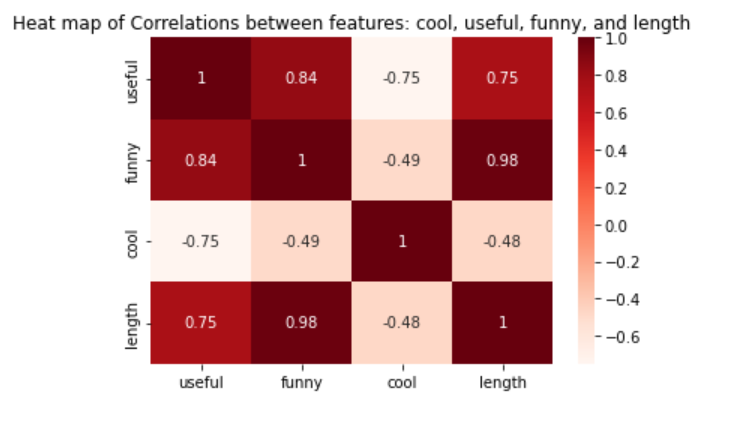
**Box-plot**

****

The 1-star and 2-star and 3-star ratings appear to contain significantly longer content based on the plot, however there are numerous anomalies that can be seen as points above the boxes. Meanwhile, 5-star rating has the shortest length.

**Heat Map**

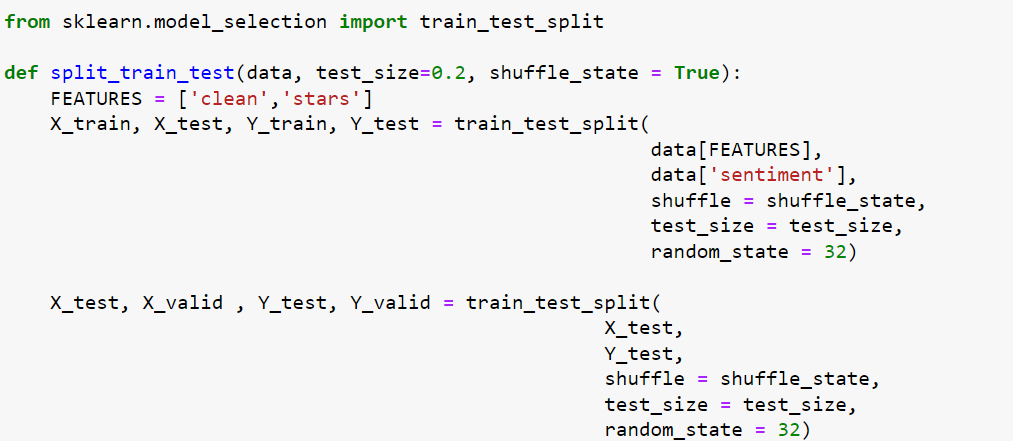


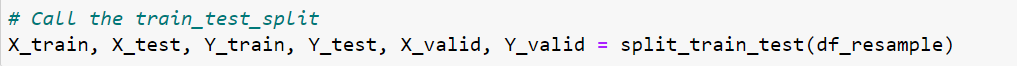


According to the heat map, there is the strongest positive correlation between funny and length, and useful also has strong correlation with funny. There is also a negative correlation relationship between cool and the other three features.

1. **Model Planning and Implementation**

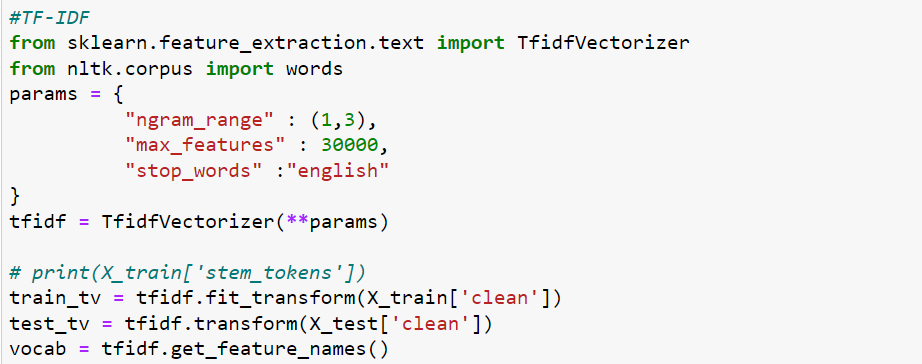
Splitting dataset with ratio 80:20 into training and testing by *train\_test\_split* from Scikit-learn.





**TF-IDF** (Term Frequency Inverse document frequency): In a collection or text, it's the statistical technique. This method determines the significance of a word inside a document. To transform text into numerical representations of vectors, we utilized the TF-IDF algorithm.

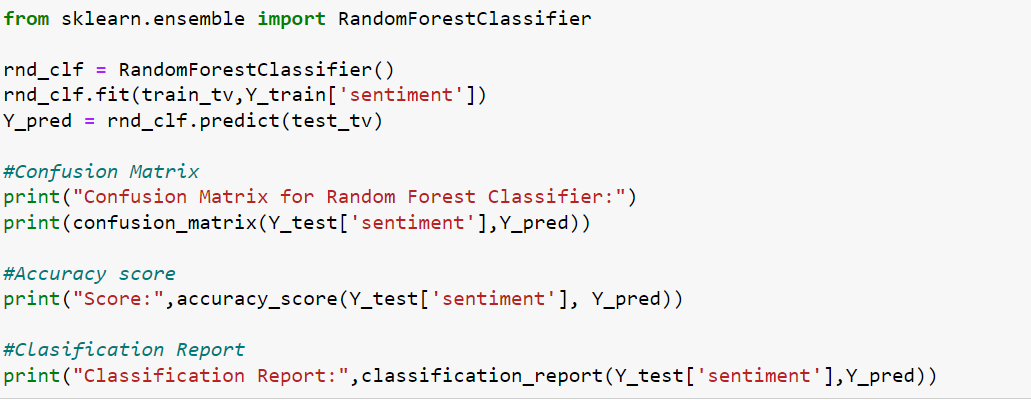
Text Vectorization is conducted after splitting the dataset into training and testing.

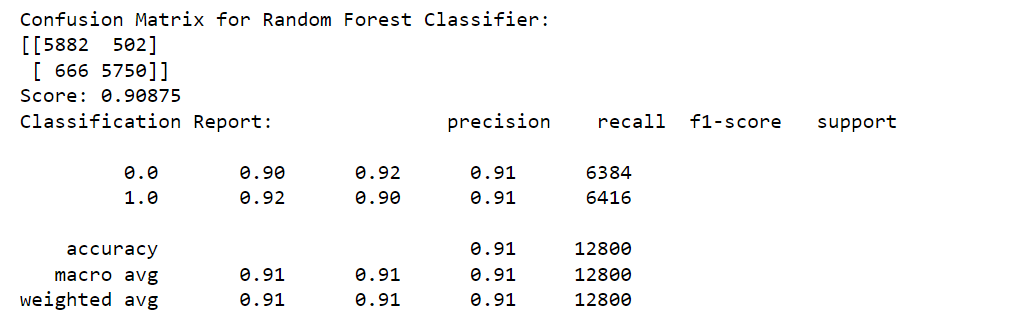


We decided to use Random Forest, ANN with grid search and ensemble modeling to predict an outcome.

**Random forests:** is a learning algorithm that is supervised. It's suitable for both classification and regression.Random forests generate decision trees from randomly chosen data samples, obtain predictions from each tree, then vote on the best option [4].

The first model is random forest:



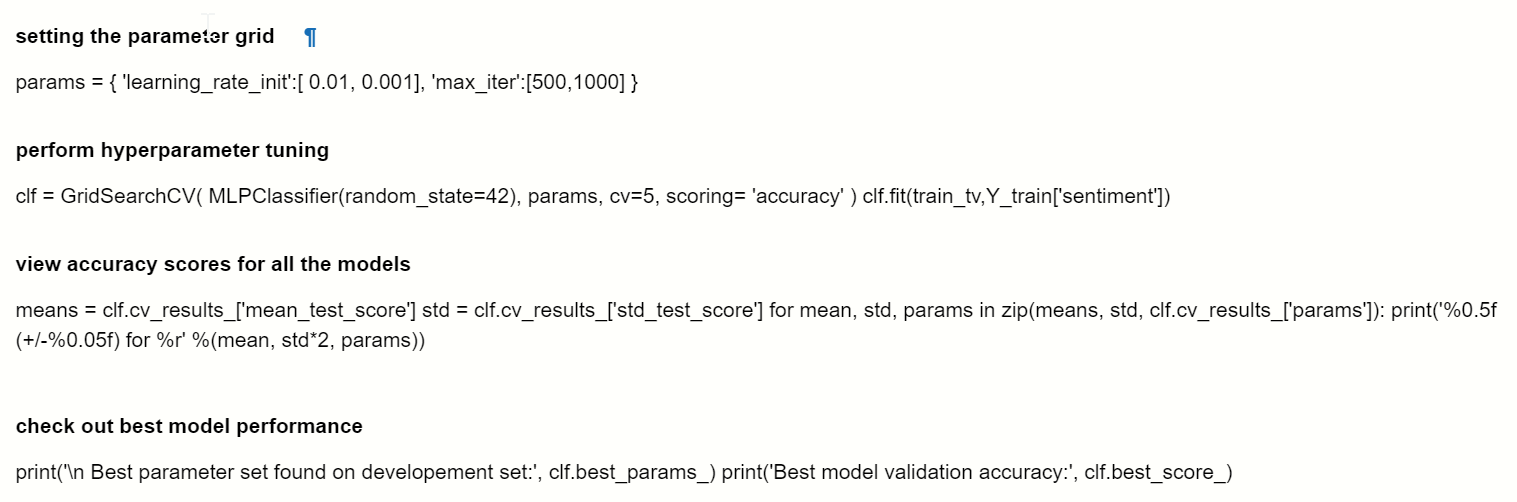


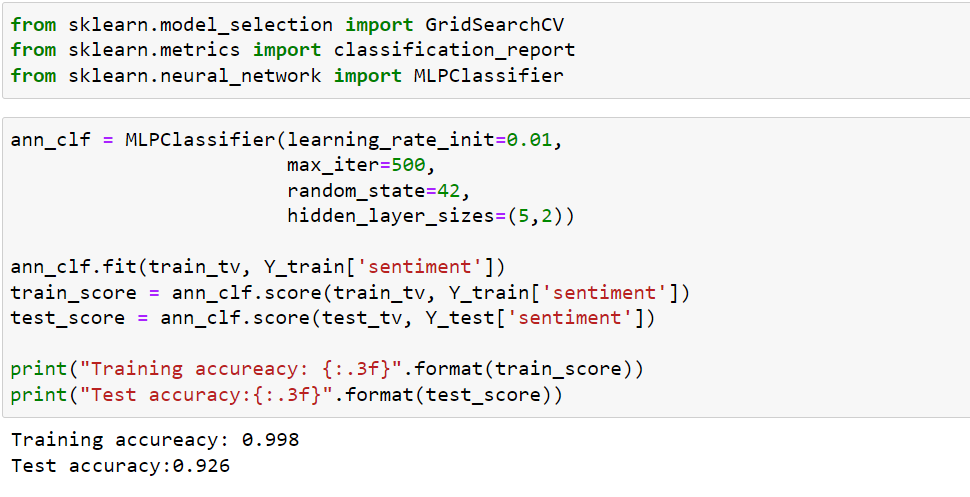
The model accuracy using actual and predicted value achieved 90.8%. However, because we're aware that some bias values exist. Hence, we did second model with ANN.

**ANN with Grid searchCV:**

**ANN** (Artificial Neural Networks): are a type of supervised machine learning in which we have both input and output data in our dataset. Through a learning process, an ANN is tuned for a specific purpose, such as pattern recognition or data categorization. Adjustments to the synaptic connections between neurons are a big part of learning. [5]

Initial idea is using ANN with grid search. However, we found out that our dataset is too huge, it runs at a snail's pace to get the result. Therefore we decided to commend this part in our jupyter notebook. keep it as a demonstration.

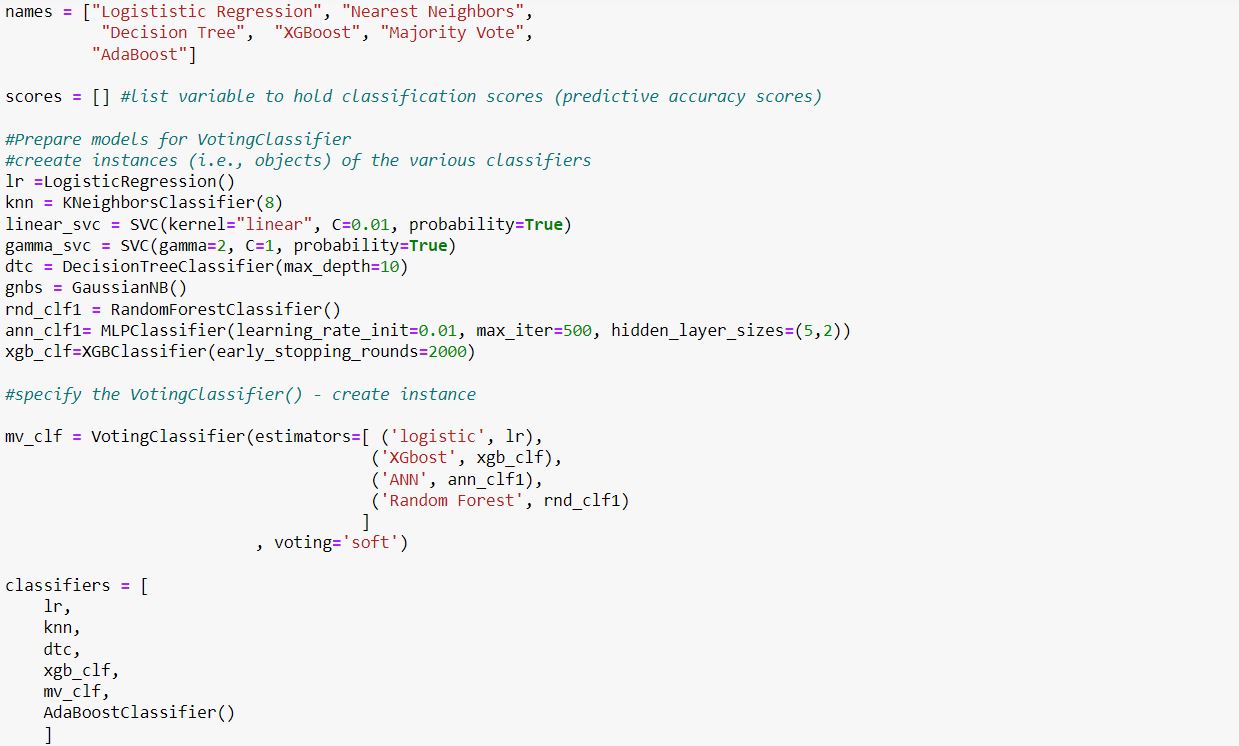




We have tried different settings of **learning\_rate\_init** and **max\_iter** to see training and testing accuracy. Result as below figure.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | learning\_rate\_init=0.001  max\_iter = 1000 | learning\_rate\_init=0.001  max\_iter = 500 | learning\_rate\_init=0.01  max\_iter = 500 | earning\_rate\_init=0.01  max\_iter = 1000 |
| Training | 0.997 | 0.997 | 0.998 | 0.998 |
| Testing | 0.935 | 0.935 | 0.925 | 0.925 |

**Ensemble Method**

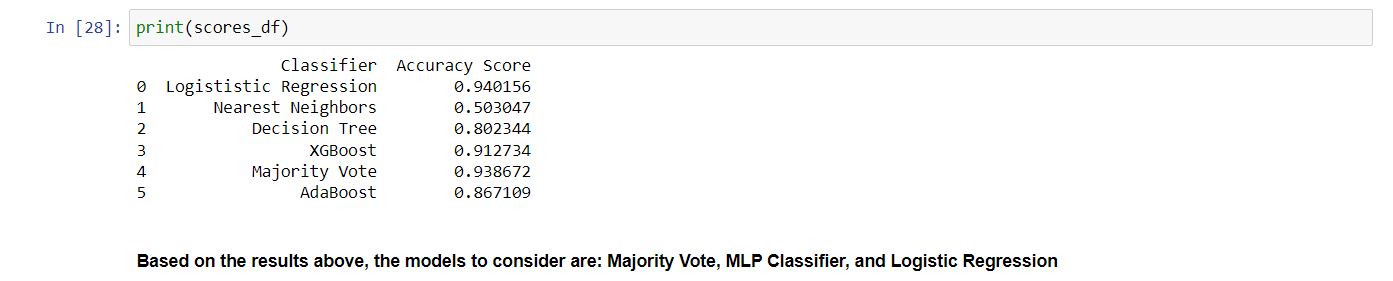


After applying Random Forest and ANN classifier, we do ensemble methods with various kinds of classification namely: Logistic Regression, Nearest Neighbors, Decision Tree, XGBoost, Majority Vote and AdaBoost.

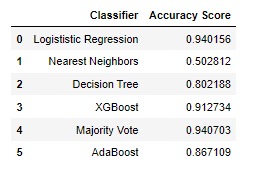
In order to get the most effective accuracy for Ensemble, we include the Random Forest and ANN Classifier into Voting Classifier as well. We tried to run this part a couple times with two different laptops with different random seeds and the results are not too different between models with highest accuracy.

1. **Results Interpretation and Implications**

Result of Laptop A:



Result of Laptop B:



As we can see from above two tables, both of the Majority Vote and the Logistic Regression score are pretty close, both are 0.94. For over all, we have 3 models that have highest score are MLP Classifier: 0.926, Logistic Regression: 0.94 and Majority Vote: 0.94

**Implications:**

* From those models we tested, those chosen models appear valid and accurate on the test data. However, the SVC is slow on large datasets and complicated models are not always ideal.
* Logistic regression is better in this case, so we could use the pipeline to test a great variety of models.
* The parameter values make sense in the context of the domain in some models except XGBoost. “This may not be accurate due to some parameters that are only used in language bindings but passed down to XGBoost core”.
* Voting Classifiers can combine the power of different good models as we do in the Ensemble method.
* Lastly, we will choose the Majority Vote for the prediction part.

1. **Out-of-sample Predictions**

The best models are the voting classifier and logistic regression, since they have pretty close scores. We have 8000 unused data to make predictions. 4000 samples have actual positive responses, and 4000 samples have actual negative responses. Here we use a voting classifier to make predictions.

**Confusion Matrix：**

Validation accuracy: 0.942

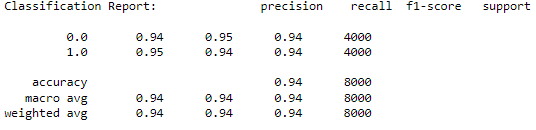
Confusion Matrix for Random Forest Classifier:

[[ 3782 218 ]

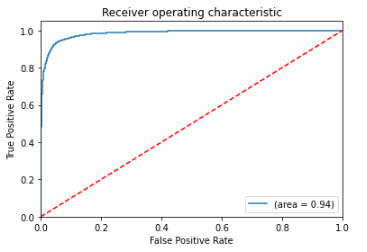
[ 250 3750]]

Validation accuracy is good, it is even better than testing results.

**Classification Report：**



**AUC:**



AUC is 0.94. It suggests that model is relatively accurate.

1. **Concluding Remarks**

We find a good model after testing a great variety of models with different hyperparameters. The final model meets our expectations, and we can perform relatively accurate predictions based on the user's comments.

1. **Member Contribution**
2. Xuan Truong Ha (20%) : Ensemble Model and Results Interpretation
3. Le Quang Thy Nguyen (20%) :Data visualization and Model of Random Forest
4. Hao Wei Ho (20%) : Data wrangling and feature selection
5. Dong Nie (20%) : Partially join modelling, predicting, combine notebook and combine video recording.
6. Cheng Yi Chen (20%) : Model of ANN with GridSearchCV and report final edit.
7. **References**

[1]LEXALUTICS - Sentiment Analysis Explained

<https://www.lexalytics.com/technology/sentiment-analysis>

[2] Spacy. Linguistic Features. Retrieved from <https://spacy.io/usage/linguistic-features/>

[3] FiloTechnologia. A simple java class for tf\*idf scoring. Retrieved from <http://filotechnologia.blogspot.com/2014/01/a-simple-java-class-for-tfidf-scoring.html>

[4] Avinash N. (2018). Understanding Random Forests Classifiers in PythonTutorial. Retrieved from <https://www.datacamp.com/community/tutorials/random-forests-classifier-python>

[5] GeeksforGeeks (2022). Implementing Artificial Neural Network training process in Python. Retrieved from <https://www.geeksforgeeks.org/implementing-ann-training-process-in-python/>

1. **Presentation Video link**

<https://collegedouglas-my.sharepoint.com/personal/nied_student_douglascollege_ca/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Fnied%5Fstudent%5Fdouglascollege%5Fca%2FDocuments%2F3290%5FPresentation%2FYelp%5Fgroup7%2Emp4&parent=%2Fpersonal%2Fnied%5Fstudent%5Fdouglascollege%5Fca%2FDocuments%2F3290%5FPresentation&ga=1>