Abstract

In this project, the performances of the Logistic Regression and Multiclass Regression linear classification machine learning models were investigated on two datasets, namely the IMDB Reviews dataset and the 20 Newsgroups dataset. The prediction accuracy of the K-Nearest Neighbors algorithm was used as a comparison benchmark with the reported accuracies from the linear classification models. It was found that the Logistic Regression algorithm was 15.09% more accurate on the IMDB Reviews dataset than the K-Nearest Neighbors algorithm but took more time to train. The Multiclass Regression algorithm was 17.89% more accurate than the K-Nearest Neighbors algorithm on the 20 Newsgroups dataset but was also slower to train.

1. Introduction

The goal of this project was to use the Logistic Regression (LR) and Multiclass Regression (MR) machine learning (ML) models to perform text classification on data obtained from two distinct datasets. The Logistic Regression model was evaluated on the IMDB Reviews dataset, while the Multiclass Regression model was evaluated on the 20 Newsgroups. The results concluded that on processed and feature-scaled datasets, the LR and MR classifiers outperformed the benchmark K-NN Model. On the IMDB dataset, LR obtained an accuracy of 84.05%, while K-NN obtained an accuracy of 68.96%. On the 20 Newsgroups dataset, MR reported an accuracy of 93.25% while K-NN reported 75.36%. The detailed performance report of each model can be found in the “Results” section of this document.

2. Relevant Work

To obtain a general idea of data pre-processing techniques, relevant ML models, and prediction accuracy numbers, it was useful to review related work that investigated similar datasets. For IMDB Reviews, [1] retrieved the dataset from the *Kaggle Bag of Words Meets Bag of Popcorn Challenge* and performed HTML tag removal, text normalization, and stopwords removal as data preprocessing steps. After filtering irrelevant features, [1] converted the dataset into the Bag of Words Document Term Matrix[1]. LR evaluation on the dataset obtained an accuracy of 87.28% with an area under the curve (AUC) of 94%. For comparison benchmarks, [1] measured the performance of the Decision Tree (DT) classifier, which underperformed with an accuracy of 71.34% and an AUC of 71%.

For the 20 Newsgroups dataset, [2] used the Spark ML Packages for Large-scale Text Multiclass Classification. The authors first partitioned the 20 Newsgroups topics based on subjects before applying Pyspark ML Usage for feature transformation and model extraction. Much like the IMDB Reviews dataset, the 20 Newsgroups texts were segmented into tokens for filtering irrelevant vocabulary. MR was then used to obtain a general prediction accuracy of 89.64%. Overall, [2] evaluated the performance of the classifier on all 20 Newsgroups and obtained varying accuracies for each one.

3. Datasets

IMDB Reviews Dataset

IMDB Reviews is a class balanced dataset containing 25,000 polar movie reviews and associated sentiment labels for training, as well as 25,000 for testing. The total vocabulary/feature size for the 50,000 reviews was initially 89,526 but was cut down to 1299 after word filtering. The class 1, positive reviews all received a sentiment score >= 5 out of 10, while class 0, negative reviews received a score of < 5.

[1] Row represents the individual review while the columns present the contained words; each cell is made to contain the frequency of the word in the corresponding review.

After inspecting the train/test folders, it was observed that positive reviews often contained appraising vocabulary such as “good”, “love”, and “excellent”, while negative reviews used condescending adjectives such as “terrible”, “bad”, and “awful”. It was therefore concluded that upon completion of feature filtering, similar words should appear in the final dataset.

To perform data pre-processing and feature selection, custom directory/file parsing algorithms were written to obtain better control over the types of words that were selected. For example, to filter out vocabulary appearing in < 1% and >= 50% of the documents, it was necessary to first obtain the word frequency in each of the 50,000 reviews. Additionally, the nltk English stopwords list was used to further filter the dataset. An important design decision that was made for these algorithms was to exclude words with contractions and numbers, as they tended to be specific to the review and appeared rarely. Feature selection was then performed with the simple linear regression hypothesis testing. In particular, features with the most positive and negative z-scores were chosen. The final feature count for the entire dataset was cut down to 257.

20 Newsgroups Dataset

The 20 Newsgroups dataset comprises 18000 newsgroups posts on 20 various topics. For the purpose of this project, four distinct categories were chosen, namely “comp.graphics”, “rec.sport.hockey”, “sci.crypt”, and “soc.religion.christian”. Similarly to the IMDB Reviews dataset, the content of the chosen group categories was first manually analyzed to determine what kinds of words to look out for when filtering features.

To obtain the set of all possible features, each category was tokenized and parsed for unique vocabulary whilst filtering out irrelevant words. For each newsgroup text, a dictionary was used to hold the frequency of each feature (word). Much like the .feat file for IMDB dataset, a list of dictionaries that mapped word indices to frequency count was created. After obtaining the dataset, mutual information was then used to select the top features per class. All classes were then combined for the final dataset.

4. Results

Part 0 - Best learning rate for Logistic Regression prediction on the IMDB Reviews dataset.

The training data was split into 80% and 20% for the training set and validation set respectively. A learning rate of $\alpha = 0.5$ was obtained by evaluating different hyperparameter values on the validation set and selecting the one with the highest prediction accuracy. All the following LR experiments use this α.

Part 1 - Horizontal bar plot of the top 20 IMDB Reviews features with the most (+)vs/(-)ve z-scores.

Figure 1 in the “Figures Appendix” shows the obtained horizontal bar plot of the top 20 features. As predicted during exploratory testing, there are expected positive/negative vocabulary such as “awful”, “special”, “rather”, “fan”, and “love” that have made it to the top 20 and possess a significant influence over how the LR makes its sentiment review. However, neutral words that have little to do with sentiment, such as “anything”, “car”, and “town”, also appear in the top 20 list. This shows that filtering using z-scores has its drawbacks, as the technique seems to retain unexpected vocabulary to provide better performance. The experiment in Part 9 of this section uses a different technique that tests if selecting features based on if they are labeled as positive or negative will have any effect on model performance.

Part 2 - Convergence plot for Logistic and Multiclass Regression.

To test the gradient correctness of the LR implementation, small perturbation was utilized on the IMDB Reviews dataset. In small perturbation, the approximated gradient was compared with the analytical gradient to obtain a difference of ~8.79e-24. The small difference signifies that the gradient calculations are accurate. Additionally, to observe that the model converges, the cross entropy should be strictly decreasing. Figure 2 plots cross entropy as a function of iteration and shows no oscillatory behavior, which means that the model is exhibiting an acceptable performance level.

To verify the correctness of the MR model, the gradient was computed over small perturbation. The small difference of 2.39e-14 signifies that the loss function is correct. Moreover, Figure 3 displays smooth decreasing curves for training and validation loss, signifying that overfitting does not occur. This further supports the claim that the MR model was implemented correctly.

Part 3 - Logistic Regression receiver operating characteristic curve (ROC)
To test prediction accuracy of the LR model, the classifier was fitted with two versions of the filtered IMDB dataset. Using the 1299 set of unfiltered features, a dataset of 257 features was created using z-score selection and a dataset composed of 257 randomly selected features was created using random feature selection. As a benchmark dataset, the Sklearn K-Nearest Neighbors was used on the z-score-selected features.

As observed from the reported accuracies, the predictions on z-score-selected features are higher than for randomly-selected features; although the values are within a similar range. This similar predictability behavior can be explained by the fact there is a high probability of randomly selecting the majority of causal features out of the 1299 initially filtered features[2].

- **Logistic Regression: highest z-score-selected features:** accuracy = 84.05%.
- **Logistic Regression: randomly-selected features**: accuracy = 70.72%.
- **Scikit K-NN: highest z-score-selected features:** n_neighbors=15; accuracy = 68.96%.

To visualize the performance of the LR model, it was useful to compare it with other ML classification methods and observe the reported accuracies and AUROC. The K-Nearest Neighbors and Decision Tree (DT) classifiers were chosen as benchmark datasets. Figure 4 displays the ROC curves.
As can be observed from the generated graph, all methods predict better than the random classifier (blue dashed line), but the Logistic Regression classifier trumps the performance of the other two models. K-NN is sensitive to random class-irrelevant features and often works poorly on high-dimensional data. The 257 features is quite a large dimension and does contain irrelevant features, which explains the poor performance of K-NN. The poor performance accuracy of the DT classifier could be explained by the fact that it does not have a way of regularizing data to eliminate features of little to no relevance. The high tree depth of 1000 could have also contributed to the model's ability to generalize unseen data.

Part 4 - A bar plot showing the AUROC of Logistic Regression and Sklearn K-NN on the test data as a function of 20%, 40%, 60%, 80%, 100% training data.
AUROC is calculated as a function of 20%, 40%, 60%, 80%, and 100% of the training data. In order to select the data percentages, the train_test_split function was used from the Sklearn model_selection with shuffle=True to prevent overfitting due to the sorted nature of the original dataset. As observed from Figure 5, the reported LR AUROCs for each of the training dataset sample sizes all fluctuate around 81%-83%. This can be explained by the fact that taking even 20% of 25,000 training examples is enough to create a best fitting curve.

Figure 6 reports K-NN AUROCs for each of the training dataset sample sizes. A K-NN classifier generally exhibits good performance when trained with a large amount of data examples. It therefore explains why the reported AUROC is proportional to increasing sample sizes.

[2] The 1299 features were obtained from filtering stopwords, contractions, and least-popular words from the original 89,526 vocabulary list.

Part 5 - A bar plot showing the AUROC of Multiclass Regression and Sklearn K-NN on the test data as a function of 20%, 40%, 60%, 80%, 100% training data.

For this section, the same procedure as in part 4 was followed, Figure 7 displays MR AUROC values of around 88%, 91%, 92%, 93%, 93% for 20%, 40%, 60%, 80%, and 100% of the training data respectively. This shows that a training set corresponding to 20% of the total training data (around 475 samples) is sufficient to fit the model with a performance comparable to any larger training data subset.[3]

Much like in "Part 4", Figure 8 displays increasing K-NN AUROC values of 63%, 68%, 72%, 73%, 75% for 20%, 40%, 60%, 80%, and 100% of the training data respectively. As expected, the more data examples K-NN uses for training, the better its prediction performance.

Part 6 - A horizontal bar plot showing the top 20 features from the Logistic Regression on the IMDB data with the coefficient as the x-axis and the feature names (i.e., words) as the y-axis.

Figure 9 displays the horizontal bar plot for the top 20 features as a function of the highest and lowest LR weights. Much like for Part 1, unexpected words such as "sit", "style", and "moments" show that there is room for improving the model and feature selection techniques.

Part 7 - A heatmap plot showing the 5 most positive features as rows for each chosen Newsgroup.

The generated heatmap in Figure 10 shows the most positive features for each of the 4 Newsgroup categories. In general, the quality of the top 20 words in the heatmap is high (relevant category words), signifying above average MR performance and feature selection.

- **comp.graphics**: "graphics", "images", "files", "color", and "image".
- **rec.sport.hockey**: "game", "hockey", "team", "teams", and "leafs".
- **sci.crypt**: "nsa", "security", "government", "key", and "label".
- **soc.religion.christian**: "god", "jesus", "bible", "people", and "church".

Part 8 - Ridge and Lasso Regression on the IMDB Reviews Dataset

The filtered IMDB Reviews dataset contained significantly more training data than the number of features. This does not simulate real-world data very well, which often contains more features than training examples and leads to overfitting. Ridge and LASSO regression models are meant to reduce model complexity and thereby prevent this overfitting. In order to simulate a real world scenario, the originally unfiltered 1299 features were trained with a small set of examples using both regression models. In addition, 257 filtered (mostly causal) features were selected with an even smaller set of examples and also trained on both Ridge and LASSO regressions.

For both experiments, the alpha hyperparameter was selected with the validation set (a portion of the training set) and the fitted model was evaluated on the test sets. It can be concluded that Ridge regression performs equally well in comparison to LR, LASSO underperforms for this dataset, and that having a large set of examples in comparison to features significantly improves performance for both types of regressions. However, as LASSO is capable of feature selection by significantly shrinking the weights, it is possible that the lower reported accuracy of LASSO for all cases is due to it filtering out most of the causal features by aggressively reducing their weights.

- **Mostly-Causal Features Ridge N>>D:** reported accuracy 83.56%.
- **Mostly-Causal Features Ridge N<<D:** reported accuracy 50.24%.
- **Mostly-Causal Features LASSO N>>D:** reported accuracy 75.26%.
- **Mostly-Causal Features LASSO N<<D:** reported accuracy 55.60%.

- **Irrelevant Features Ridge N>>D:** reported accuracy 86.97%.

[3] For the most consistent results, the learning rate and number of iterations of the model were kept at constant values of 0.0001 (reasonable learning rate according to [4]) and 250.

- **Irrelevant Features Ridge N<<D:** reported accuracy 50.02%.
- **Irrelevant Features LASSO N>>D:** reported accuracy 75.56%.
- **Irrelevant Features LASSO N<<D:** reported accuracy 53.95%.

Part 9 - IMDB Reviews different feature filtering technique

Using z-scores for feature selection has its drawbacks: it may retain unexpected words in order to improve performance. As observed in Figure 1 from Part 1, vocabulary such as "car" and "went" do not belong. In order to see if specifically selecting positive/negative sentiment vocabulary will improve model performance, an experiment was conducted with two sets of text files obtained from [3], in which one set contained positive vocabulary, while the second set contained negative vocabulary. After filtering the dataset of 1299 features to only include words that belong to these two lists, the final feature count ended at 244. After fitting and predicting with LR, it was discovered that the prediction accuracy dropped to 71.06% which is significantly lower than the LR accuracies reported in the previous sections. This shows that using a z-score for feature selection is a better technique to selecting feature vocabulary based on whether a word is considered positive or negative.

Additionally, to verify if the top 20 z-score-selected features would improve with this new filtering technique, z-score filtering was further applied on the dataset of 244 features. The final feature count was 44 and the prediction accuracy further dropped to 69.03%. This shows that more features improves the fit of the model and therefore allows for better prediction. Moreover, Figure 11 shows that the z-score does select more sentiment-related vocabulary. This can be explained by the fact that only sentiment-related vocabulary was left after initially applying the new filtering technique - so this improvement is actually a false positive. It can therefore be concluded that using a list of positive/negative vocabulary to select features does not improve performance.

5. Discussion and Conclusion

Although data pre-processing and feature filtering tasks were accomplished, the time it takes to generate all the necessary datasets in Part 1 is around 5 minutes. In an improved iteration of the project, the time could be significantly cut down by using more efficient data structures and algorithms. Additionally, the use of RAM storage of the datasets in Collab was quite significant - at some point during the experiments the usage almost reached the provided 15 GB. It would therefore be useful to find more compressed storage strategies for datasets of size akin to IMDB and 20 Newsgroups.

Despite any additional potential improvements, the overall task of testing the performance of the LR and MR models was accomplished. The data was acquired, analyzed, and preprocessed to include the most relevant features. The ML models were then fully implemented and classifier performance was evaluated on two benchmark datasets.

6. Statement of Contributions

Everyone contributed an equal amount of effort. All team members initially implemented each ML model individually, however towards the end, each member took a specific model, improved it, and performed the testing experiments. The report was also written and previewed by everyone in the group.

References

[1] S. Tripathi, R. Mehrotra, V. Bansal and S. Upadhyay, "Analyzing Sentiment using IMDB Reviews dataset," *2020 12th International Conference on Computational Intelligence and Communication Networks (CICN)*, 2020, pp. 30-33, doi: 10.1109/CICN49253.2020.9242570.

[2] S. Wang, J. Luo, L. Luo, "Large-scale Text Multiclass Classification Using Spark ML Packages", *Journal of Physics: Conference Series,* 2022, https://iopscience.iop.org/article/10.1088/1742-6596/2171/1/012022/meta.

[3] Bing Liu. "Sentiment Analysis and Subjectivity." An chapter in
Handbook of Natural Language Processing, Second Edition,
(editors: N. Indurkhya and F. J. Damerau), 2010.

[4] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. "Deep learning (adaptive computation and machine learning series)." *Cambridge Massachusetts* (2017): 321-359. Page 434.

Figures Appendix

FIGURE 1

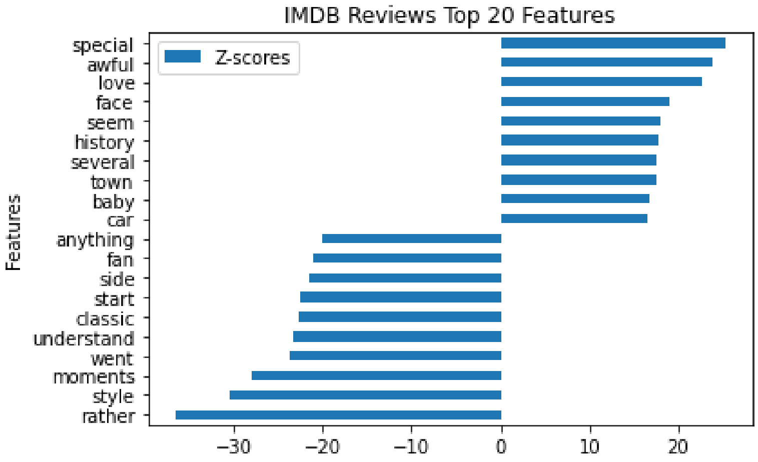


FIGURE 2

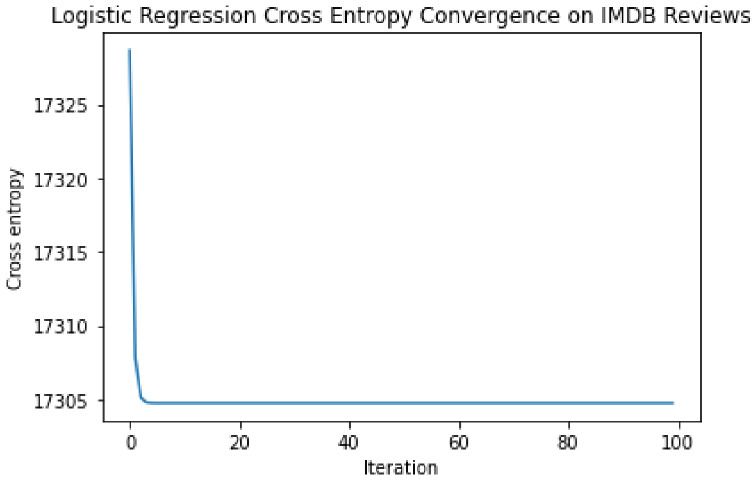


FIGURE 3

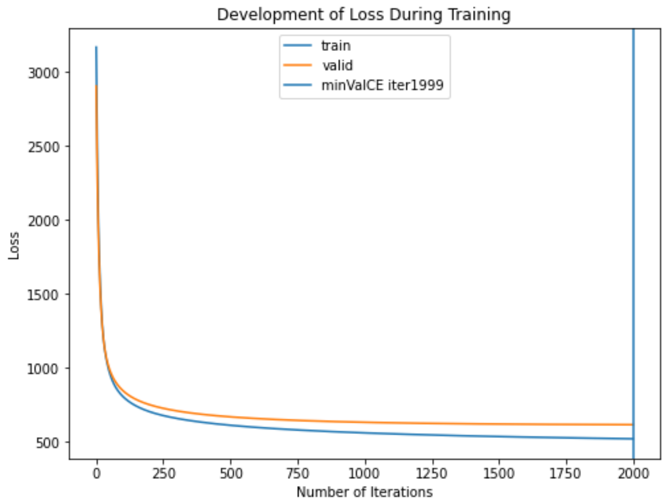


FIGURE 4

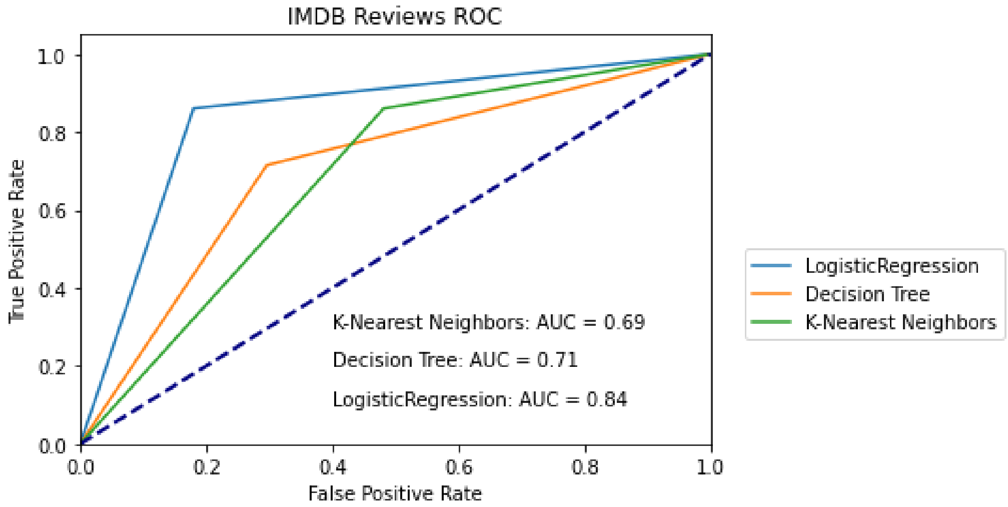
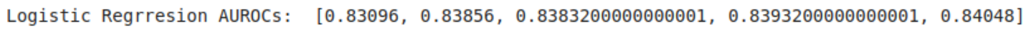


FIGURE 5



```
Logistic Regrresion AUROCs:  [0.83096, 0.83856, 0.8383200000000001, 0.8393200000000001, 0.84048]
```

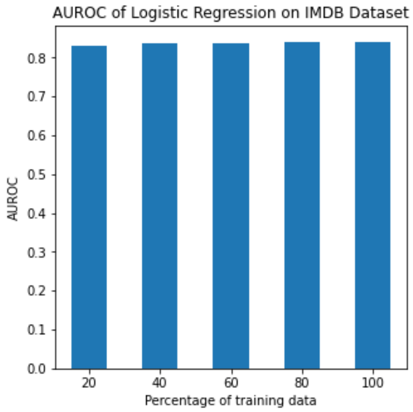
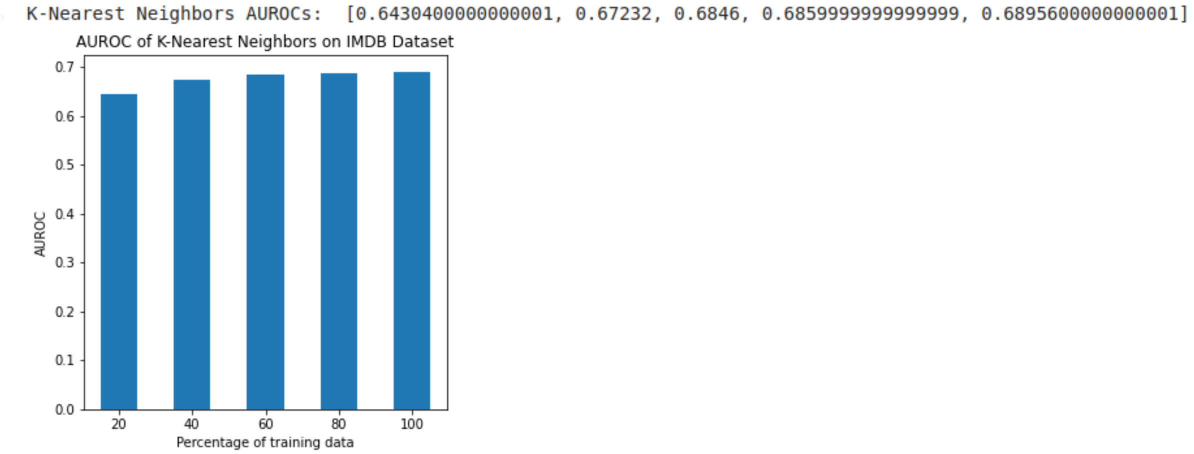


FIGURE 6



```
K-Nearest Neighbors AUROCs:  [0.6430400000000001, 0.67232, 0.6846, 0.6859999999999999, 0.6895600000000001]
```

AUROC of K-Nearest Neighbors on IMDB Dataset
AUROC
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
20
40
60
80
100
Percentage of training data

FIGURE 7

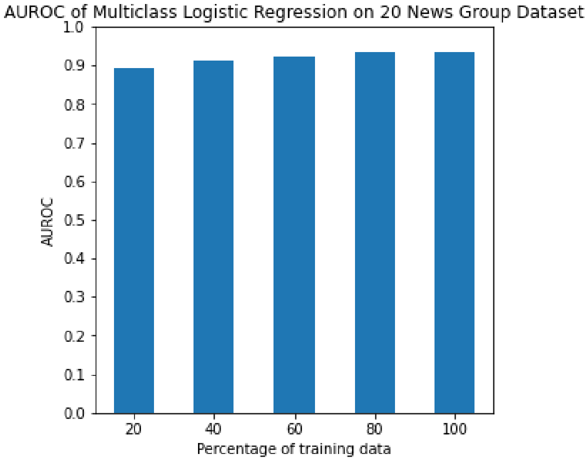


FIGURE 8

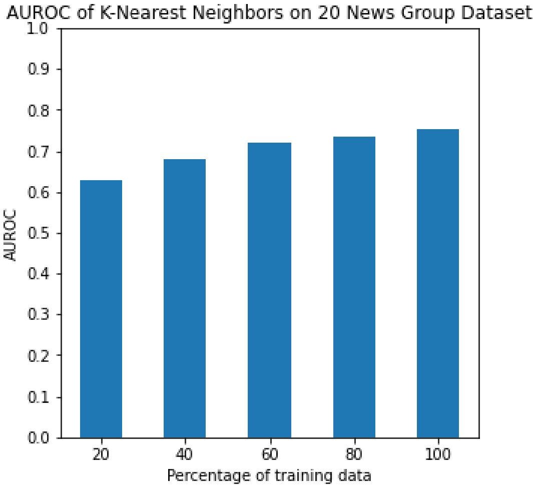


FIGURE 9

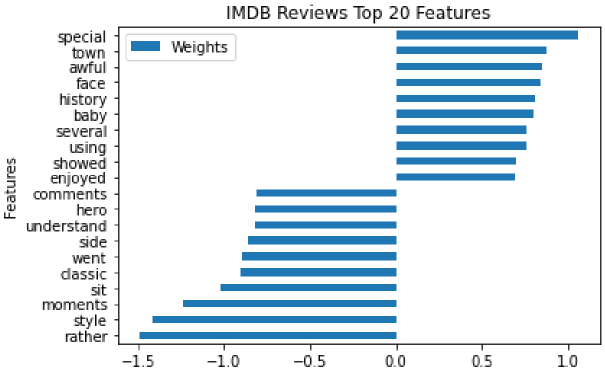


FIGURE 10

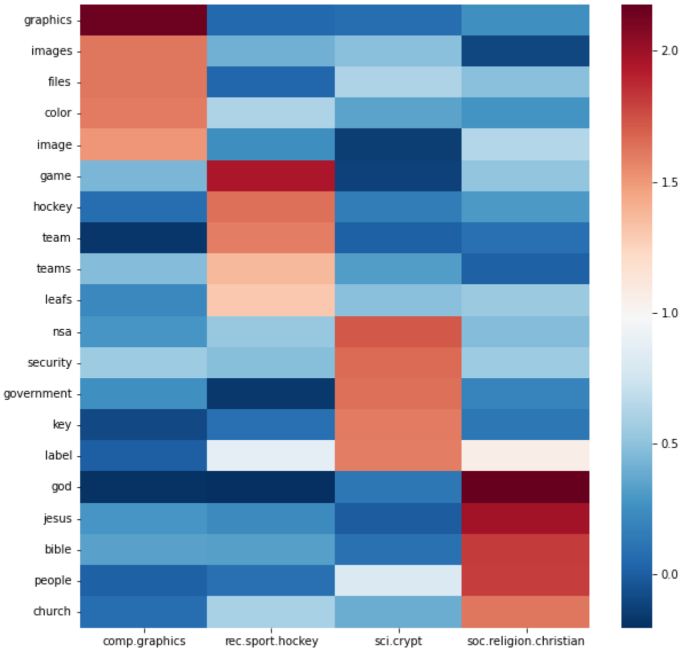
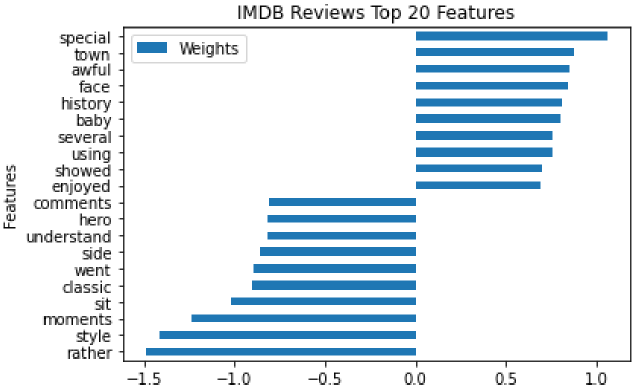


FIGURE 11


IMDB Reviews Top 20 Features
Weights
Features
special
town
awful
face
history
baby
several
using
showed
enjoyed
comments
hero
understand
side
went
classic
sit
moments
style
rather