Binary and Multi-class Classification of Movie Reviews and News Textual Data

Group 47 - Preetesh Rambarun, Ben Cheung, Andrew Chao

Abstract—Classification is one of the most common supervised learning tasks of machine learning, of which regression is a widely used method to implement a classifier. In this assignment, we explore and build two regression models: Logistic Regression and Multi-class Regression. We train and test our implementations on the two given datasets, IMDB Movie Reviews and 20 Newsgroups. To improve the performances of the classifiers, we perform feature selections by employing techniques such as Simple Linear Regression hypothesis testing and LASSO regression. Next, using validation sets, we tune the classifier parameters to improve accuracy and efficiency as well as to avoid overfitting. We perform various experiments to investigate the effects of different model parameters, feature selection and dataset preprocessing methods. As a benchmark, we compare the performance of our regression-based classifiers with the scikit-learn KNN model, in which we observe significantly better model performance for both datasets.

1 Introduction

Textual information becomes increasingly important nowadays as big-data analysis has found wide ranges of daily applications and immense business potential. However, textual information mostly comes in huge quantities in the modern digital world and is evidently overwhelming for manual handling. To extract and organize critical information from a large amount of textual data, *text classification* is an important task in ML. This includes filtering through large datasets, categorizing them by the desired parameters (e.g., topics, urgency, language) and organizing an informative summary for the end-users, such as movies rating and news article categories. [1]

In this assignment, we work with two large datasets: the Internet Movie Database (IMDB) and the 20 Newsgroups dataset (Newsgroups). IMDB has a total of 465 million items that provides entertainment selections to users including user ratings about films and TV contents [2], while Newsgroups contain 20,000 documents of articles about 20 different topics. [3]

We develop a logistic model for binary classification on the IMDB dataset, and a multi-class model for classification on the Newsgroups dataset. We fine-tune the models by selecting the classifier parameters with validation sets, and train the models on the given datasets with prior feature selection.

For the IMDB dataset, our logistic model has better accuracy and significantly outperforms KNN in training time. For the Newsgroups dataset, our multi-class model achieves considerably better accuracy but longer training time than KNN.

1.1 Related Works

Maas, Andrew, et al [2] introduced the IMDB dataset in a study for sentiment analysis, which classifies reviews as negative or positive. The IMDB dataset has been used in previous research to investigate methods for text classification. In [4], the authors compared the performances of logistic regression and RNN deep learning on the IMDB dataset about sentiment analysis. The results showed that the RNN algorithm is more efficient because it does not require feature extraction and provides higher accuracy (88%) than logistic regression accuracy.

Lang, Ken [3] introduced the Newsgroups dataset which contains a wide range of news documents and researchers have utilized the dataset to experiment with text applications. For example, this research [7] paper uses logistic regression to classify the pre-processed data on the Newsgroups dataset and to investigate the efficiency of the multi-label regression classification. They observed that the classification accuracy and F1-score can reach 89.64% and 89.64% respectively, with 5-fold cross-validation experiments.

2 Models

Regression is a technique that investigates the relationship between independent features and a dependent outcome. Trained regression models can be used for predictions on the outcome of new unseen inputs, which further extends the usage to missing data handling to preserve sample size.

2.1 Logistic Regression

Logistic regression is one of the simplest regression models for binary classification tasks. Typically, a logistic regression model uses the sigmoid function to transform a prediction by regression from an unbounded range to a prediction probability in $[0, 1]$.

We consider Cross-Entropy (CE) as the cost function of the logistic model. CE is a convex function, and it can be interpreted as Bernoulli likelihood. Therefore, CE is a loss function with a good optimization property and a nice probabilistic interpretation.

The CE function measures the distance between the predicted outcome and the true label. In order to improve the model accuracy, the model aims to minimize the CE loss function. Since there is no closed form for the minimizer, we accomplish the minimization by gradient descent.

2.2 Multi-class Regression

Multi-class regression extends logistic regression for classification tasks with more than two classes. The multi-class regression model uses the softmax function to define the probabilistic outcome for each class.

Like logistic regression, we use CE as the cost function for optimization. In this case, the CE is equal to the sum of log-predictions of all classes :

$$J(W) = -\sum_{c=1}^{C} y_c \log \hat{y}_c.$$

Similarly to logistic regression, we perform gradient descent to minimize the cost and update the multi-class regression model.

2.3 Gradient Descent

For both of our models, we perform the optimization by gradient descent. We implement 4 stopping criteria for the gradient descent snippet:

1) *Maximum iterations*: The number of iterations exceeds the maximum iterations designated;
2) *Increasing validation error*: The validation error starts increasing and causes overfitting;
3) *Small gradient*: The gradient is smaller than a chosen threshold;
4) *Small cost change*: The change in cost function is below a threshold;

For conditions (3) and (4), we measure the quantities in a relative(multiplicative) sense: the gradient or cost change is regarded to be small if the quantity after the update is within a certain percentage of the old value. Concretely, the gradient descent stops with (3) if

$$\frac{|\|g_{new}\|_2 - \|g_{old}\|_2|}{\|g_{old}\|_2} \times 100\% < \theta_{grad}\%,$$

where g_{old} and g_{new} are the gradient vectors before and after updates, and θ_{grad} is the gradient change threshold. The gradient descent stops with (4) if

$$\frac{|L_{new} - L_{old}|}{L_{old}} \times 100\% < \theta_{loss}\%,$$

where L_{old} and L_{new} are the losses before and after updates, and θ_{loss} is the loss change threshold.

2.4 Model Parameter Selection

For our models, there are four parameters to finetune for the training: learning rate (α), the maximum number of iterations (N_{max}), gradient norm tolerance (θ_{grad}) and cost function change tolerance (θ_{loss}).

The choice of α is a crucial for efficiency as convergence may be too slow if α is too small. If α is too large, a large oscillation may be created and the solution is overshot. Furthermore, the validation error starts increasing at an earlier stage as α increases, showing signs of over-fitting.

The maximum number of iterations ensures that gradient descent terminates in a reasonable amount of time. The two tolerance parameters decide the convergence criterion: for a sufficiently small gradient or cost function change, it is regarded that the algorithm converges. A suitable choice of tolerance parameters ensures an efficient convergence without deterring the accuracy due to early stopping.

3 DATASETS

3.1 IMDB Reviews Dataset

3.1.1 Data Inspection

We load the training and testing sets for the IMDB reviews from the "labeledBow.feat" files. We observe that each dataset contains 25000 reviews (instances). Since the dataset contains 89526 vocabularies (features), it requires further processing before training.

3.1.2 Data Processing

We filter out words with a proportion of occurrence above the stopwords threshold (50%) and below the rare-words thresholds (1%). We then perform feature selection using absolute z-scores following the simple linear regression hypothesis testing principle. By inspecting the distribution of absolute z-scores (Appendix A.1), we find that the majority of absolute z-scores is below 10, except for some individual features with absolute z-scores as large as over 40.

We select the features with the highest absolute z-scores (top 100 words for the basic dataset). By inspecting the words selected, we observe that words with the most positive or most negative z-scores are relatively strong indicators of a good (e.g., "great", "wonderful",

"excellent") or bad (e.g., "bad", "worst", "waste") movie, respectively. The list of chosen words also contains some more neutral words (e.g. "?", "even", "minutes").

3.1.3 LASSO Feature Selection

In addition to the requirement of the assignment, we also perform LASSO feature selection for the binary classification task on the IMDB dataset. The LASSO feature selection has the advantage of sparse selection. We keep the features that have non-zero coefficients by LASSO selection, in which we obtain a dataset with 94 features retained.

3.2 20-News Group Dataset

3.2.1 Data Inspection

We import the training and test data from the sklearn package (datasets.fetch_20newsgroups) and remove headers, footers and quotes for each document. The dataset contains 20 available newsgroups, each addressing a different topic. We choose four easily distinguishable categories - "comp.graphics", "rec.sport.hockey", "sci.med", and "soc.religion.christian". The data contain a moderate number of documents (2377 for training set and 1582 for testing set) and a large number of features (30945 for training set and 26892 for testing set).

3.2.2 Data Processing

We use CounterVector to convert the texts into numerical feature vectors in order to train our model ("tokenization"). We use the same stopwords and rare-words thresholds to eliminate less important features. After filtering, the number of features retained is 1170.

To further process the data, we use mutual information (MI) score to select the most important words for each of the four classes. A higher MI score indicates a higher dependency of the outcomes on the variables. We select the top 100 feature words for each class, and the combined list of the selected features based on individual classes contains 277 features. This reduces the dataset size reasonably for effective training.

4 Results

4.1 IMDB Dataset

4.1.1 Linear Regression Hypothesis Testing

We generate a bar plot, Figure 1, for the IMDB data showing the feature with the 10 most positive and 10 most negative z-scores. By inspection, the top positive (resp. negative) words are mostly sensible strong indicators for good (resp. bad) movies.

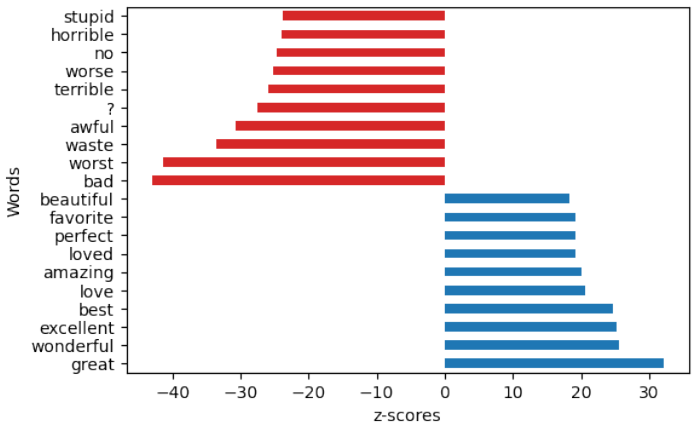

Fig. 1. **Top 10 positive (blue) and 10 negative (red) features for IMDB dataset**

4.1.2 Logistic Regression Convergence

We check the gradient computation with a small perturbation, and the perturbation factor is very small (1.9216×10^{-18}), verifying that the gradient calculations and CE are correct.

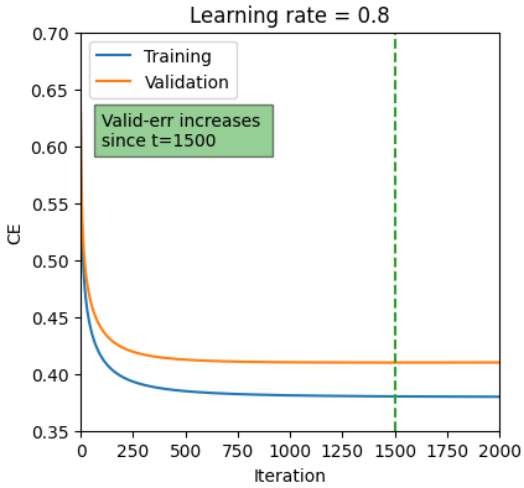

Fig. 2. **Logistic regression convergence** ($\alpha = 0.8$)

We vary the learning rate α and observe a decreasing validation error for $\alpha = 0.2$ and 0.5. The validation error starts increasing before maximum iterations as α surpasses 0.8. Figure 2 shows the α= 0.8 convergent behaviour. The convergence plots (Appendix A.2) suggest that a reasonable α is less than 0.8.

We also check the AUROCs and run-time for different learning rates and tolerance parameters with the validation set by trial and error. The results are in Appendix A.3 and A.4. We determine that the best parameters for logistic regression on IMDB are: $\alpha = 0.6$, $N_{max} = 1000$, $\theta_{grad} = 0.2$, $\theta_{loss} = 0.02$.

4.1.3 Logistic and KNN Model Performance

We compare our logistic model performance to the sklearn-KNN model based on AUROCs and run-

time. The run-time for the logistic model (1.87 s) is significantly shorter than that for KNN (16.04 s).

Figure 3 shows a plot of the ROC curves of two models on the IMDB test data. We observe that the logistic regression substantially outperforms KNN in both prediction performance and run-time for the IMDB dataset.

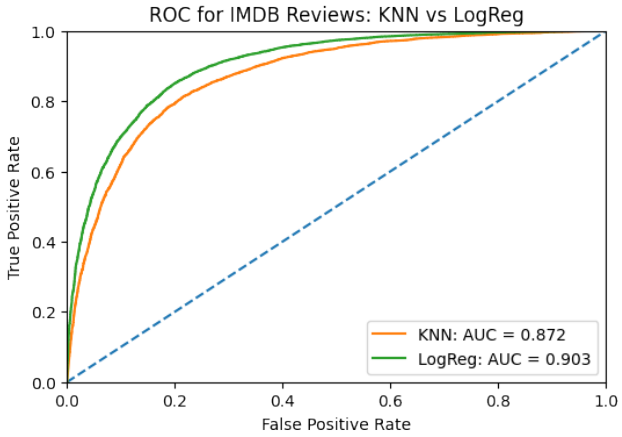

Fig. 3. **ROC curves for logistic regression and KNN model**

The bar plot in Figure 4 shows the AUROCs of the two models on different training data proportions. We observe that for every proportion of training data used, the AUROC of logistic regression is around 0.9, which is higher than that of KNN (around 0.87).

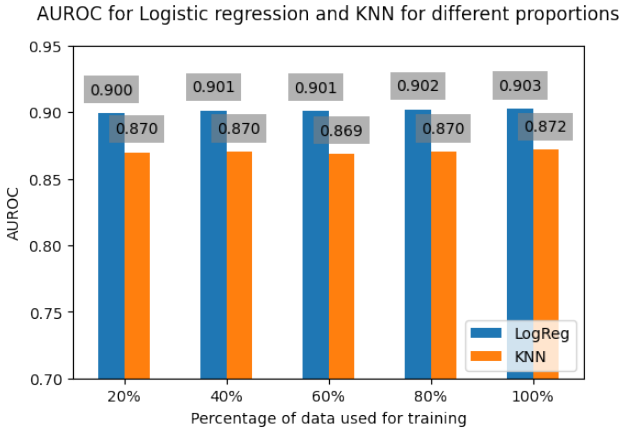

Fig. 4. **AUROC of Logistic Regression and KNN with different percentages of training data**

4.1.4 *LASSO Feature Selection*

We also use the scikit-learn LASSO linear model for feature selection with weight penalty tuned. We find that the AUROC achieved by the LASSO-based dataset is lower (around 77 %) than the z-score-based dataset (see Appendix A.7). We observe that there is only a small overlap (Appendix A.8) between the 94 features chosen by LASSO and 100 features chosen by z-scores.

4.2 Newsgroups Dataset

4.2.1 *Multi-class Regression Convergence*

The perturbation factor, from small perturbation on the gradient, is very small (1.8408×10^{-11}). This verifies the gradient and loss function calculations.

By varying the learning rate (α), we observe an increase in validation error when $\alpha = 9 \times 10^{-4}$ (with $N_{max} = 500$). From the convergence plots (complete results in Appendix B.1), this suggests that a reasonable α is less than 9×10^{-4}. Figure 5 shows an example of the convergent plots of our multi-class model.

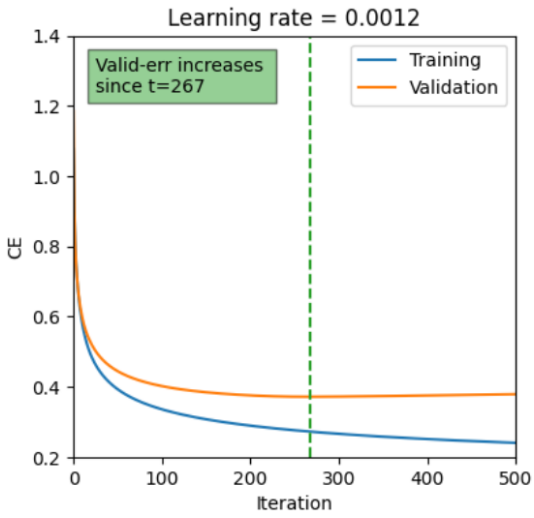

Fig. 5. **Multiclass regression convergence plot with** $\alpha = 0.0012$

We also check the accuracy and run-time for different learning rates and tolerance parameters with the validation set. The results are in Appendix B.2 and B.3. We determine that optimal parameters are: $\alpha = 6 \times 10^{-4}$, $N_{max} = 500$, $\theta_{grad} = 0.1$, $\theta_{loss} = 0.1$.

4.2.2 *Multi-class and KNN Model Performance*

The bar plot in Figure 6 shows the accuracy of the two models on different training data proportions.

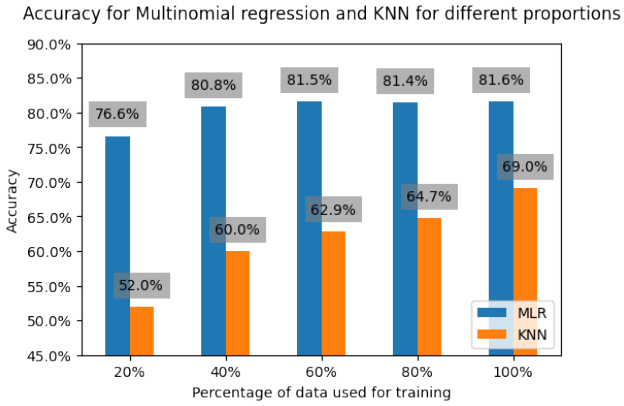

Fig. 6. **Accuracy of Multiclass Regression and KNN with different percentages of training data**

We observe that for every proportion of training data used, the accuracy of multi-class regression is around 81%, which outperforms the best accuracy for KNN (69.0%). For the run-time, our multi-class model (9.52 s) is considerably slower than KNN (0.18 s).

Figure 7 shows the heat map based on the highest weights for the multi-class classifier. We have the top 5 most positive features as rows and each class as columns. We observe that the features with the top weights for each class are indeed related to the class.

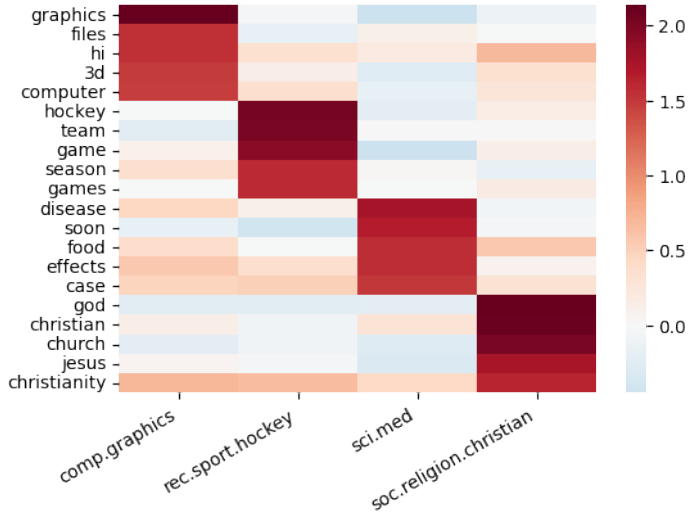

Fig. 7. **Heat map for Newsgroup data for each class**

5 Discussion and Conclusion

5.1 Model Interpretability

We observe that the training results of our models are easily interpretable. For the IMDB dataset, the z-scores selection method based on the linear regression principle supplies features that are strong indicators for good and bad movies in an intuitive sense. For the Newsgroup dataset, in the heatmap generated based on the most significant weights per class, the most important features determined by the multi-class model are indeed strongly related to the respective classes. This reflects the advantage of the high interpretability of regression-based classifiers.

5.2 Feature Selection

For the IMDB dataset, we notice that the performance of logistic regression is better with features selected by z-scores as compared to the features selected by LASSO. This may be attributed to the black-box nature of LASSO algorithm, in which one can pose little manual intervention to prevent certain features from being eliminated or selected. Also, only the reviews with ratings 0 - 4 and 7 - 10 are presented for training, the missing range might cause the dataset to be less suited for a regression-based method like LASSO.

5.3 Dataset Selection and Preprocessing

We have experimented with the different choices of datasets based on the number of features selected, data standardization and frequency-capping. We refer readers to Appendix A.6 and B.5 for full discussions. We remark that there is no universal best choice of selection and preprocessing method.

5.4 KNN Performance

We observe that KNN has an inferior prediction power on both datasets. This could be attributed to the fact that KNN is a lazy learner that acquires no prior knowledge of suitable weights of features, so the distance-based prediction takes little consideration of the importance of features.

On the other hand, for the run-time, we note that KNN runs significantly faster for Newsgroups dataset (with 2377 training data) as compared to the IMDB dataset (with 25000 training data). This is expected as KNN makes predictions by computing distances with all training data to find the nearest neighbours. As seen in the results, KNN performance deteriorates with a smaller proportion of training data used, so training on a partial training dataset is not a viable enhancement for KNN. This reveals the efficiency limitation of KNN on large training sets.

5.5 Future Works

For further improvement in the z-score selection method with the IMDB dataset, one may tokenize the dataset phrase-based instead of word-based. The current word-based approach results in high absolute z-scores for certain neutral words, and this might be due to the tendency of such words to appear in a strong indicator phrase. For example, ”money” is a neutral word, while the phrase ”a waste of money” is a strong indication of a bad movie. Phrase-based tokenization can capture these instances more accurately.

One may also explore the use of principal component analysis for dimensionality reduction and feature selection. The effects of Group LASSO or elastic net regression may also be investigated.

6 Statement Of Contributions

All members contributed to writing the report and discussing ideas. Ben implemented the data prepossessing, and set up the Logistic and Multi-class models. Andrew implemented LASSO feature selection functions and contributed to model evaluation. Preetesh performed related works research, literature review, and model and results analysis.

REFERENCES

[1] Hsu, Bi-Min. "Comparison of supervised classification models on textual data." Mathematics 8.5 (2020): 851.

[2] Maas, Andrew, et al. "Learning word vectors for sentiment analysis." Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies. 2011.

[3] Lang, Ken. "Newsweeder: Learning to filter netnews." Machine Learning Proceedings 1995. Morgan Kaufmann, 1995. 331-339.

[4] K. Amulya, S. B. Swathi, P. Kamakshi and Y. Bhavani, "Sentiment Analysis on IMDB Movie Reviews using Machine Learning and Deep Learning Algorithms," 2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT), 2022

[5] Nithin, V. R., et al. "Predicting movie success based on IMDb data." International Journal of Data Mining Techniques and Applications 3 (2014): 365-368.

[6] Osojnik, Aljaž, Panče Panov, and Sašo Džeroski. "Multi-label classification via multi-target regression on data streams." Machine Learning 106.6 (2017): 745-770.

[7] Wang, Shuang, Jian Luo, and Liangfu Luo. "Large-scale Text Multiclass Classification Using Spark ML Packages." Journal of Physics: Conference Series. Vol. 2171. No. 1. IOP Publishing, 2022.

APPENDIX A
IMDB DATASET

A.1 Z-score Distribution

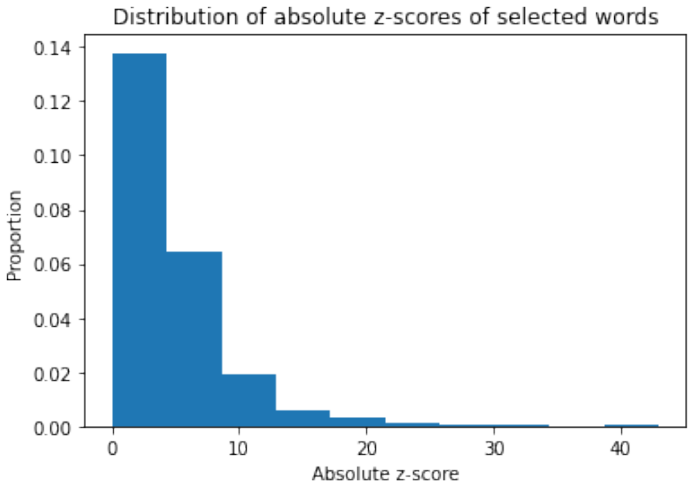

Fig. 8. **Z-score distribution for the IMDb dataset**

A.2 Convergence plots

For varying learning rates, we observe that as the learning rate increases, the AUROC slightly improves while the run-time decreases.

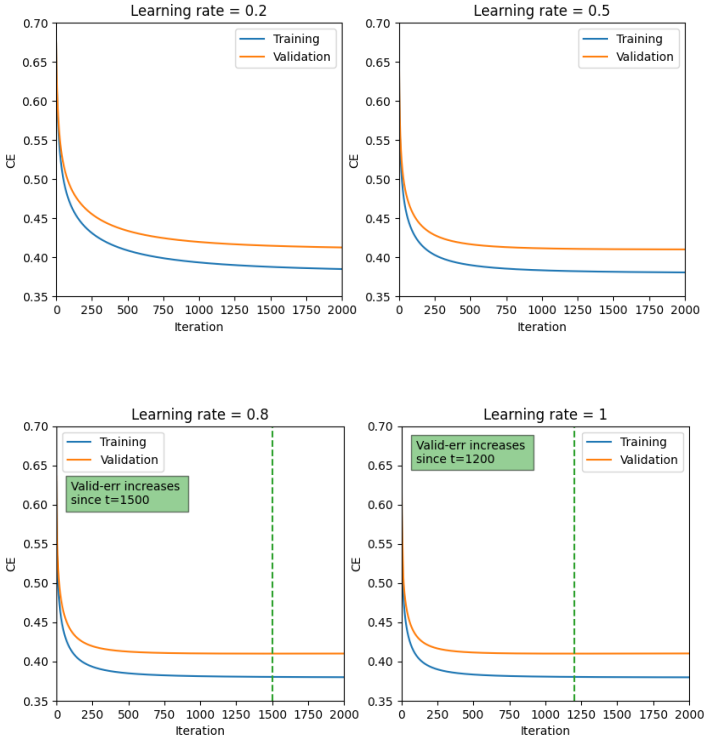

Fig. 9. **IMDB convergence plot with varying learning rates**

A.3 Effects of Varying Learning Rate

We test the logistic regression model performances on varying learning rate, gradient error threshold and cost change threshold. For varying learning rate, we

observe that as the learning rate increases, the AUROC slightly improves while the run-time decreases.

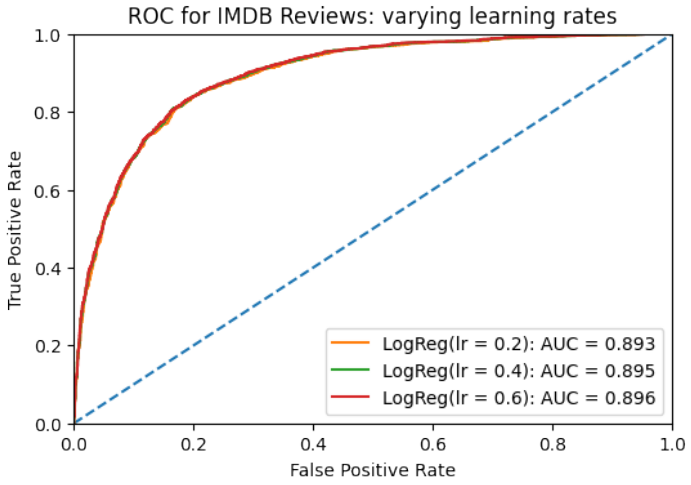

Fig. 10. **ROC for the IMDB data with varying learning rate**

TABLE 1
Effects of varying learning rate for Logistic regression

α	Time (s)	Stopping iteration	Termination method
0.2	5.59	918	Small cost change
0.4	4.90	663	Small cost change
0.6	4.55	539	Small cost change

A.4 Effects on ROC with varying thresholds

For gradient error threshold, we observe that as the threshold increases to 0.2, it becomes more likely that the training terminates with small change in gradient, however the AUROC is observably lower as the threshold passes 0.5. This could suggest that the stopping condition is set too early before convergence. On the other hand, the run-time decreases with an increased threshold.

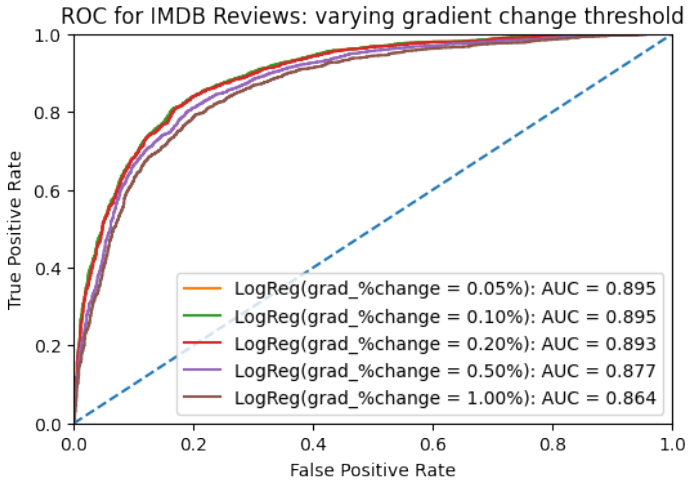

Fig. 11. **ROC with varying gradient change threshold**

For cost function change threshold, we observe that as the threshold increases, both the run-time and the AUROC decrease. The drop in AUROC could suggest that the stopping condition is set too early before convergence.

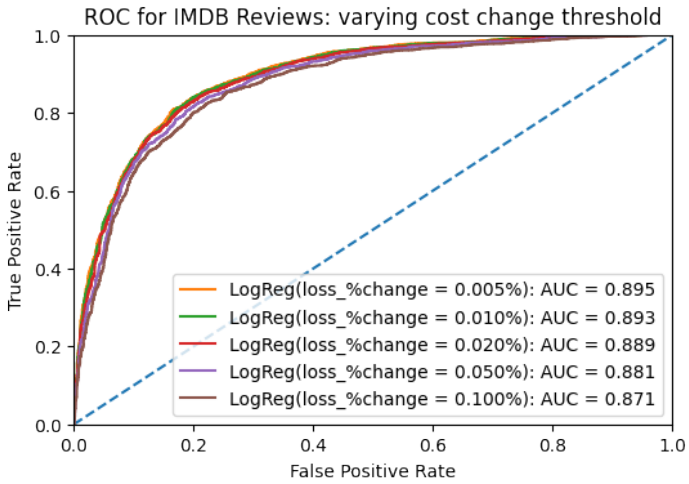

Fig. 12. **ROC with varying cost change threshold**

A.5 Logistic and KNN Model Comparison

The following chart shows the run-times for both models on different training proportions. We observe that for every proportion of training data used, the training time of logistic regression is around half of the training time of KNN.

TABLE 2
Run-time for Logistic regression and KNN for different proportions of training data

Model	20%	40%	60%	80%	100%
Logistic	0.46 s	0.84 s	1.16 s	1.61 s	1.90 s
KNN	3.37 s	6.31 s	9.26 s	13.52 s	15.60 s

A.6 Different selection and processing methods

We compare the logistic regression performance on each dataset with a different set of classifier parameters. The datasets are:

- Original: the dataset chosen by selecting the features with top-100 absolute z-scores;
- Standardized: the dataset by standardizing the feature array of Original dataset;
- Occurrence-based: the dataset with word frequencies replaced by occurrences (1 if the word appears in the document, 0 otherwise);
- 300-feature: the dataset chosen by selecting the features with top-300 absolute z-scores;
- 50-feature: the dataset chosen by selecting the features with top-50 absolute z-scores;

- 10-feature: the dataset chosen by selecting the features with top-10 absolute z-scores;
- 100-random-feature: the dataset chosen by selecting 100 features randomly from the filtered dataset.

We observe that original dataset and standardized dataset achieve a comparable AUROC. The performance worsens as the number of features selected decreases. All of these perform drastically better than choosing random features for classification.

TABLE 3
Different data selections and processing methods

	50% data		100% data	
	AUROC	**Time (s)**	**AUROC**	**Time (s)**
Original	0.901	1.03	0.903	1.85
Standardized	0.907	2.00	0.907	3.74
Occurrence-based	0.897	0.82	0.898	1.61
300-feature	0.926	3.31	0.927	9.00
50-feature	0.880	1.37	0.882	2.33
10-feature	0.804	0.65	0.804	0.99
100-random-feature	0.692	3.26	0.695	6.10

A.7 ROC for LASSO selection

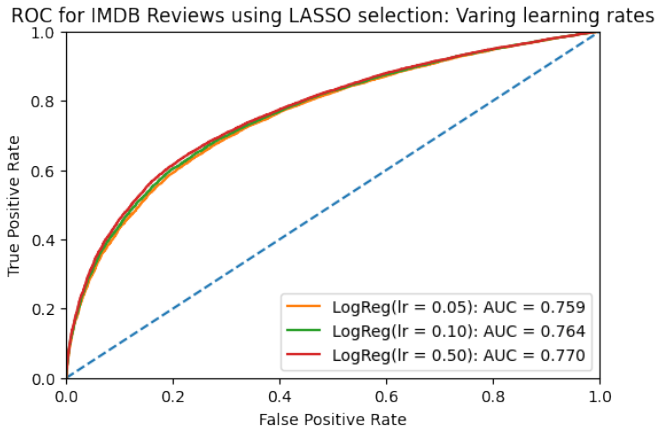

Fig. 13. **ROC for LASSO selection with varying learning rate**

A.8 LASSO feature selection

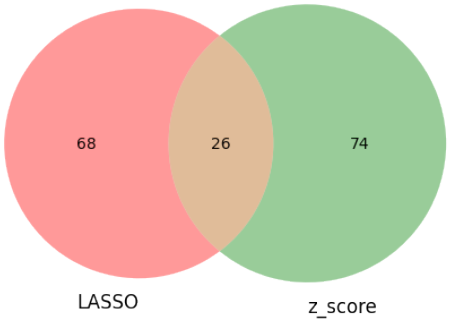

Fig. 14. **LASSO and z-score feature overlap**

APPENDIX B
NEWSGROUPS DATASET

B.1 Convergence plots

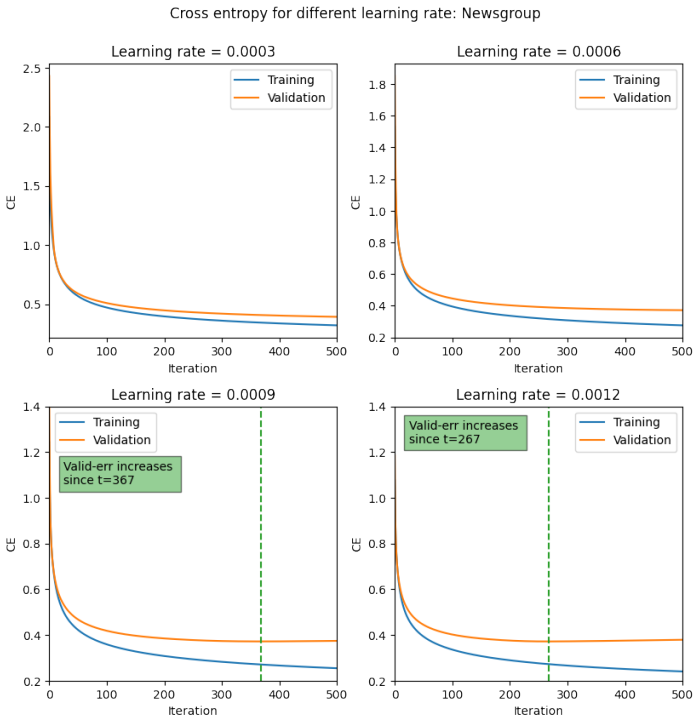

Fig. 15. **Newsgroup convergence plot with varying learning rates**

B.2 Effects of varying learning rate

TABLE 4
Varying learning rates for Multi-class Regression

α	**Accuracy**	**Time (s)**	**Iteration**	**Termination**
5×10^{-4}	87.18%	14.29	428	Small cost
6×10^{-4}	87.61%	15.38	422	Small cost
7×10^{-4}	87.39%	13.78	417	Small cost

B.3 Effects of accuracy for varying thresholds

TABLE 5
Varying gradient change threshold for Multi-class Regression

θ_{grad}	**Accuracy**	**Time (s)**	**Iteration**	**Termination**
0.10%	87.61%	20.96	422	Small cost
0.15%	87.61%	15.51	422	Small cost
0.20%	87.18%	10.78	325	Small gradient
0.25%	86.76%	8.72	261	Small gradient
0.30%	86.13%	8.71	219	Small gradient

TABLE 6
Varying loss change threshold for Multi-class Regression

θ_{loss}	Accuracy	Time (s)	Iteration	Termination
0.05%	86.76%	8.72	261	Small gradient
0.10%	86.13%	7.71	222	Small cost
0.15%	84.87%	5.44	154	Small cost
0.20%	85.29%	3.97	119	Small cost

B.4 Multi-class and KNN Model Comparison

We observe that for every proportion of training data used, KNN performs significantly faster.

TABLE 7
Run-time for multi-class regression and KNN for different proportions of training data

Model	20%	40%	60%	80%	100%
MLR	2.54s	6.59s	7.24s	8.09s	9.52
KNN	0.04s	0.13s	0.09s	0.12s	0.18s

B.5 Different selection and processing methods

We compare the multi-class regression performance on datasets selected and preprocessed by different methods.

- Original: the dataset chosen by selecting the union of features with top-100 MI per class;
- Standardized: the dataset by standardizing the feature array of Original dataset;
- Occurrence-based: the dataset with word frequencies replaced by occurrences (1 if the word appears in the document, 0 otherwise);
- Top-20-per-class: the dataset chosen by the union of features with top-20 MI per class;
- Top-200-per-class: the dataset chosen by the union of features with top-200 MI per class;
- 200-random-feature: the dataset chosen by selecting 200 features randomly from the filtered dataset.

TABLE 8
Different data selections and processing methods

	50% data		100% data	
	Accuracy(%)	**Time (s)**	**Accuracy(%)**	**Time (s)**
Original	0.800	5.15	0.816	9.43
Standardized	0.800	5.60	0.816	9.42
Occurrence-based	0.810	6.13	0.824	7.60
Top-20-per-class	0.795	1.76	0.757	0.77
Top-200-per-class	0.790	1.69	0.798	3.25
200-random-feature	0.587	5.04	0.597	6.30

We observe that original dataset and standardized dataset achieve comparable accuracy. The highest accuracy is from a word occurrence-based data selection.

Unlike in IMDB, with more (200 per class) or fewer (20 per class) features selected than the standard dataset (Original), the accuracy is lower than that of Original dataset. The performances of Top-n-per-class datasets may be explained by the number of total features selected. While Top-200-per-class dataset selects the most features, the percentage of features chosen (from the theoretical maximum $= 200 \times 4 = 800$) is 62.75%, which is substantially lower than the two other datasets (74.75% for Original, 86.25% for Top-20-per-class). This may imply that a portion of the features chosen appears as top features for more than one class, so these features do not help (or hinder) the classification.