Final Project Report [Group: 6]
CSE343: Machine Learning, Winter 2022

Mohd Naki
(2018052)
naki18052@iiitd.ac.in

Vaibhav Gupta
(2019341)
vaibhav19341@iiitd.ac.in

1. Introduction

This section will contain the problem statement and the motivation.

1.1. Problem Statement

The objective of our project is to classify the positive and negative reviews of consumers over different types of products and build a supervised learning model to polarize huge amounts of reviews.

1.2. Motivation

The motivation behind implementing this technique is that organisations always want to find consumer opinions and emotions about their services and products. Prospects also want to know the opinions and emotions of existing consumers before they purchase the product or service.

2. Related Work

We referenced a research paper from Stanford which attempts to do sentiment analysis on the same dataset: https://cs229.stanford.edu/proj2018/report/122.pdf

The best performing model from the above paper is a LSTM (Long Short Term Memory) model which achieved 73.5% training accuracy and 71.5% testing accuracy.

3. Dataset and Evaluation

The dataset is a list of 34,660 consumer reviews for Amazon products like the Kindle, Fire TV Stick, and more provided by 'Datafiniti's Product Database'. The dataset includes basic product information, rating, review text, and more for each product. This dataset is a sample from a larger dataset.

After removing samples with unwanted and empty values, 34,627 samples were left. These were further split as:

- Train set: 64% = 22161 samples
- Validation set: 16% = 5540 samples
- Test set: 20% = 6926 samples

Dataset url: https://www.kaggle.com/datasets/datafiniti/consumer-reviews-of-amazon-products?select=1429_1.csv

3.1. Features

The dataset has 21 features (*Figure 1*) . Some of which are ID, Name, brand, category, manufacturer, review date, review id, review rating, review text, user city, user province, user name.
Out of these we have extracted review rating (as target) and review text (as independent feature) to train the models.

3.2. Pre-processing

Sentiments are classified as follows:

$$Positive : rating > 3$$

$$Neutral : rating = 3$$

$$Negative : rating < 3$$

We have used the 'Bag of Words' strategy to turn the data into numerical features. We pre-processed the text using:

- **Tokenization:** Breaking the review text into words.
- **Stop-words Filtration:** Words like "the", "are" are filtered.
- We used occurrence counting, which built a dictionary of features from integer indices with word occurrences.
- We converted this dictionary of texts into a feature vector.

The above was achieved by CountVectorizer() from scikit-learn. We have 27701 training samples with 12487 distinct words. In longer text documents, we see a higher average count value of words that are insignificant, this particular problem will overweigh the text documents that have lower average counts with same frequency. To overcome this redundancy, we used Tfidf-Transformer() from scikit-learn.

3.3. Some observations about the dataset

From *Figure 2* and *Figure 3* it can be infered that the dataset is unbalanced.
Figure 4 shows the separability of the dataset by plotting the frequent and infrequent words occurring in positive and negative reviews.

For example: 'love', 'highly recommended', 'perfect', 'excellent' occur in positive reviews. And 'defective', 'returning', 'poor', 'disappointing' occur in negative reviews.

3.4. Evaluation metrics

- Accuracy
- Confusion matrix
- F1 score
- Weighted F1 score
- Precision score
- Weighted Precision score
- Recall score
- Weighted Recall score
- Log loss
- ROC-AUC plots
- Learning Curve
- Validation Curve

4. Methodology

This section will describe the different methodologies adopted and what hyperparameters were chosen.

4.1. Models trained

- Multinomial Naive Bayes
- Logistic Regression
- Linear Support Vector Machine
- K-Nearest Neighbour Classifier
- Neural Network

4.2. Multinomial Naive Bayes

The model was trained without fitting class priors (*Figure 5*) and smoothing parameter was set as 1.3 (*Figure 6*).

From *Figure 7* and *Figure 8* we can observe that the model is trained correctly and it does not overfit or underfit the dataset.

4.3. Logistic Regression

For training the model, L1 regularization was chosen (*Figure 10*) and the regularization parameter was set as 5 (*Figure 11*) to overcome overfitting.

From *Figure 12* and *Figure 13* we can observe that the model is trained correctly and it does not overfit or underfit the dataset.

4.4. Linear Support Vector Machine

There was no observable difference in model performance on varying the regularization parameter (*Figure 15*). It was kept as 1.

From *Figure 16* we can observe that there is a large gap between the error of the train and validation set which means that the model has overfit the dataset.

4.5. K-Nearest Neighbour Classifier

Best performance was found at K = 2000 (*Figure 19*).

From *Figure 20* and *Figure 21* we can observe that the model is trained correctly and it does not overfit or underfit the dataset.

4.6. Neural Network

The network has 22,161 neurons in the input layer (Number of samples in the train set), 150 in the hidden layer (*Figure 25*) and 3 in the output layer (Number of classes).
The 'adam' solver (*Figure 23*) was chosen to perform the parameter update. It has a learning rate = 0.001 and l2 regularization = 0.0001.

Two neural network models were trained. One with the 'ReLU' activation function and the other with 'Sigmoid'.

Figure 26 and *Figure 27* show the learning curve of models with ReLU and Sigmoid activation respectively. We observe that both models are trained correctly and have similar performance.

5. Results & Analysis

This section contains the results and the analysis.

5.1. Results

Table 1 shows the results obtained from the test sets of the respective models.

Since, we are working with an imbalanced dataset, instead of 'Accuracy', we will use 'F1 score' and 'Log loss' for determining the best performing model.

The ROC plots for respective models is shown by figures: 9, 14, 18, 22, 28, and 29.

5.2. Analysis

Based on the results shown in *Table 1*, we can observe that the Neural Network with Sigmoid activation performs best since it has the highest F1 score and lowest error.

F1 score varies between 0 and 1, and all of our models have an F1 score of less than 0.5. F1 score is highly dependent upon the intensity of the imbalance in the dataset as it gives equal weightage to each class.

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

$$avg.F1 = Average(Cl1_F1 + Cl2_F1 + Cl3_F1)$$

To overcome this flaw we calculate weighted F1 score.

$$avg.WeightedF1 = \frac{N1 * Cl1_F1 + N2 * Cl2_F1 + N3 * Cl3_F1}{N1 + N2 + N3}$$

Metrics	Multinomial Naive Bayes	Logistic Regression	Linear SVM	K-Nearest Neighbour	Neural Network (ReLU)	Neural Network (Sigmoid)
Accuracy	0.923	0.929	0.930	0.928	0.930	***0.931***
max. F1 score	0.960	0.963	0.964	0.963	0.965	0.965
avg. F1 score	0.363	0.342	0.390	0.321	0.447	***0.448***
Weighted avg. F1 score	0.90	0.90	0.90	0.90	***0.91***	***0.91***
max. Precision score	0.932	0.931	0.932	0.928	0.938	0.938
avg. Precision score	0.484	0.588	0.637	0.309	0.630	0.651
Weighted avg. Precision score	0.82	0.89	0.90	0.86	0.90	0.90
max. Recall score	0.990	0.998	0.999	1.000	0.994	0.995
avg. Recall score	0.354	0.344	0.372	0.333	0.410	0.409
Weighted avg. Recall score	0.92	0.93	0.93	0.93	0.93	0.93
Log loss	2.594	2.418	2.353	2.423	2.368	***2.338***

Table 1. Metrics obtained from test set of respective models. (*Note: values may vary by 3-5%*)

Cl1, Cl2, Cl3 denote the 3 classes. N1, N2, N3 are the number of samples in each class

5.3. Conclusion

Based on the above we can conclude that all of our models perform very well on the given dataset. The Neural Networks marginally outperform other models. Among the Neural Networks, the one with Sigmoid activation has lower error than the one with ReLU activation.

5.4. Comparison

The state of the art model mentioned in Section 2 has an **accuracy of 71.5% on the test set**.

Our Neural Network with Sigmoid activation has an **F1 score of 0.91 on the test set**.

It is difficult to compare the 2 models exactly because the model in Section 2 is trained on data where there are 5 classes instead of 3. Also it is mentioned the classes with lower number of samples were re-sampled 15 times to have less imbalance in the dataset. There are no other metrics provided in the paper referred to in Section 2 which makes comparison a little more difficult.

5.5. Possible future improvements

The dataset used in this project is a subset with 34,660 samples of a larger dataset which has over 334,000 samples. The larger dataset was not used as it was not openly accessible. There are also other datasets with millions of samples available. Having more data to play with will give more insight into the results of different models.

As for learning techniques, we can explore other types of Support Vector Machines (*Figure 30*), We can increase the complexity of the Neural networks by adding more layers and explore different types of Neural Networks like CNNs, RNNs. We can also explore NLP specific models like BERT, GPT2, GPT3 etc.

6. Contribution

This section contains the work done by each member.

6.1. Deliverables

Deliverables promised in the project proposal:

Mohd Naki:

- Pre-processing ✓
- Implementing LSVM ✓
- Implementing LSTM ×
- Implementing Neural Network ✓
- Confusion Matrix ✓
- Area under ROC curve ✓

Vaibhav Gupta:

- Pre-Processing ✓
- Implementing Naive Bayes ✓
- Implementing KNN ✓
- Confusion Matrix ✓
- Log Loss ✓
- Area under ROC curve ✓

LSTM was was removed because a group member dropped the course.

6.2. References & Citations

- https://www.kaggle.com/datafiniti/consumer-reviews-of-amazon-products
- https://datascience.stackexchange.com/questions/62303/difference-between-learning-curve-and-validation-curve#:~:text=A%20learning%20curve%20plots%20the,is%20too%20simple%20(biased)
- https://www.dataquest.io/blog/learning-curves-machine-learning/
- https://www.analyticsvidhya.com/blog/2019/08/detailed-guide-7-loss-functions-machine-learning-python-code/
- https://github.com/dataiku-research/mealy/blob/main/examples/plot_mealy.py
- https://scikit-learn.org/stable/auto_examples/model_selection/plot_validation_curve.html#sphx-glr-auto-examples-model-selection-plot-validation-curve-py
- https://scikit-learn.org/stable/auto_examples/model_selection/plot_learning_curve.html#sphx-glr-auto-examples-model-selection-plot-learning-curve-py
- https://mickzhang.com/amazon-reviews-using-sentiment-analysis
- https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/#:~:text=A%20learning%20curve%20is%20a,from%20a%20training%20dataset%20incrementally
- https://machinelearningmastery.com/the-model-performance-mismatch-problem/
- https://colab.research.google.com/github/DerwenAI/spaCy_tuTorial/blob/master/spaCy_tuTorial.ipynb#scrollTo=fw69vnvUELaQ
- https://scikit-learn.org/

6.3. Individual Contributions

All the code written resides in one jupyter notebook divided into multiple sections.

Member 1 **Mohd Naki:**
Did EDA and pre-processing. Implemented Logistic Regression, Linear SVM, Neural Network with ReLU activation.

Section wise contribution:

- A Quick look at the Data
- Exploring the Data
- Analysing the sentiment : assigning the sentiments and splitting feature and label dataframes
- Pre-processing and feature extraction
- Logistic regression
- Linear Support Vector Machine
- Neural Network : NN with ReLU

Member 2
Vaibhav Gupta:
Did EDA and pre-processing. Implemented Multinomial Naive Bayes, KNN classifier, Neural Network with Sigmoid activation.

Section wise contribution:

- Splitting the Data
- Correlation
- Analysing the sentiment : plotting the sentiment wise distribution of dataset & making scatterplot to learn about separability
- Pre-processing and feature extraction
- Multinomial Naive Bayes
- K-Nearest Neighbour Classifier
- Neural Network : NN with Sigmoid

7. Figures, Plots & Tables

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34660 entries, 0 to 34659
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    34660 non-null  object
 1   name                  27900 non-null  object
 2   asins                 34658 non-null  object
 3   brand                 34660 non-null  object
 4   categories            34660 non-null  object
 5   keys                  34660 non-null  object
 6   manufacturer          34660 non-null  object
 7   reviews.date          34621 non-null  object
 8   reviews.dateAdded     24039 non-null  object
 9   reviews.dateSeen      34660 non-null  object
 10  reviews.didPurchase   1 non-null      object
 11  reviews.doRecommend   34066 non-null  object
 12  reviews.id            1 non-null      float64
 13  reviews.numHelpful    34131 non-null  float64
 14  reviews.rating        34627 non-null  float64
 15  reviews.sourceURLs    34660 non-null  object
 16  reviews.text          34659 non-null  object
 17  reviews.title         34655 non-null  object
 18  reviews.userCity      0 non-null      float64
 19  reviews.userProvince  0 non-null      float64
 20  reviews.username      34658 non-null  object
dtypes: float64(5), object(16)
memory usage: 5.6+ MB
```

Figure 1. Columns present in the dataset

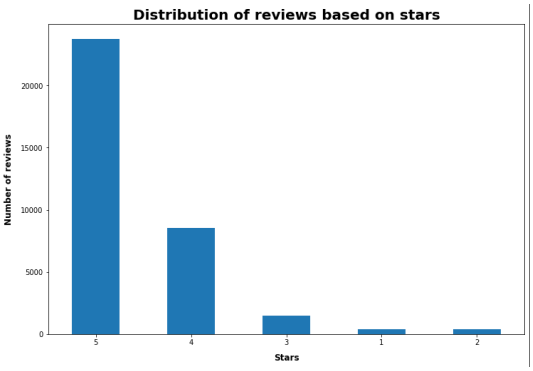

Figure 2. Number of reviews in each rating

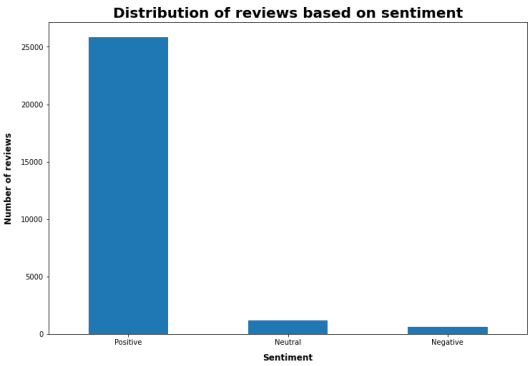

Figure 3. Number of reviews in each sentiment

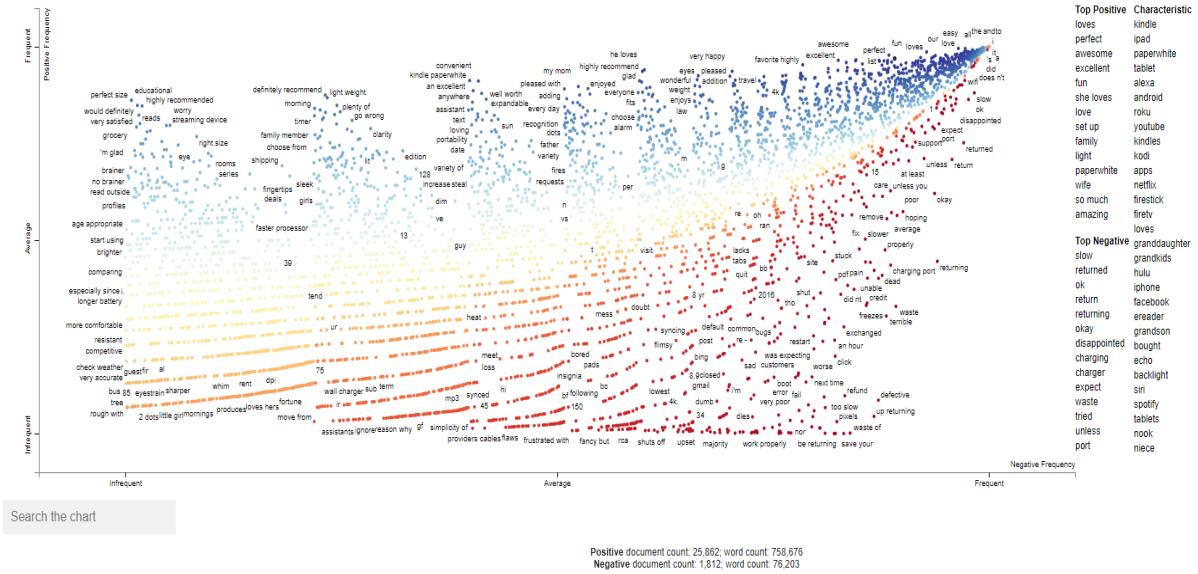

Figure 4. Separability of dataset

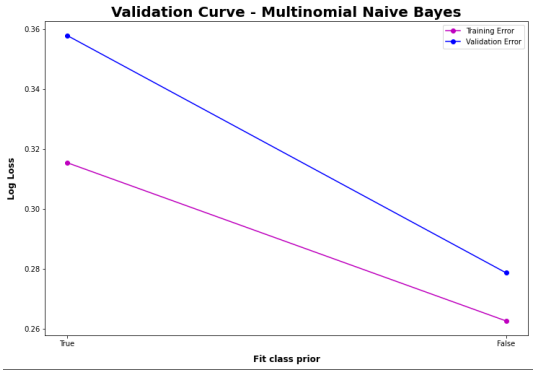

Figure 5. Validation plot to evaluate if fitting class prior yields lower error

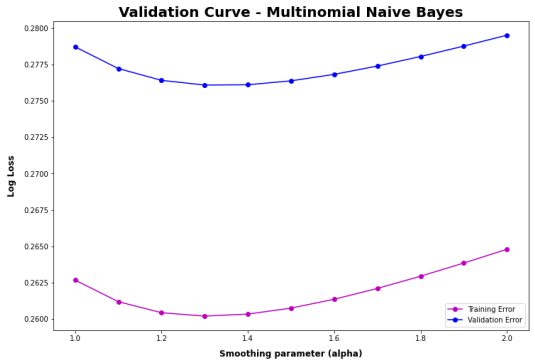

Figure 6. Validation plot to evaluate which value of smoothing parameter yields lower error

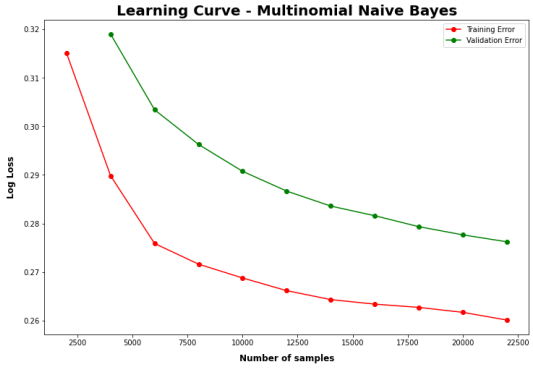

Figure 7. Learning curve plotting Log Loss as a function of training samples

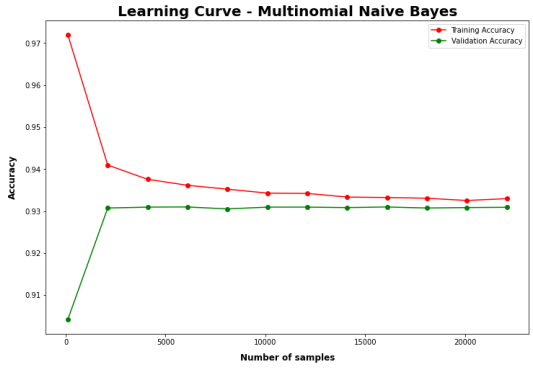

Figure 8. Learning curve plotting Accuracy as a function of training samples

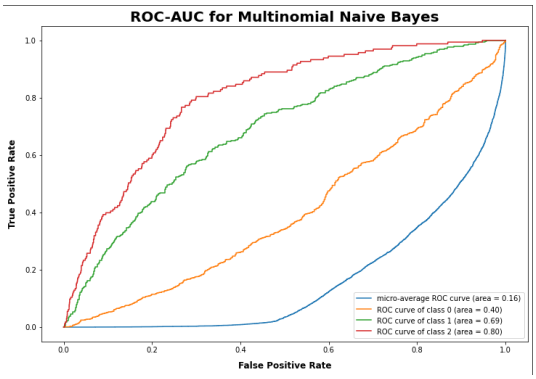

Figure 9. ROC curve for Multinomial Naive Bayes model

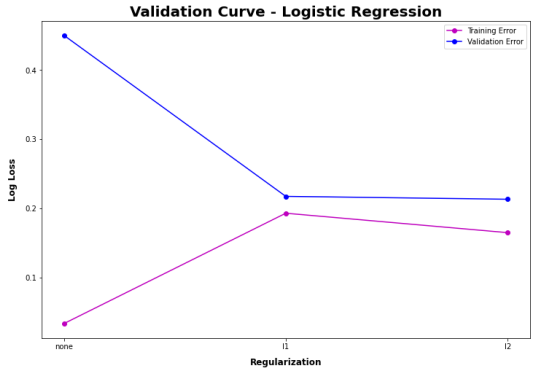

Figure 10. Validation plot to evaluate which type of regularization yields lower error

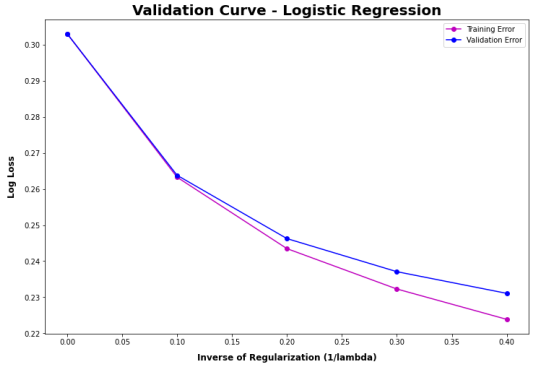

Figure 11. Validation plot to evaluate what value regularization yields lower error

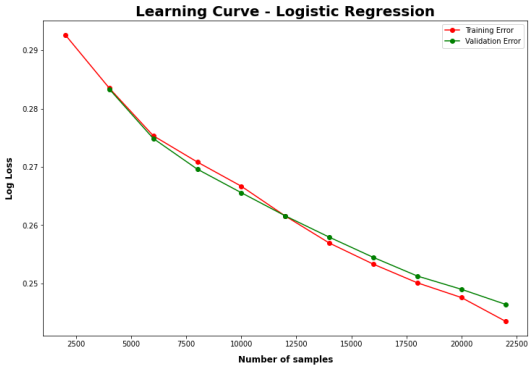

Figure 12. Learning curve plotting Log Loss as a function of training samples

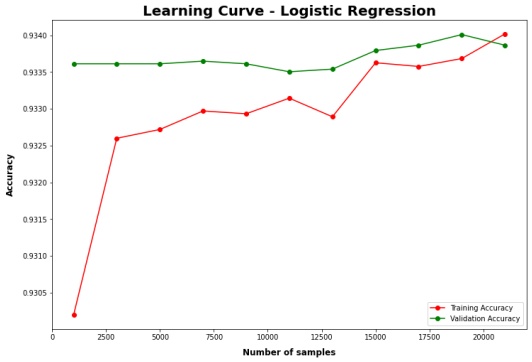

Figure 13. Learning curve plotting Accuracy as a function of training samples

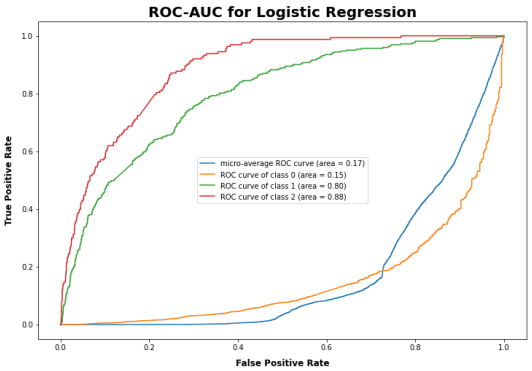

Figure 14. ROC curve for Logistic Regression model

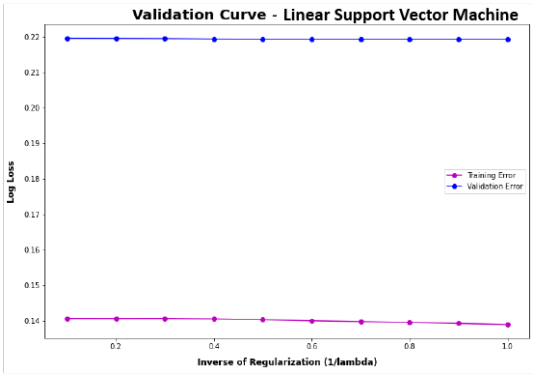

Figure 15. Validation plot to evaluate what value regularization yields lower error

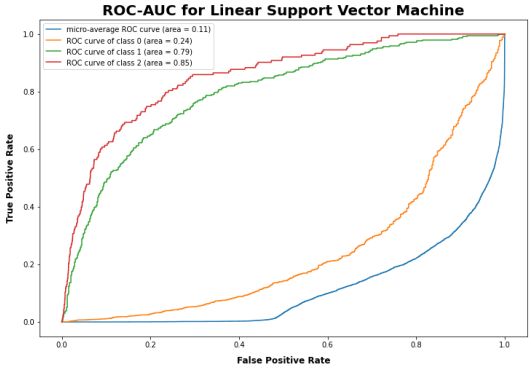

Figure 18. ROC curve for Linear Support Vector Machine model

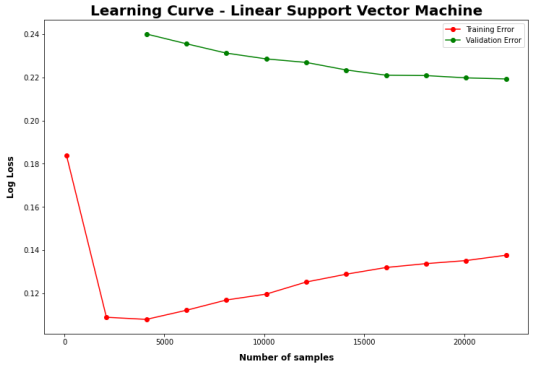

Figure 16. Learning curve plotting Log Loss as a function of training samples

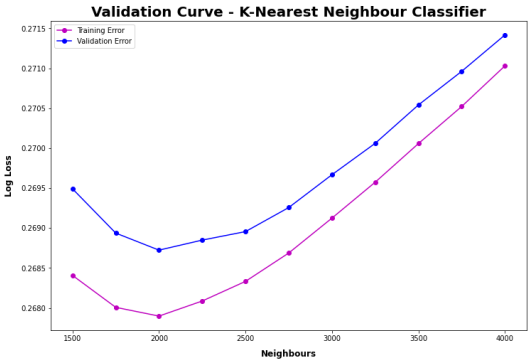

Figure 19. Validation plot to evaluate what number of neighbours yields best performance

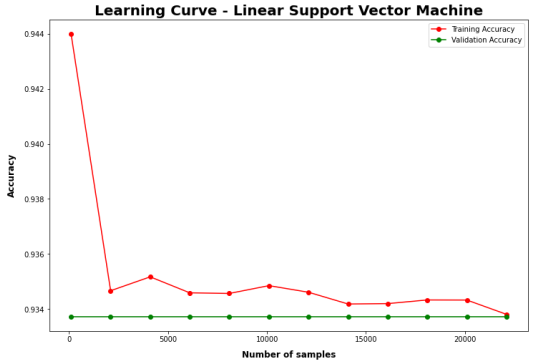

Figure 17. Learning curve plotting Accuracy as a function of training samples

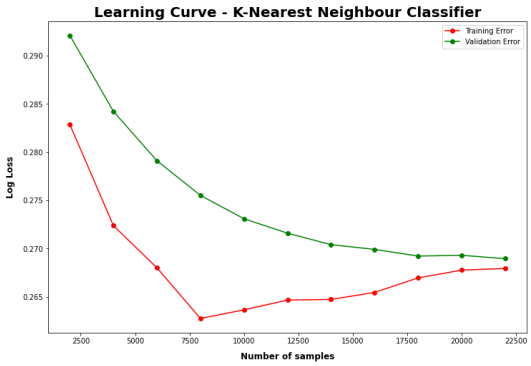

Figure 20. Learning curve plotting Log Loss as a function of training samples

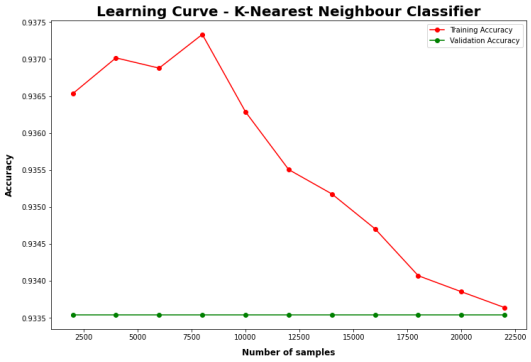

Figure 21. Learning curve plotting Accuracy as a function of training samples

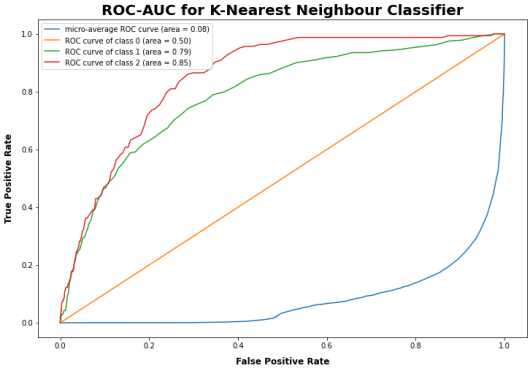

Figure 22. ROC Curve for KNN model

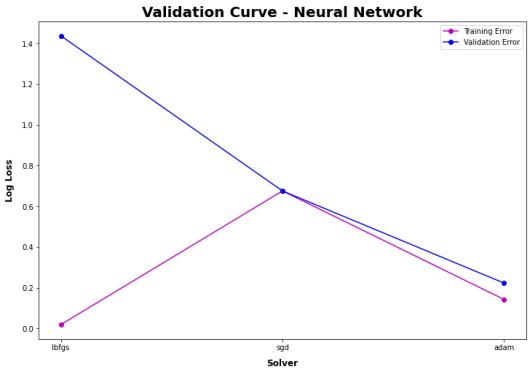

Figure 23. Validation plot to determine which solver to use

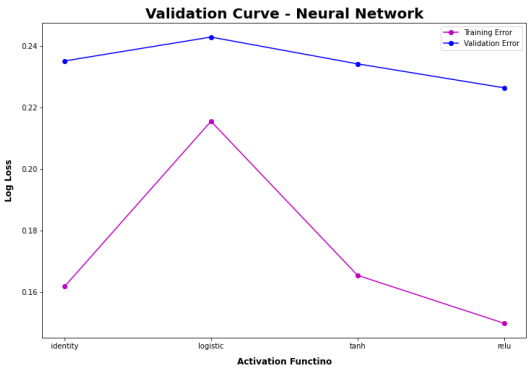

Figure 24. Validation plot to determine which activation function to use. identity = linear, logistic = sigmoid

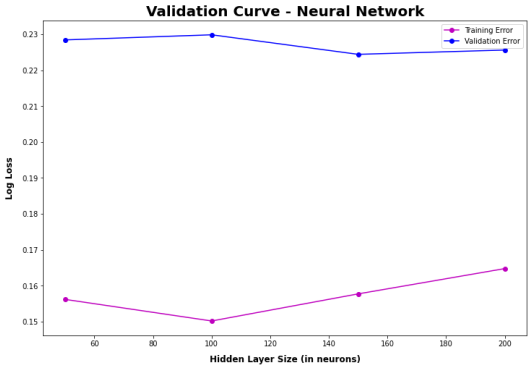

Figure 25. Validation plot to determine the number of neurons in the hidden layer

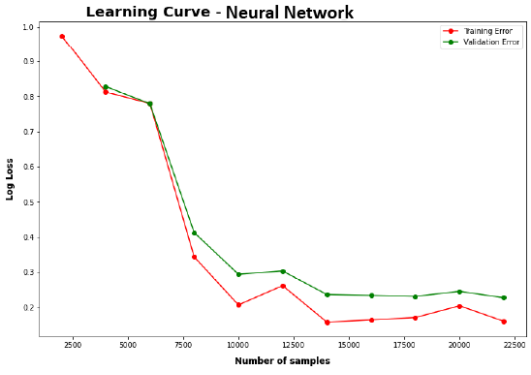

Figure 26. Learning curve plotting Log Loss as a function of training samples for ReLU activation

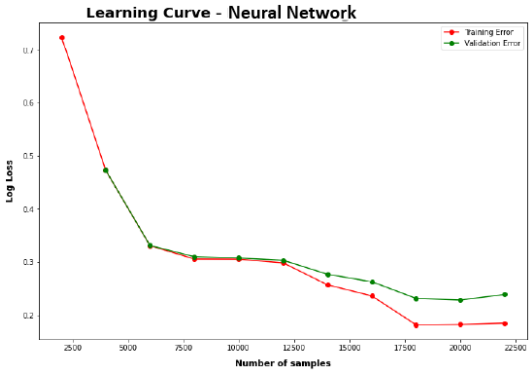

Figure 27. Learning curve plotting Log Loss as a function of training samples for Sigmoid activation

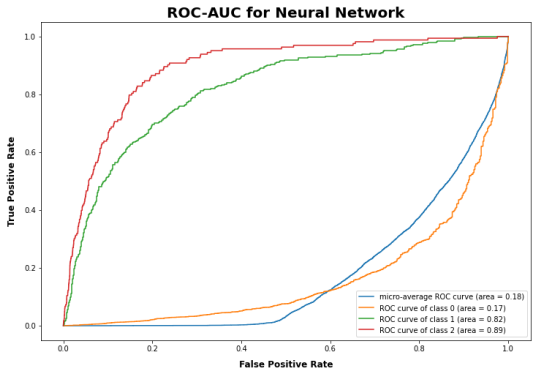

Figure 28. ROC curve for Neural Network with ReLU activation

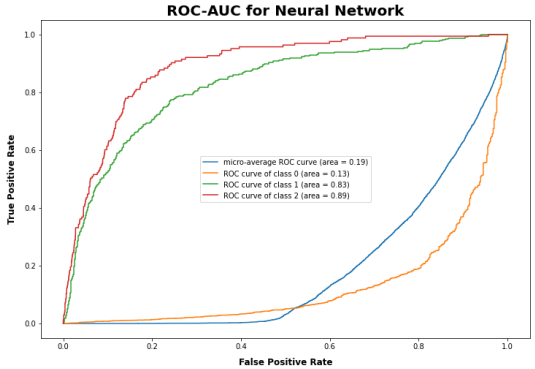

Figure 29. ROC curve for Neural Network with Sigmoid activation

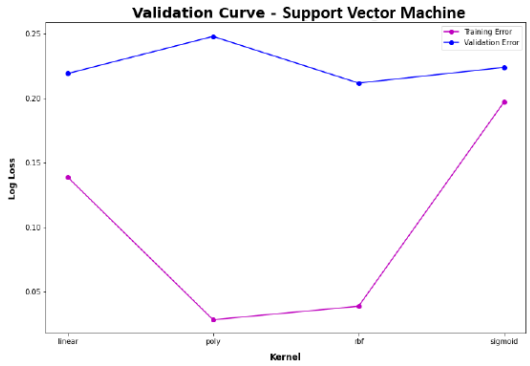

Figure 30. Validation plot to determine performance of different kernels for SVM