Text Review Sentiment Analysis using Classification Modelling vs Regression Modelling – Modelling Analysis and Results Report

CONTRIBUTORS: ADAM CLEAVER BENG, ING. LUKAS TOPINKA, M.SC.
DATASCIENTEST SUPERVISOR: MAËLYS BASSILEKIN BLANC

Introduction

This report summarizes the results of machine learning models implemented to predict the sentiment of a given review and correctly assign the corresponding number of review stars. The data preprocessing is described in the separate supporting report: Text Review Sentiment Analysis using Classification Modelling vs Regression Modelling - Data Quality Report

Models Implemented

Regression Models

- Linear Regression
- Lasso
- Ridge
- Elastic Net
- Histogram Gradient Boosting Regressor

Classification Models

- Logistic Regression
- Support Vector Machine Classification
- K Nearest Neighbors
- Decision Tree Classifier
- Random Forest Classifier
- Naive Bayes
- Histogram Gradient Boosting Classifier

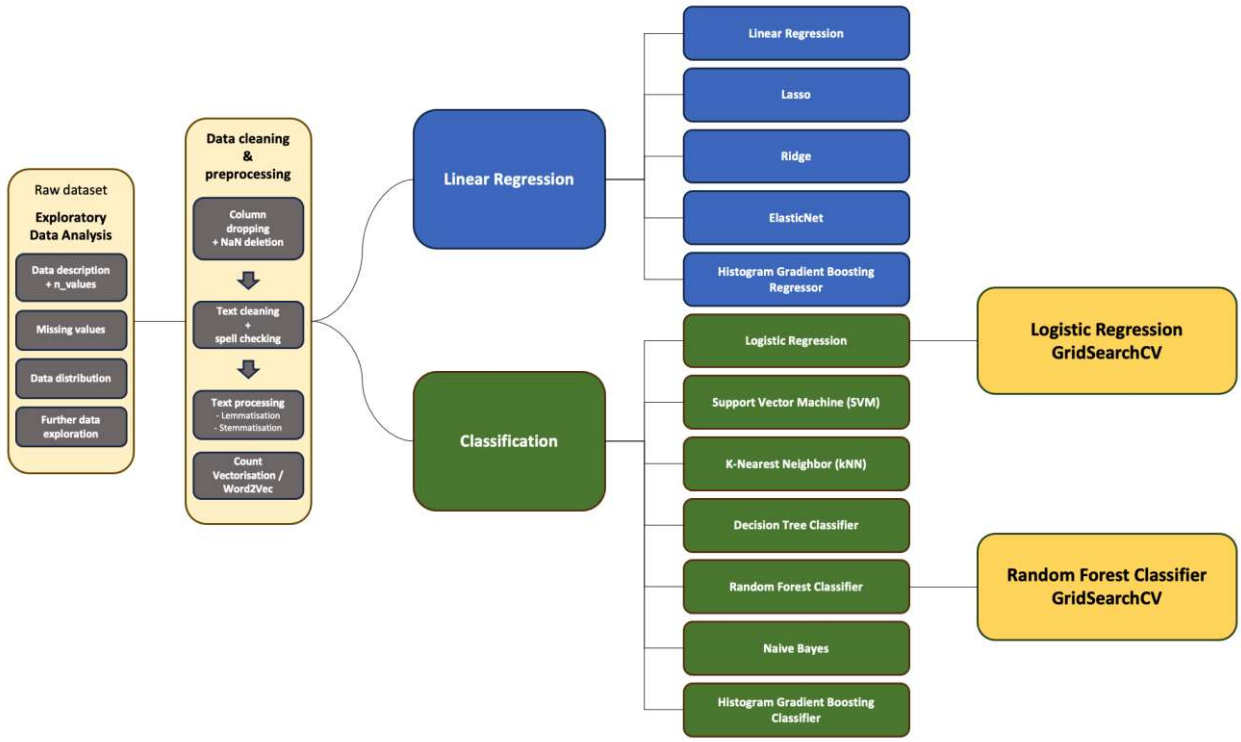
Method

The Classification and Regression methods are processed in parallel; the method is essentially identical in both processes.

In both methods an initial exploration is undertaken by running the various pre-processed data through each modelling technique. These results are compared to further models which are fed data subject to various types of sampling and assessed for improvements.

Following the initial exploration, the Classification models are sorted, and the best performing models are subjected to further investigation, by constructing a parameter grid to process in a GridSearchCV to uncover any potential model optimizations and maximize performance. This differs slightly from the regression models, which have the gridsearchCV included in the basic modelling methods and the best performing parameters were carried forward to more complex modelling methods.

Fig 1. Project Schema



Regression Modelling

The classification models return exact categories 1-5, the regression models will only return a continuous series of results. This has its benefits; the spread of the data can be analyzed in greater detail than the classification reports. The visibility of the data can be used to identify edge cases and see how noise in the categories behaves, which is not possible in the classification reports.

One of the drawbacks of the regression models is the need to split the results into categories before calculating performance. The cutting method is also analyzed as part of the Regression Analysis process.

Initially a pipeline concept was pursued, creating custom functions to run the various regression models, however this was abandoned when it became clear that the addition of sampling and cutting the data would limit the pipelines utility significantly. Instead, custom functions and a custom looping function to chain functions together were introduced. The pipeline origin of some of the custom functions is apparent where the function class is clearly designed for pipeline implementation.

The HistGradientBoostingRegressor (HGBR) and the HistGradientBoostingClassifier (HGBC) were chosen over the GradientBoostingRegressor (GBR) and the GradientBoostingClassifier (GBC) to reduce runtimes. Due to the histogram design of these models which is aimed at larger datasets typically consisting of over 10k features and the utilization of OpenMP for parallelization, much more CPU power can be contributed to the modelling and results in much faster results. Considering the 446-minute runtime for the main HGBR parameter test was considered fast, this can give some indication of the advantages of parallelization in comparison to the non-parallelized GBR model, which was manually terminated after 26 hours running on the Word2Vec dataset. It was suspected that with just 300 features the Word2Vec data may have been more accurate using the GBR, but the runtimes remained unreasonable despite the reduced features.

The Sampling Methods were separated out to maintain reasonable function runtimes. Regressors with multiple argument parameters to be tested were initially tested on what will be referred to as basic sampling methods. These include no sampling, RandomOverSampling and RandomUnderSampling, these are being called basic because they were implemented without any argument parameters being adjusted. The best regression parameters from the basic sampling were used in the more complex sampling methods to reduce runtimes again.

Sampler Argument Variables

SMOTE:

- “k_neighbors” argument values
  - 5
  - 50
  - 100
  - 250
  - 500
  - 1000

CentroidCluster:

- KMeans estimator.
    - "n_init" - "auto", allowing the sampler to find the number of initializations where the data converges.
    - "init" - "k-means++", setting the initialization points as widely as possible with an even distribution within the data variance.
    - “n_clusters”
        - 5
        - 25
        - 50
        - 250
        - 1000

Regression Model Argument Variables:

Lasso:

- Alpha values
    - 0.001
    - 0.01
    - 0.03
    - 0.05
    - 0.07
    - 0.09
    - 0.1
    - 0.2
    - 0.3

Ridge:

- Alpha values
    - 0.001
    - 0.01
    - 0.03
    - 0.05
    - 0.07
    - 0.1
    - 0.3

ElasticNet:

- Alpha values
  - 0.001
  - 0.01
  - 0.03
  - 0.05
  - 0.07
  - 0.1
  - 0.3
- L1 ratio
  - 0.3
  - 0.5
  - 0.7.

HGBR:

- Learning Rate
  - 0.1
  - 0.8.
- Max Depth
  - 50
  - 1000.

All Models were run through GridSearchCV modelling, which automatically split the data into train and test samples and took a record of the mean score for each configuration of parameters. As a guiding rule, cv was set to 5 creating 5 splits in the data and producing a reliable cross validation of the regression score. However, this cv was lowered in the face of extensive run times.

In the case of Word2Vec processed data, the best models were sourced from the previous investigations in the word stem vector methods and the optimal arguments were carried forward, assuming they would be equally effective. This Includes HGBR with 0.5 learning_rate and max_depth=1000 and the LinearRegression with default arguments. Data was scaled using the MinMaxScaler and resampled using the RandomOverSampler.

Classification Modelling

Classification modelling is more straight forward than the Regression modelling. Learning key lessons from the Regression modelling, less runs are completed to return a similar number of results.

The models are run with a reduced dataset which is approx. 50% of the total dataset size, it remains imbalanced.

Model parameters are fixed for the initial explorations, these are so:

- Logistic Regression
  - default arguments
- Support Vector Machine Classification
  - default arguments
- K-Nearest Neighbour
  - n_neighbors - 5
- Decision Tree Classifier
  - default arguments
- Random Forest Classifier
  - default arguments
- Multinomial Naïve Bayes
  - default arguments
- HGBC
  - input features are converted to dense matrices
  - default arguments

The word2vec dataset is scaled using the MinMaxScaler and resampled using the RandomOverSampler. The data is only processed using the most effective modelling techniques from the investigations above.

The selected models, Logistic Regression and Random Forest Classifier and were optimized using a grid search cross validation to scan across argument variables. The cv value was set to 5 to create 5 splits in the data and produce a reliable cross validation of the classification score. The cv was lowered in the face of extensive run times just as in the regression model grid searches.

Word2Vec preprocessing with Selected Classification Model Grid Search Arguments:

- Logistic Regression
  - C
    - 0.01
    - 1
    - 100
  - Penalty
    - L1
    - L2
    - elasticnet
  - L1 ratio - Only affects elastic net runs.
    - 0.3
    - 0.5

        - 0.7
- Random Forest Classifier
    - Max Depth
        - None
        - 10
    - Min_samples_leaf
        - 1
        - 2
    - Min_samples_split
        - 2
        - 5
    - n_estimators
        - 50
        - 100
        - 200
    - Criterion
        - Entropy
        - log_loss
    - Bootstrap
        - True
        - False

Although the SVM Model gave the best overall accuracy, it was dropped from future processing as its accuracy was comparable to both Random Forest and Logistic Regression which ran significantly faster.

The arguments which produced the best accuracies in the Word2Vec grid search cross validation were carried forward to future model analyses regardless of their accuracy compared to the original dataset. The assumption is taken forward that the datasets would behave similarly in the modelling conditions as only these model arguments are varied; however, this is open to scrutiny and should be tested. These tests were omitted from this investigation due to project time restraints.

Results

Regression Results

Best Regression Model Arguments

HGBR:

- Learning Rate – 0.5
- Max Depth - 1000

Ridge:

- Alpha – 0.3
- Sampler - SMOTE Sampler
    - K neighbors – 1000

Linear Regression:

- Sampler - SMOTE Sampler
    - K neighbors - 1000

Lasso:

- Alpha – 0.001

Elastic Net

- Alpha – 0.001
- L1 Ratio – 0.3
- Sampler - RandomOverSampler

Table 2. Best Regression Model Results - Ordered by best Test Accuracy

Model	Token Method	Vector Method	Sampler	Mean accuracy	Mean precision	Mean recall	Mean f1 score	Mean r2 score	Mean mse
HGBR	lemmatized Train Data	TFIDF	None	0.604	0.740	0.604	0.642	0.563	0.747
HGBR	lemmatized Test Data	TFIDF	None	0.593	0.728	0.593	0.632	0.509	0.829
Ridge Regression	lemmatized Train Data	TFIDF	Smote	0.523	0.750	0.523	0.562	0.892	0.700
Ridge Regression	lemmatized Test Data	TFIDF	Smote	0.382	0.643	0.382	0.435	0.510	1.393
Linear Regression	English Stemmer Train Data	TFIDF	Smote	0.697	0.802	0.697	0.724	0.838	0.465
Linear Regression	English Stemmer Test Data	TFIDF	Smote	0.380	0.512	0.380	0.423	0.162	2.384
Lasso Regression	lemmatized Train Data	Count Vector	None	0.294	0.659	0.294	0.322	0.488	1.470
Lasso Regression	lemmatized Test Data	Count Vector	None	0.280	0.628	0.280	0.303	0.444	1.580
ElasticNet	lemmatized Train Data	Count Vector	Random Over Sampler	0.303	0.680	0.303	0.331	0.507	1.415
ElasticNet	lemmatized Test Data	Count Vector	Random Over Sampler	0.269	0.615	0.269	0.300	0.423	1.641

Classification Results

Table 3. Best Classification Model Results - Ordered by best Test Accuracy

Model	Token Method	Vector Method	Sampler	Mean accuracy	Mean precision	Mean recall	Mean f1 score	Mean r2 score	Mean mse
SVM train data	lemmatized	TFIDF	None	0.874	0.882	0.814	0.856	0.762	0.403
SVM test data	lemmatized	TFIDF	None	0.770	0.725	0.77	0.709	0.410	0.996
Logistic Regression	Lemmatized test data	TFIDF	None	0.769	0.720	0.769	0.715	0.408	1.004
Logistic Regression	lemmatized Test Data	TFIDF	None	0.762	0.703	0.763	0.707	0.376	1.053
Random Forest	Lemmatized train data	Count Vector	None	0.971	0.971	0.971	0.970	0.955	0.076
Random Forest	Lemmatized test data	Count Vector	None	0.753	0.708	0.753	0.684	0.267	1.24
HGBC train data	English Stemmer	Count Vector	None	0.782	0.752	0.782	0.745	0.407	1.00
HGBC test data	English Stemmer	Count Vector	None	0.742	0.682	0.742	0.698	0.299	1.18
Decision Tree	Lemmatized train data	Count Vector	None	0.971	0.971	0.971	0.970	0.954	0.076
Decision Tree	Lemmatized test data	Count Vector	None	0.683	0.663	0.683	0.673	0.158	1.421
Naïve Bayes train data	English Stemmer	Count Vector	Random Under Sampler	0.523	0.517	0.523	0.514	0.311	1.378
Naïve Bayes test data	English Stemmer	Count Vector	Random Under Sampler	0.624	0.711	0.624	0.659	0.111	1.501
KNN Train data	English Stemmer	Count Vector	Random Under Sampler	0.556	0.556	0.556	0.552	0.232	1.537
KNN test data	English Stemmer	Count Vector	Random Under Sampler	0.484	0.660	0.484	0.541	-0.214	2.050

Google Word2Vec Preprocessing

The main major advantage of this text process is that 300 features are processed much faster than a sparse matrix or dense matrix containing between 10,0000 and 70,000 features, depending on the contributing processes and the number of records used to train the dataset.

Due to time constraints in the project, only two models with low processing times were modelled with the word2vec text process, and no word stemming was used. These models were the classifiers Logistic Regression and Random Forest.

The grid search cross validation was performed on this dataset to minimize runtimes overall and the results of these tests were used in all future modelling regardless of the final accuracy compared to previous datasets.

Best Classification Model Arguments

Logistic Regression Parameters:

- solver – saga
- C - 100
- L1 ratio - 0.3
- penalty - elasticnet

Random Forest Parameters:

- max_depth - None
- min_samples_leaf – 1
- min_samples_split – 2
- n_estimators – 200

Naive Bayes:

- Sampler – RandomOverSampler

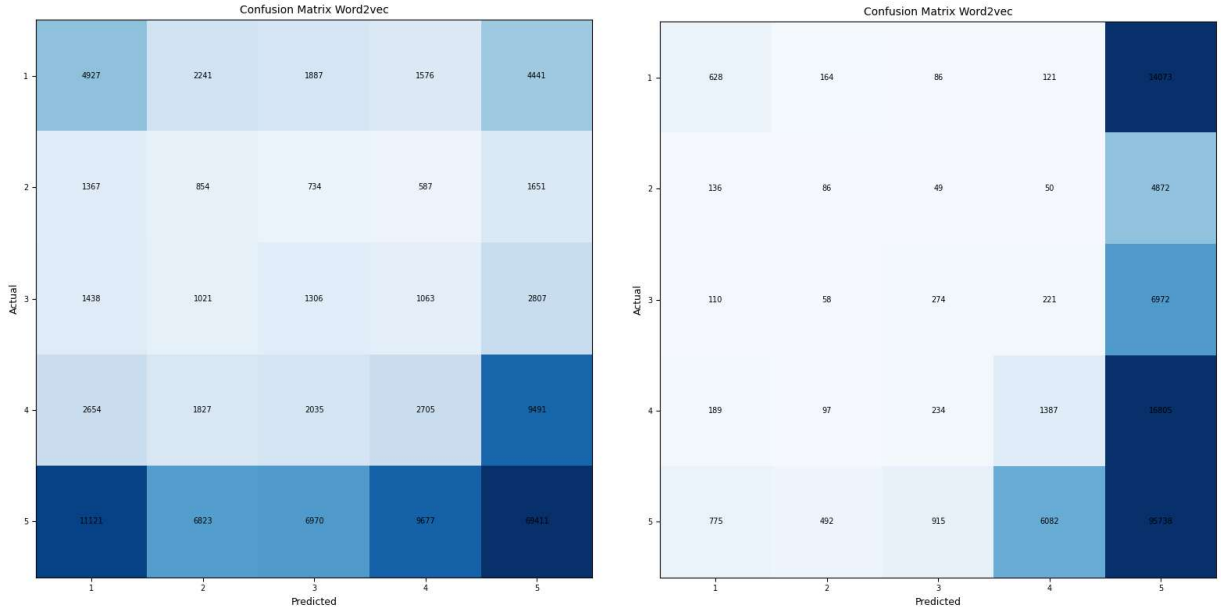
K Nearest Neighbors:

- Sampler – RandomOverSampler

Table 4. Google Word2Vec Grid Search CV Model Comparisons

Model	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy	Mean Squared Error
Logistic Regression	Google Word2Vec	0.526	0.60	0.53	0.70	0.56	0.57	0.34	-
Random Forest	Google Word2Vec	0.651	0.55	0.65	0.36	0.57	0.26	0.07	2.31

Fig 5. Google Word2Vec Confusion Matrices



Like the regression modelling, the score has worsened in comparison to the target baseline accuracy of 69%.

This is most likely due to the mismatch of the search engine preprocessing model and the sentiment analysis modelling that is being performed. It is possible to customize the Word 2 Vec process and this will likely prove to be a good starting point for further analysis.

Regression Cutting analysis

Regression Cutting

Difficulties occurred when comparing the Regression model predictions to the original classes in the target variable. The error between the datasets was large as the prediction returned a continuous series which extended beyond 0-5. To allow a fair comparison between the datasets a series cut function was used, separating the continuous dataset into groups:

- 1: all values less than 1.5
- 2: all values between 1.5 and 2.5
- 3: all values between 2.5 and 3.5
- 4: all values between 3.5 and 4.5
- 5: all values greater than 4.5

Since it is feasible that that this could introduce error, the individual series were analyzed to check for obvious errors. For example, if the data in each classification does not grossly overlap and lots of points reside in categories 1 and 5, then it could be advantageous to scale the results before cutting to compress the results into the correct classifications.

Model	Token Method	Vector Method	Sampler	Mean accuracy	Mean precision	Mean recall	Mean f1 score	Mean r2 score	Mean mse
HGBR	Lemmatized train data	TFIDF	None	0.604	0.740	0.604	0.642	0.563	0.747
HGBR	Lemmatized test data	TFIDF	None	0.593	0.728	0.593	0.632	0.509	0.829

Table 6. Best model Performance

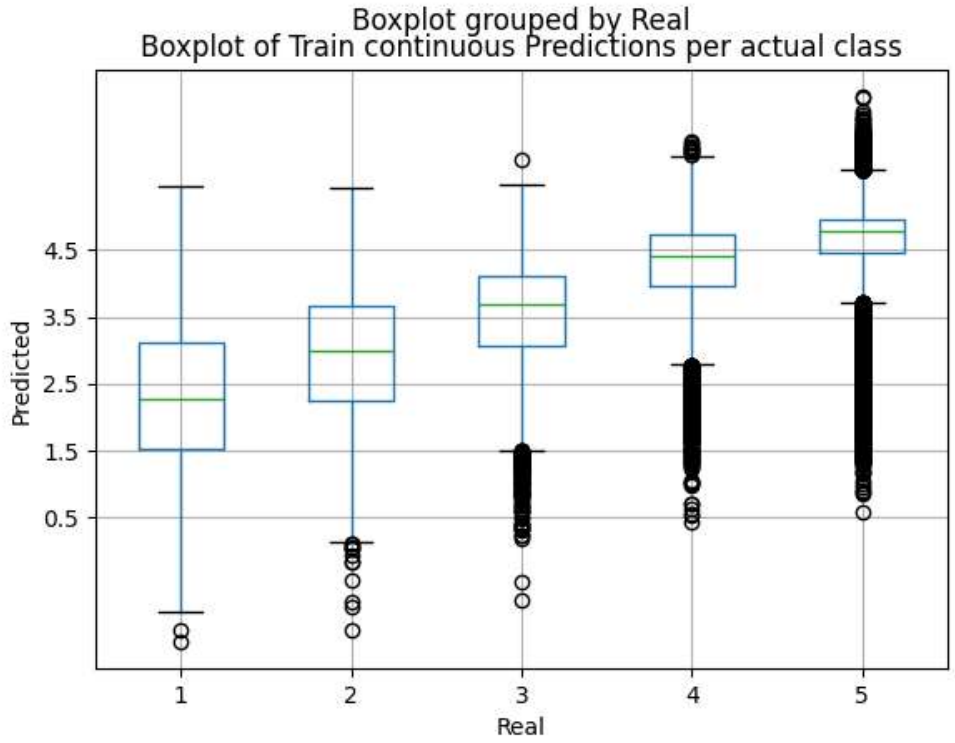


Fig 7. Boxplot between target variable classes and continuous prediction Train Data (half).

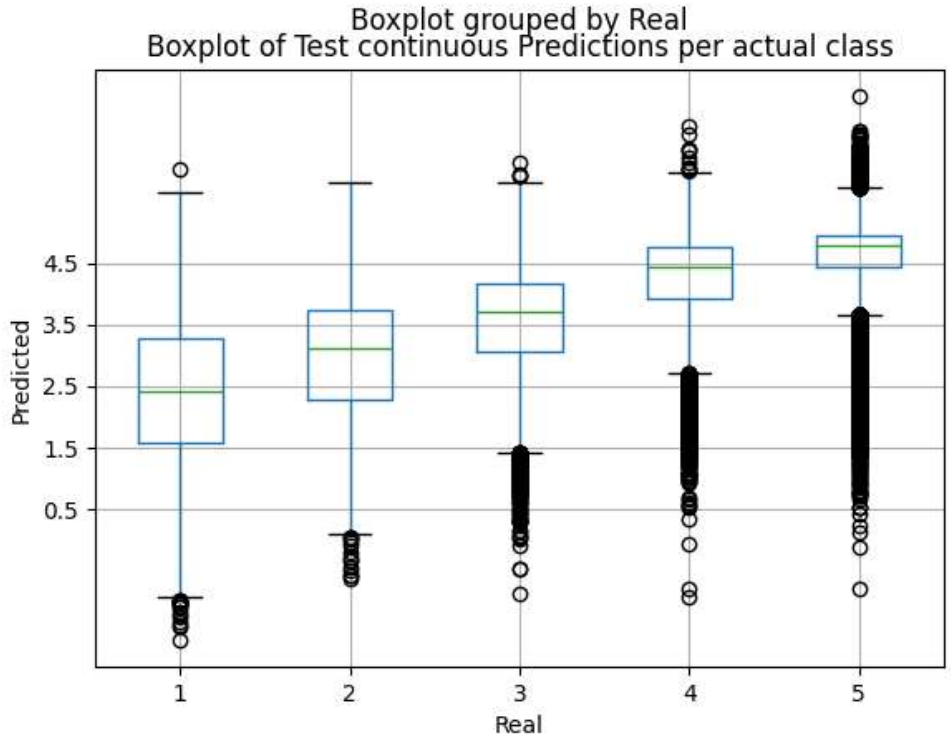


Fig 8. Boxplot between target variable classes and continuous prediction Train Data (half).

These graphs were generated from the best HGBR prediction. The graphs make it clear that the linear regression generated by the HGBR is battling a lot of predictive overlaps between classes. The range of each box is as long as the full spread of the data and the Inter Quartile Range also spans 3 classes even on the smallest box. The gentle linear trend is visible by observing the distributions, but the spread of the distributions explains the low R squared and coefficient scores.

Table 9. Classification Box and Whisker Values Train Data

Real Class	mean	Standard Deviation	Min value	Lower Quartile	Median	Upper Quartile	Max Value
1	2.319	1.072	-1.343	1.519	2.281	3.131	5.450
2	2.935	0.934	-1.185	2.254	3.004	3.665	5.419
3	3.557	0.809	-0.723	3.072	3.681	4.121	5.850
4	4.288	0.637	0.425	3.958	4.415	4.742	6.128
5	4.668	0.486	0.590	4.451	4.783	4.949	6.786

Table 10. Classification Box and Whisker Values Train Data

Real Class	mean	Standard Deviation	Min value	Lower Quartile	Median	Upper Quartile	Max Value
1	2.406	1.133	-1.610	1.587	2.414	3.281	6.031
2	2.983	0.989	-0.622	2.280	3.103	3.745	5.802
3	3.559	0.874	-0.856	3.070	3.705	4.168	6.125
4	4.272	0.690	-0.915	3.932	4.421	4.753	6.720
5	4.652	0.525	-0.799	4.431	4.779	4.949	7.195

- **Real Class 1**: Cut min value to 1.5
  The mean values of both Train and Test Data are above 2. The lower quartile range is close to 1.5 for both, indicating that only 25% of this class will be cut into the correct class. The interquartile range overlaps with the inter quartile range in class 2, where the class 1 mean is close to the class 2 lower quartile and the class 1 upper quartile is close to the class 2 mean.
- **Real Class 2**: Cut 1.5 to 2.5
  In both datasets the inter quartile range overlaps with the cutting thresholds, slightly more than 25% of this data will be correctly classified. The interquartile range overlaps with both class 1 and class 3 inter quartile ranges. The edges of the interquartile range are close to the mean value of the next class, showing large overlaps.
- **Real Class 3**: Cut 2.5 to 3.5
  The mean values of the datasets lie above the edge of the cutting thresholds. The lower quartiles lie within the threshold and the lower range extends to 1.5 before being considered extreme values. Between 25% and 50% of the data is correctly classified. There is less overlap between Class 3 and 4 than the previous classes.

- **Real Class 4**: Cut 3.5 to 4.5
  Both mean Values lie around 4.2, which is inside the upper edge of the cutting threshold. The lower quartiles are also within the cutting threshold and close to the center of the thresholds. The ranges extend down below 3 but are considered extreme above 2.5. between 35% and 50% of the data is correctly classified, and since the dataset distribution is tighter, more of this class is correctly classified.
- Real Class 5: Cut 4.5 to max value
  The mean values of both Datasets are close to 4.6, just above the lower bound of the cuttting thresholds. Over 50% of the data in this dataset will be correctly classified. This class is the most tightly distributed but overlaps with class 4 more than class 3 overlaps with class 4.

Key Observations:

- All data is skewed upwards, excepting class 5.
- The mean points of each distribution form a straight line with a gentler gradient than the real data.
- Class distributions tighten as class increases, all have very wide ranges.

Hypothesis

It is conceivable that the noise in the lesser represented classes pushes the distribution upwards, which skews the mean point of each class upwards towards the most represented datapoint.

It is possible that accuracy of the regression model could be improved by expanding the dataset around the mean value. This could be accomplished by translating each datapoint to the origin, by subtracting the mean values of the predicted and real data from each point.

The average conversion between the predicted values and the actual values can then be derived and every data point in the dataset can be multiplied by the average conversion. The datapoints can be restored to their respective classes by translating the mean values to each new value accuracy. This distribution can be reinspected and will highlight an improvement to the cutting method.

If a score improvement is observed, then the usefulness of this translation can be analyzed by performing the same translation on the test data, using the values from the training data.

Table 11. HGBR Model train expansion conversion values

Predicted value mean	4.269
Real Class value mean	4.270
Class 1 Real mean/Predicted mean	1.677
Class 2 Real mean/Predicted mean	1.702
Class 3 Real mean/Predicted mean	1.782
Class 4 Real mean/Predicted mean	14.213 - ANOMOLY
Class 5 Real mean/Predicted mean	1.831
Average magnitude	1.748

To apply this translation to the dataset, each prediction has the Predicted value mean subtracted, is multiplied by the Average magnitude and has the Predicted value mean added back on. The cutter is then applied to measure accuracy and other metrics as before.

Table 12. Translated model prediction results

Dataset	Derivation	Mean accuracy	Mean precision	Mean recall	Mean f1 score	Mean r2 score	Mean mse
HGBR	Lemmatized, TFIDF, No Sampler, train data	0.604	0.740	0.604	0.642	0.563	0.747
HGBR	Lemmatized, TFIDF, No Sampler, test data	0.593	0.728	0.593	0.632	0.509	0.829
Modified train	y = (y-4.269) * 1.748 + 4.269	0.652	0.716	0.652	0.680	0.543	0.781
Modified test	y = (y-4.269) * 1.748 + 4.269	0.639	0.706	0.639	0.668	0.468	0.897
Train	Detected improvement	0.048	- 0.024	0.048	0.038	-0.020	0.034
Test	Detected improvement	0.046	- 0.022	0.046	0.036	-0.041	0.068

Fig 13. Translated Train Data Boxplot

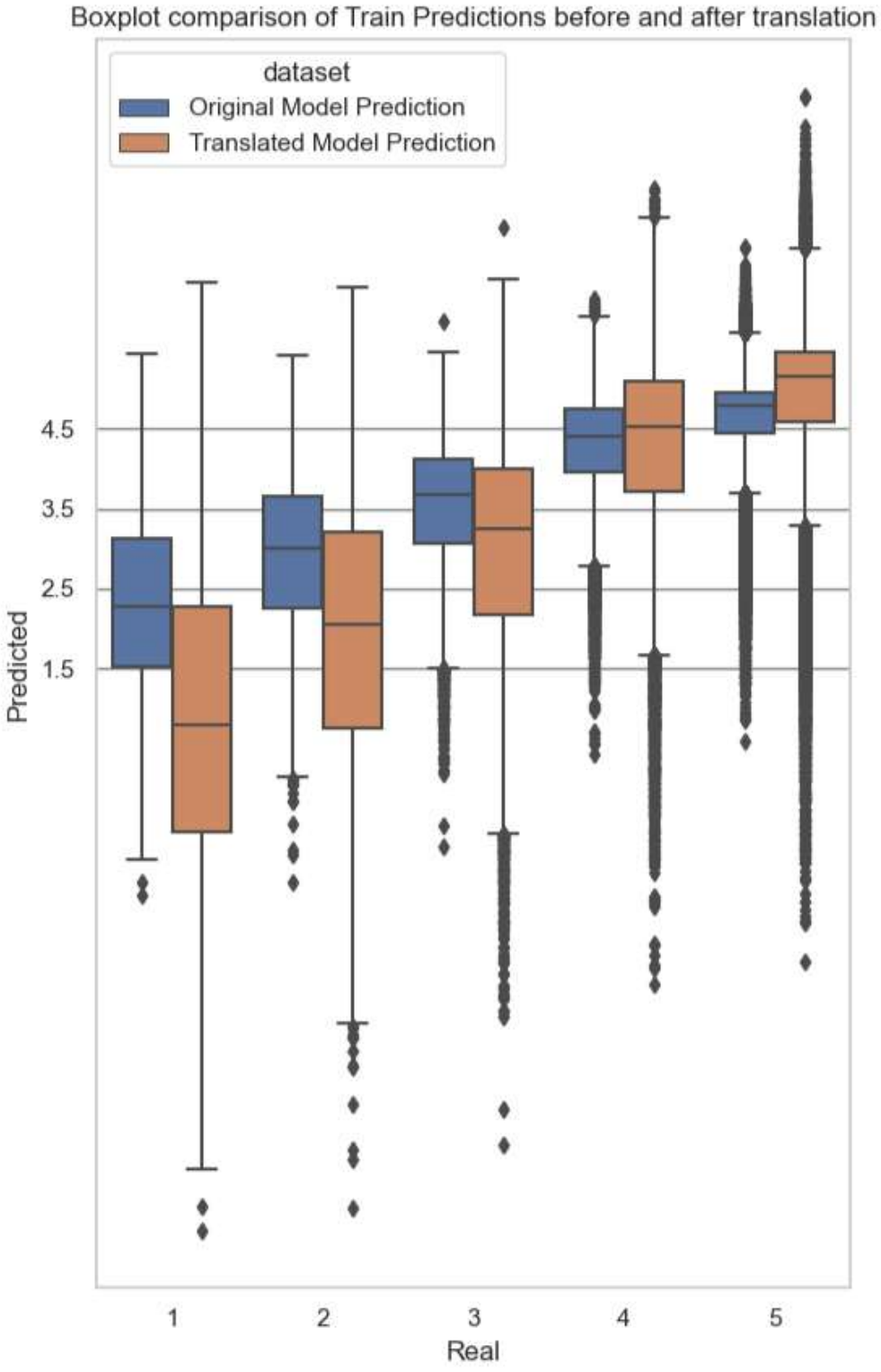


Fig 14. Translated Test Data Boxplot

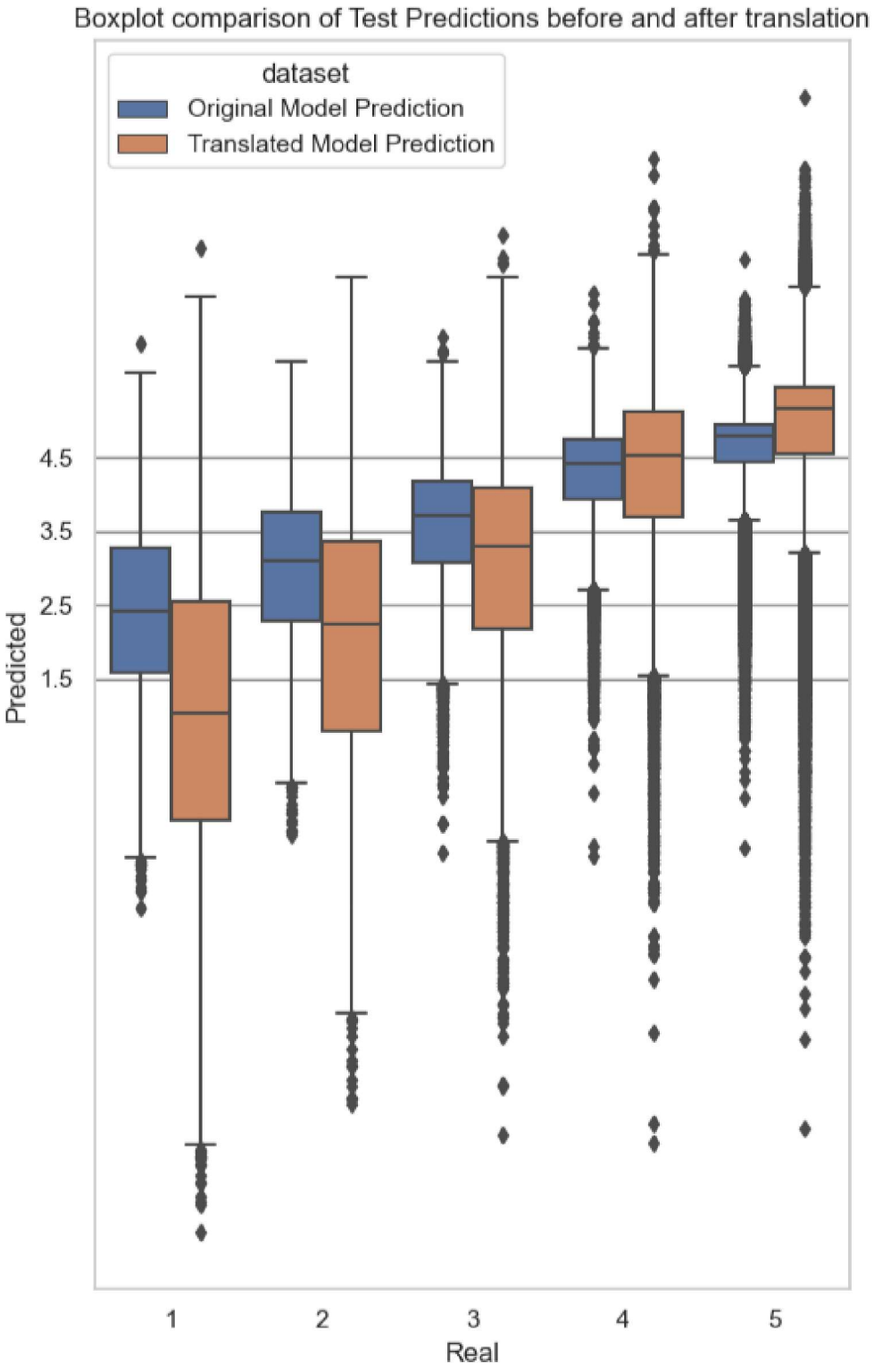


Fig 15. Best HGBR Confusion Matrices

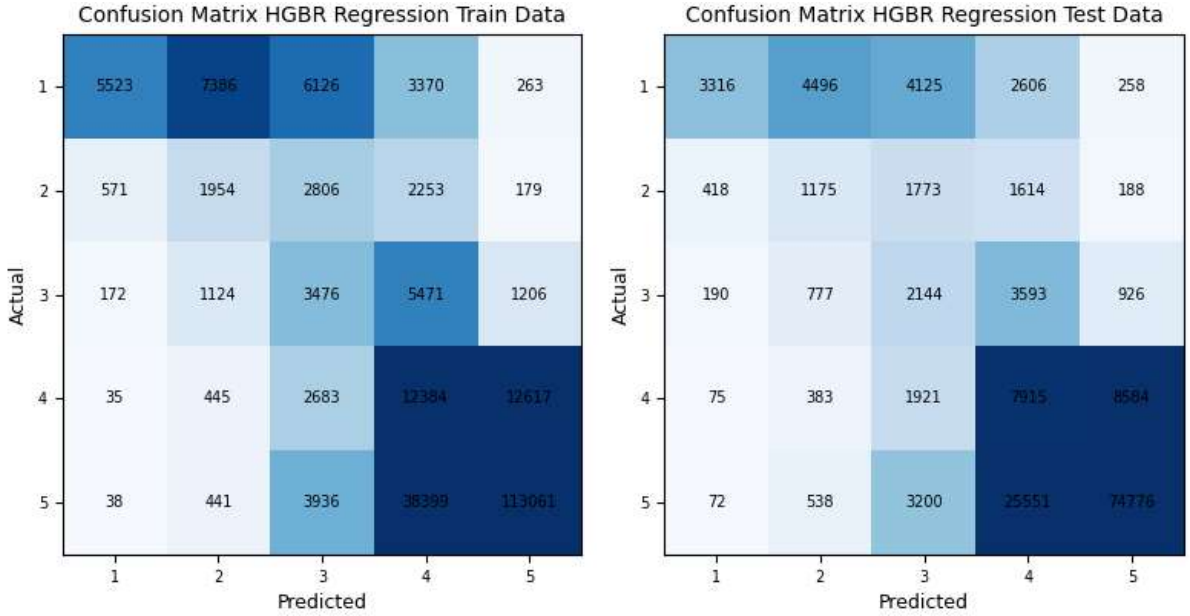
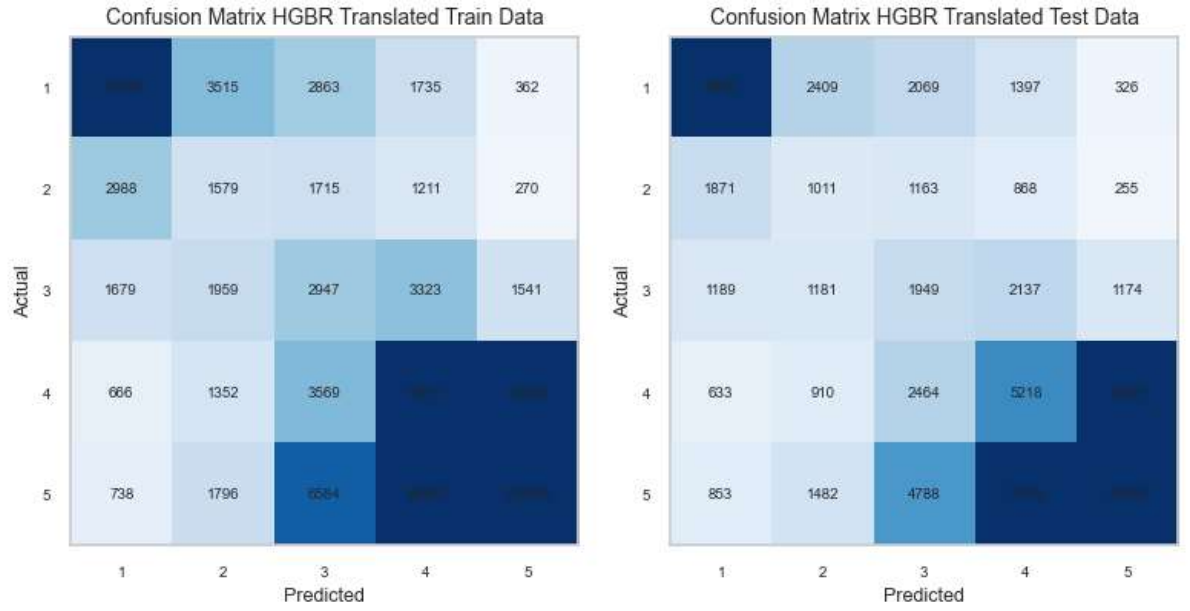


Fig 16. Translated HGBR Confusion Matrices



Cut Method Conclusion

Graphically, the new alignment of means is much better however the expansion of the predicted values has further increased the distributions of the classes, which ultimately means that more of the predicted values in class 1 and 5 are accurately classified, but less of all the other classes. Averaging about the same overall accuracy.

Translating the data in this way does not improve the overall accuracy as the modification does not affect the overlapping relationships between classes, which is also true of other modifications that could be made before cutting the data or cutting with a softmax method.

The box-plot analysis of the regression models does make it much easier to see how the model is performing compared to the Classification models where only a confusion matrix can be used for a graphical representation of the performance.

Overall Results Comparisons

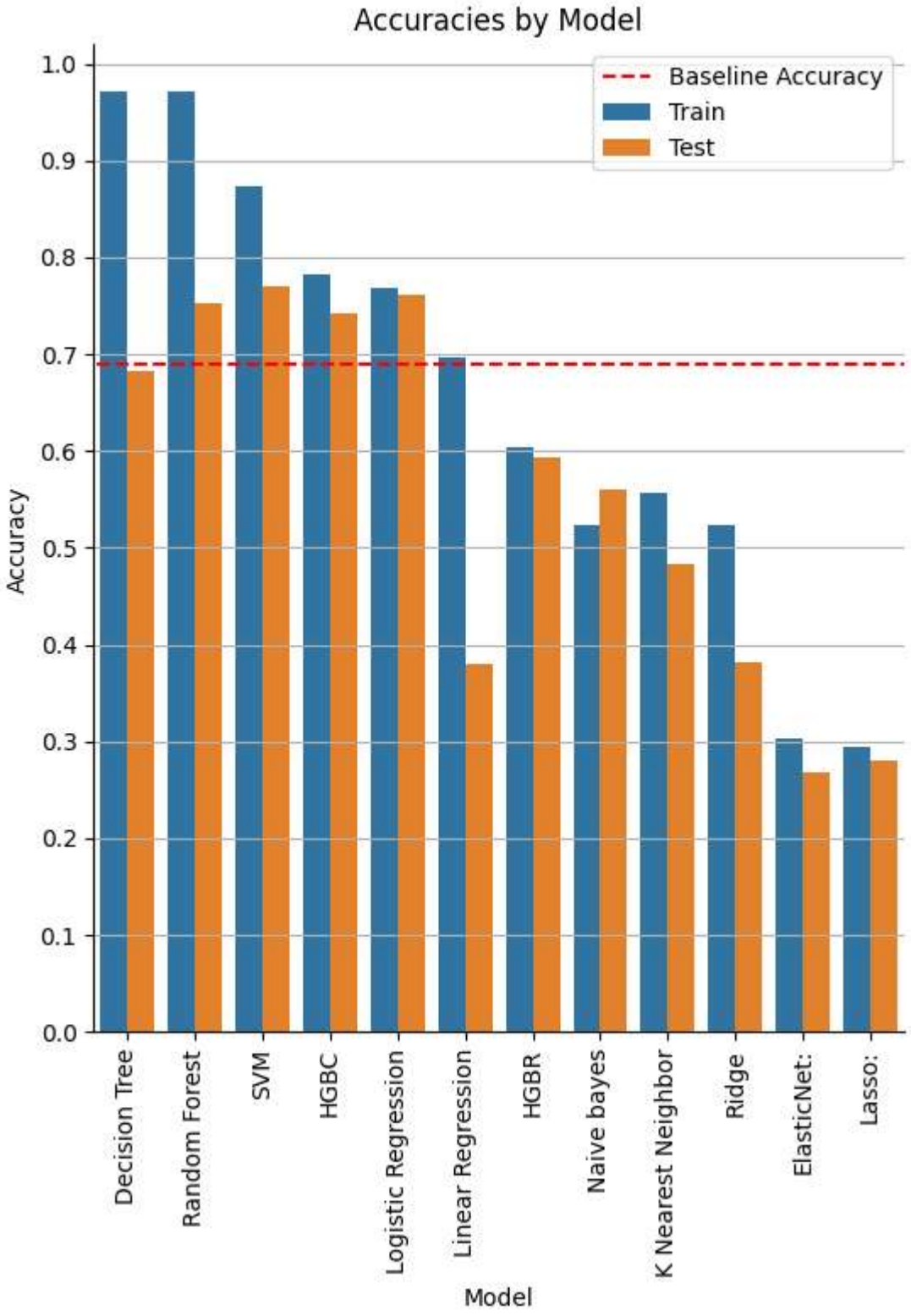
Only classification models improved upon the baseline score of 69%. The best performance was delivered by the SVM classification which achieved 77% accuracy on the test data. Logistic Regression and Random Forest also proffered scores between 75% and 77%. Histogram Gradient Boosting Classifier submitted a slightly lower score of 74%. These were the only models that improved upon the baseline 69% accuracy, though Decision Tree closely matched the 69% accuracy. K nearest Neighbor classifier was the worst performing classification model and Naive Bayes posted a unique result as the only model that produced a better test accuracy than train accuracy.

The regression models failed to post scores close to or above the baseline 69% accuracy. Across the board the regression models were less accurate than a basic model, the best performer was the Histogram Gradient Boosting Regressor, which posted a test accuracy score of 59%. Linear Regression managed to match the Baseline Accuracy in its train data but had the largest discrepancy between training data and test data, indicating the most overfitting.

Many models show indications of overfitting, with higher train scores posted compared to test scores. The classification model Logistic Regression shows the least amount of variance between the train and test scores, indicating that the model has found its best accuracy with the dataset in its current form.

An 8% increase on the baseline accuracy is not enough to inspire confidence in the outlined business cases, so more investigation may be necessary to better predict the data.

Fig 17. Bar chart displaying all results in order of best accuracies (train or test)



On Sampling

The performance of the sampled data provides extra indication that the data may need extra preprocessing to enable better performance. Random Over Sampler was the only sampler which performed marginally better, but the improvement was not significant enough to warrant the extra computation time for the extra data created. No sampling often returned the best accuracies and nominal mean squared errors which indicates that the features of the data may not have accurately represented the target value. It is likely that the important features or relationships between features were clouded by unimportant variables, so when sampling was performed to balance the dataset, the expected improvement was diminished. This is most likely because the unimportant features are being duplicated by over sampling and clouding the important features or because the important features are being overlooked and removed during under sampling.

Appendix A – Full Regression Results

Best Linear Regression Results

Table A.1 - LinearRegression RandomOverSampler, RandomUnderSampler and No sampling

tokenizer	vectorizer	sampler	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean Test mse
lemmatized	TFIDF Vectorized	rOs	0.563	0.563	0.563	0.563	-0.274	2.988
stemmatized	TFIDF Vectorized	rOs	0.525	0.525	0.525	0.525	0.086	2.020

Table A.2 - LinearRegression Smote Sampler – K Neighbors Comparison

tokenizer	vectorizer	sampler	Smote k neighbors	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
lemmatized	TFIDF Vectorized	smote	500	0.566	0.566	0.566	0.566	-0.26	2.96
stemmatized	TFIDF Vectorized	smote	1000	0.535	0.535	0.535	0.535	0.095	1.98

Fig A.3 - LinearRegression Smote K Neighbor Comparison

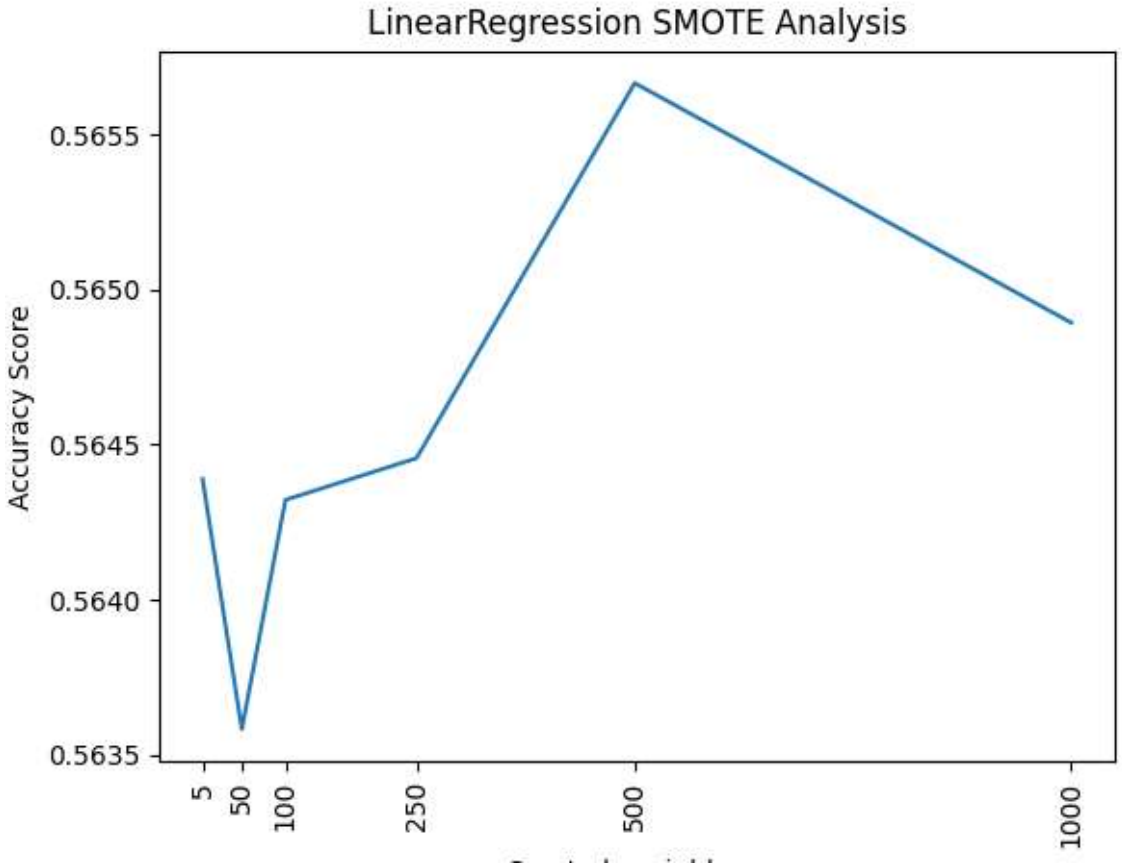


Table A.4 - LinearRegression ClusterCentroids – cc clusters comparison

tokenizer	vectorizer	sampler	Cc clusters	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
stemmatized	Count Vectorized	cc	5	0.299	0.299	0.299	0.299	-9.023	4.600

Fig. A.5 - Linear Regression ClusterCentroids cc cluster comparison

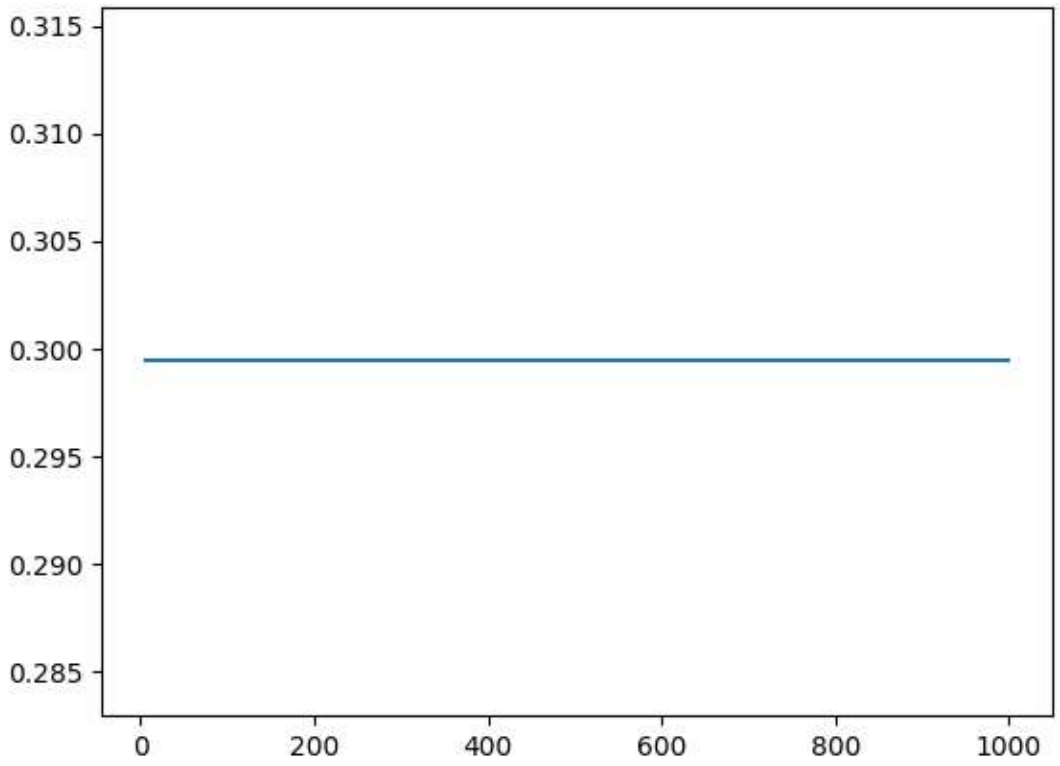


Table A.6 - LinearRegression Best Model Train and Test Results

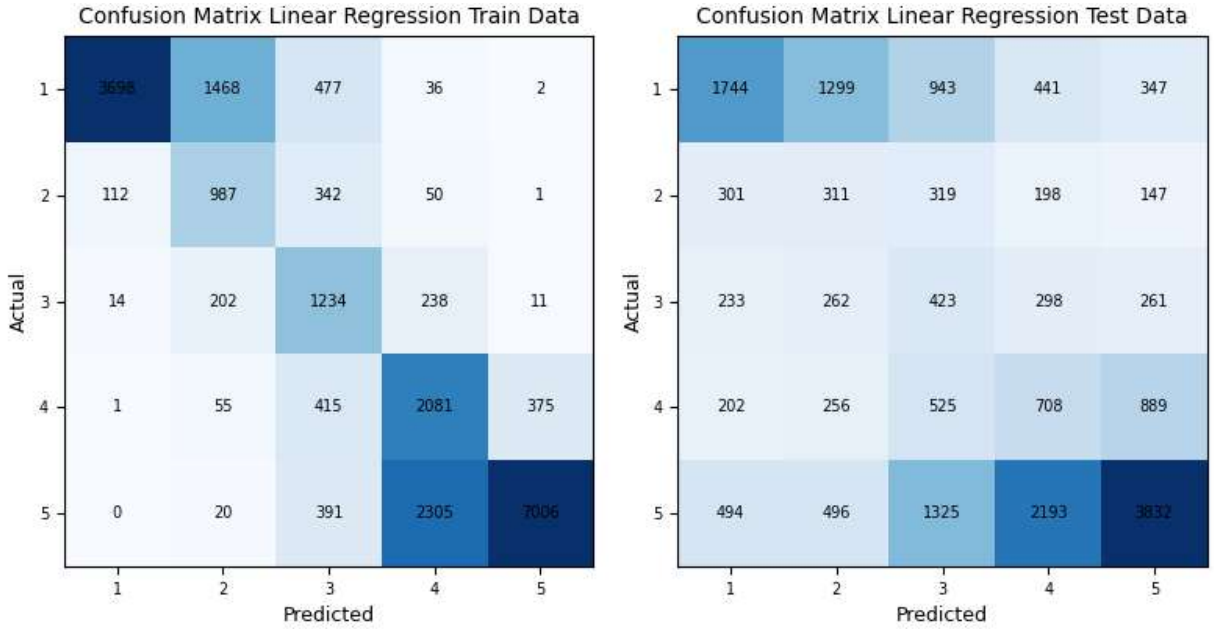
Train

tokenizer	vectorizer	sampler	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
stemmatized	TFIDF Vectorized	Smote - k neighbors: 1000	0.697	0.802	0.697	0.724	0.838	0.465

Test

tokenizer	vectorizer	sampler	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
stemmatized	TFIDF Vectorized	Smote - k neighbors: 1000	0.380	0.512	0.380	0.423	0.162	2.38

Fig. A.7 - Linear Regression Best Results Confusion Matrices



Best Lasso L1 Regression Results

Table A.8 - Lasso RandomOverSampler, RandomUnderSampler and No sampling

tokenizer	vectorizer	sampler	Param lasso alpha	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	None	0.001	0.284	0.284	0.284	0.284	0.442	1.602
lemmatized	Count Vectorized	rOs	0.001	0.271	0.271	0.271	0.271	0.318	1.50

Fig A.9 - Lasso Alpha Comparison

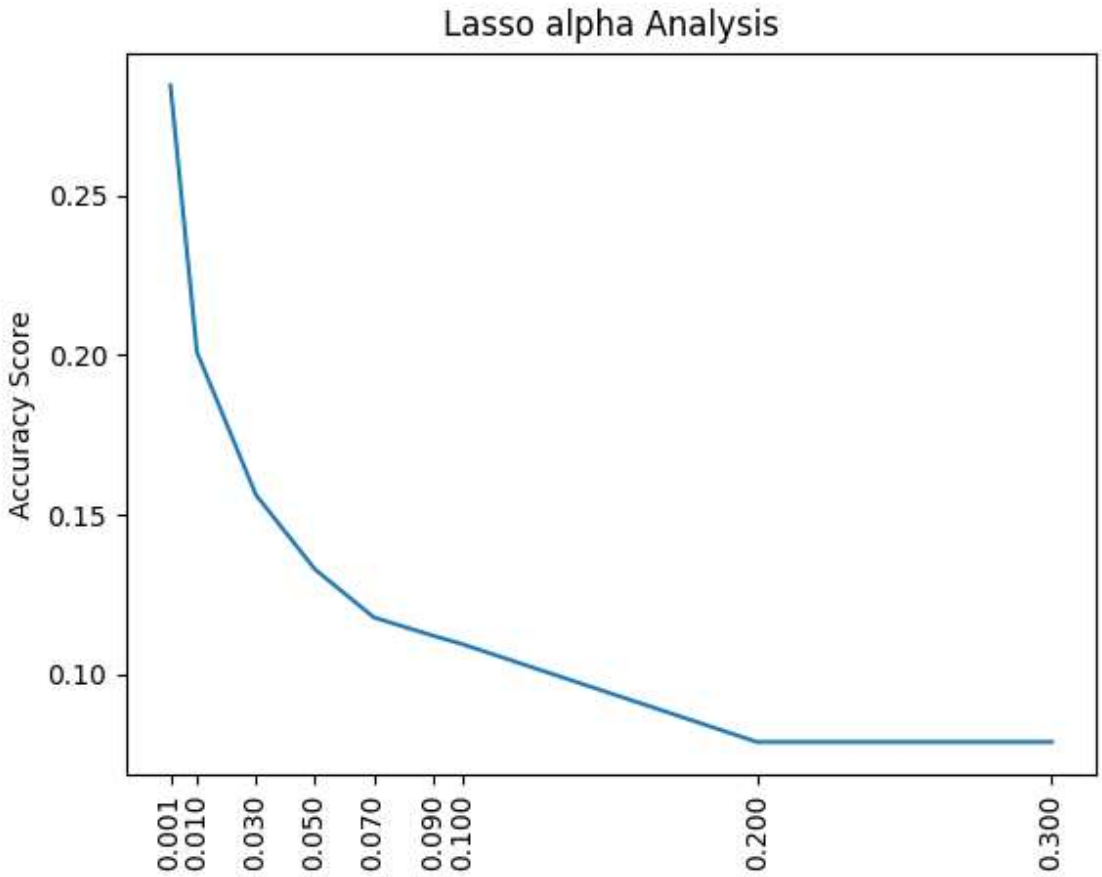


Table A.10 - Lasso Smote Sampler – K Neighbors Comparison

tokenizer	vectorizer	sampler	Smote k neighbors	Param lasso alpha	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
stemmatized	TFIDF Vectorized	smote	500	0.001	0.224	0.224	0.224	0.224	0.284	1.61
lemmatized	Count Vectorized	smote	1000	0.001	0.211	0.211	0.211	0.211	0.305	1.54

Fig A.11 - Lasso Smote K Neighbor Comparison

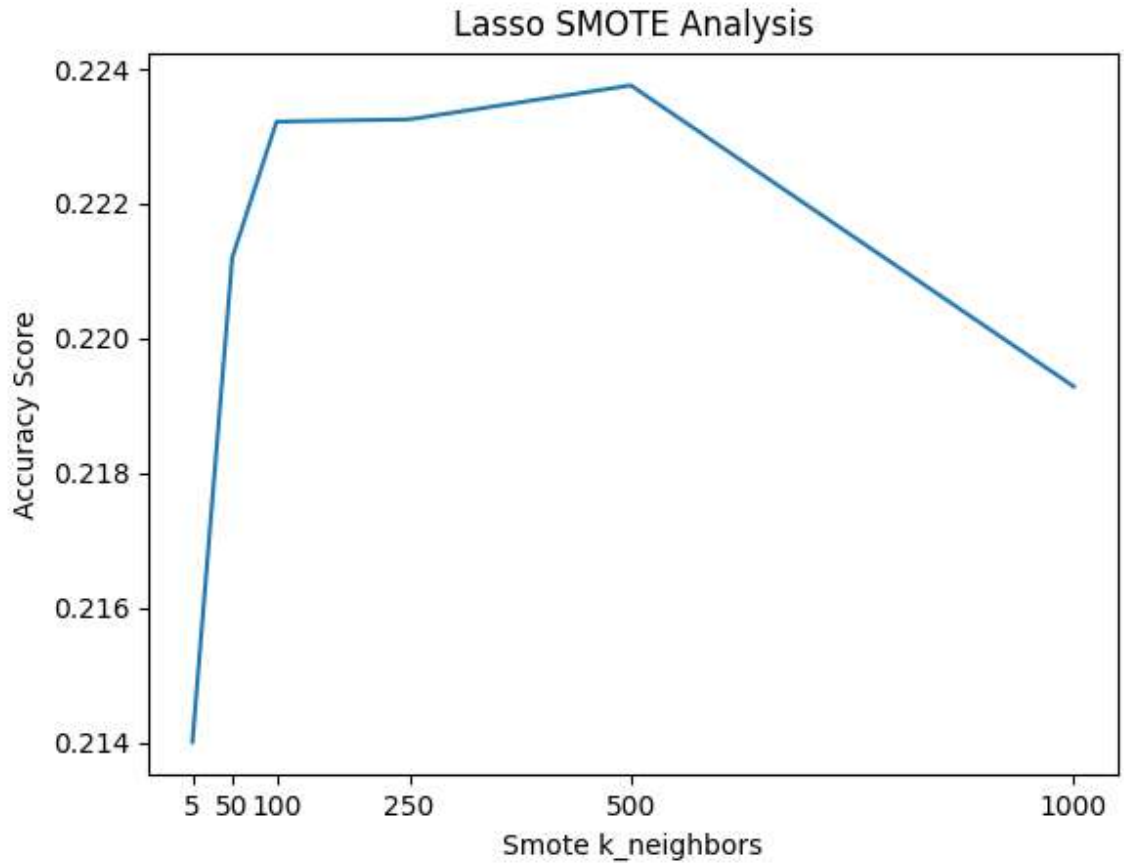


Table A.12 - Lasso ClusterCentroids – cc clusters comparison

tokenizer	vectorizer	sampler	Cc clusters	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	cc	5	0.214	0.214	0.214	0.214	-4.578	1.97

Fig. A.13 - Lasso ClusterCentroids cc cluster comparison

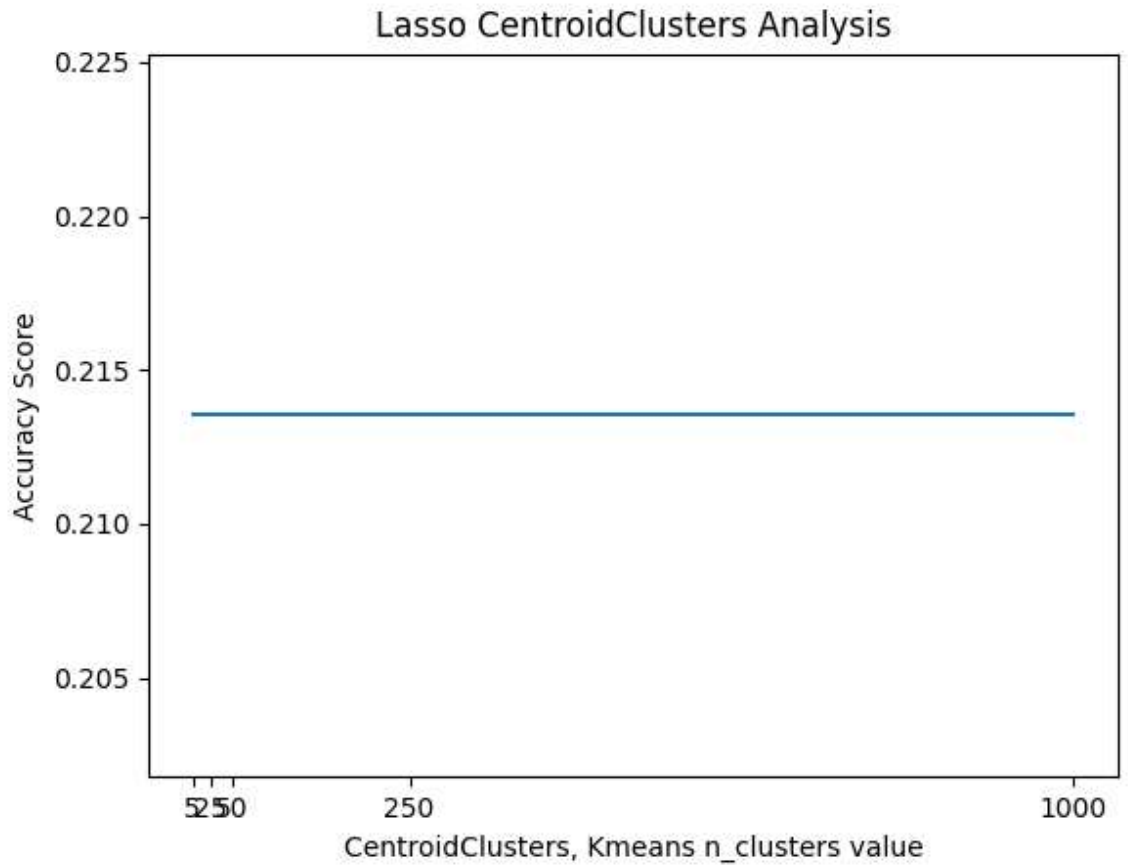


Table A.14 - Lasso Best Model Train and Test Results

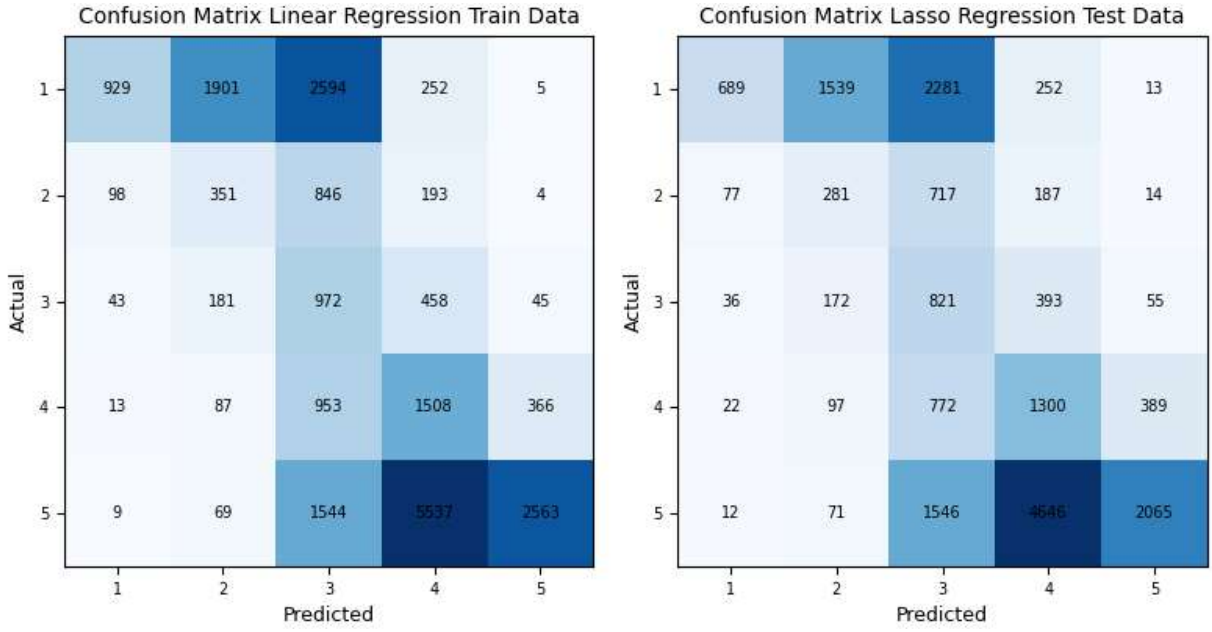
Train

tokenizer	vectorizer	sampler	Param lasso alpha	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	None	0.001	0.294	0.659	0.294	0.322	0.488	1.470

Test

tokenizer	vectorizer	sampler	Param lasso alpha	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	None	0.001	0.280	0.628	0.280	0.303	0.444	1.580

Fig. A.15 - Lasso Best Results Confusion Matrices



Best Ridge L2 Regression Results

Table. A.16 - Ridge RandomOverSampler, RandomUnderSampler and No sampling

tokenizer	vectorizer	sampler	Param ridge alpha	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	rOs	0.001	0.557	0.557	0.557	0.557	-0.25	2.915
lemmatized	TFIDF Vectorized	None	0.3	0.379	0.379	0.379	0.379	0.512	1.402
lemmatized	TFIDF Vectorized	rOs	0.3	0.478	0.478	0.478	0.478	0.427	1.11

Fig A.17 Ridge Alpha Comparison

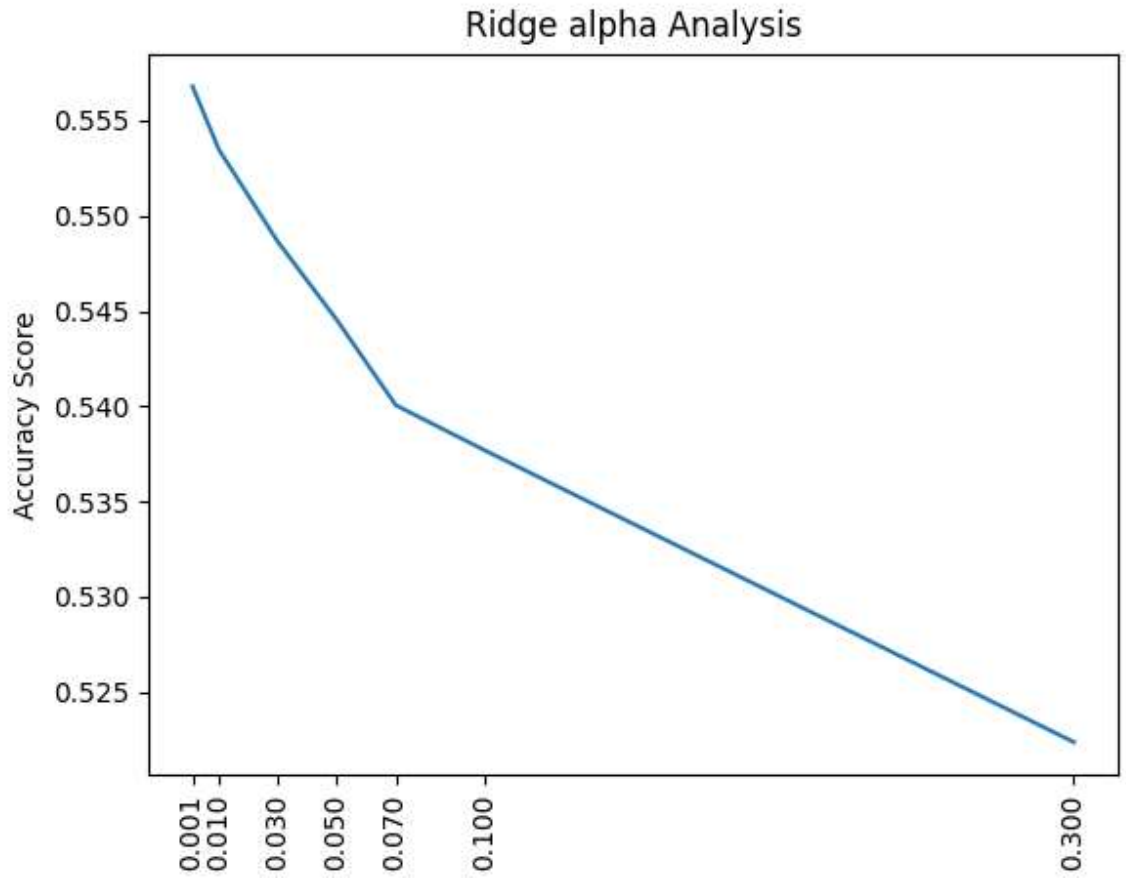


Table. A.17 - Ridge Smote Sampler – K Neighbors Comparison

tokenizer	vectorizer	sampler	Smote k neighbors	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
lemmatized	TFIDF Vectorized	smote	1000	0.490	0.490	0.490	0.490	0.43	1.081

Fig A.18 - Ridge Smote K Neighbor Comparison

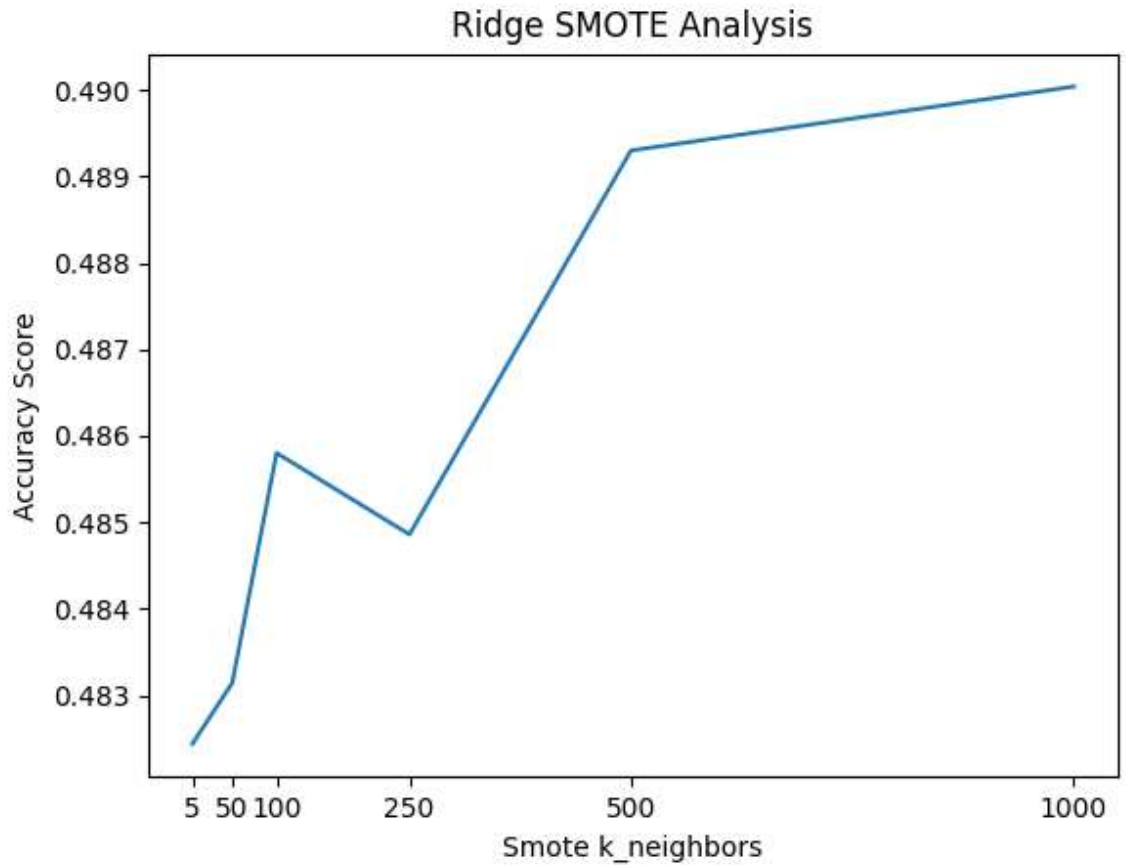


Table. A.19 - Ridge ClusterCentroids – cc clusters comparison

tokenizer	vectorizer	sampler	Cc clusters	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	cc	5	0.312	0.312	0.312	0.312	-5.460	2.353
stemmatized	TFIDF Vectorized	cc	5	0.295	0.295	0.295	0.295	-4.07	1.841

Fig. A.20 - Ridge ClusterCentroids cc cluster comparison

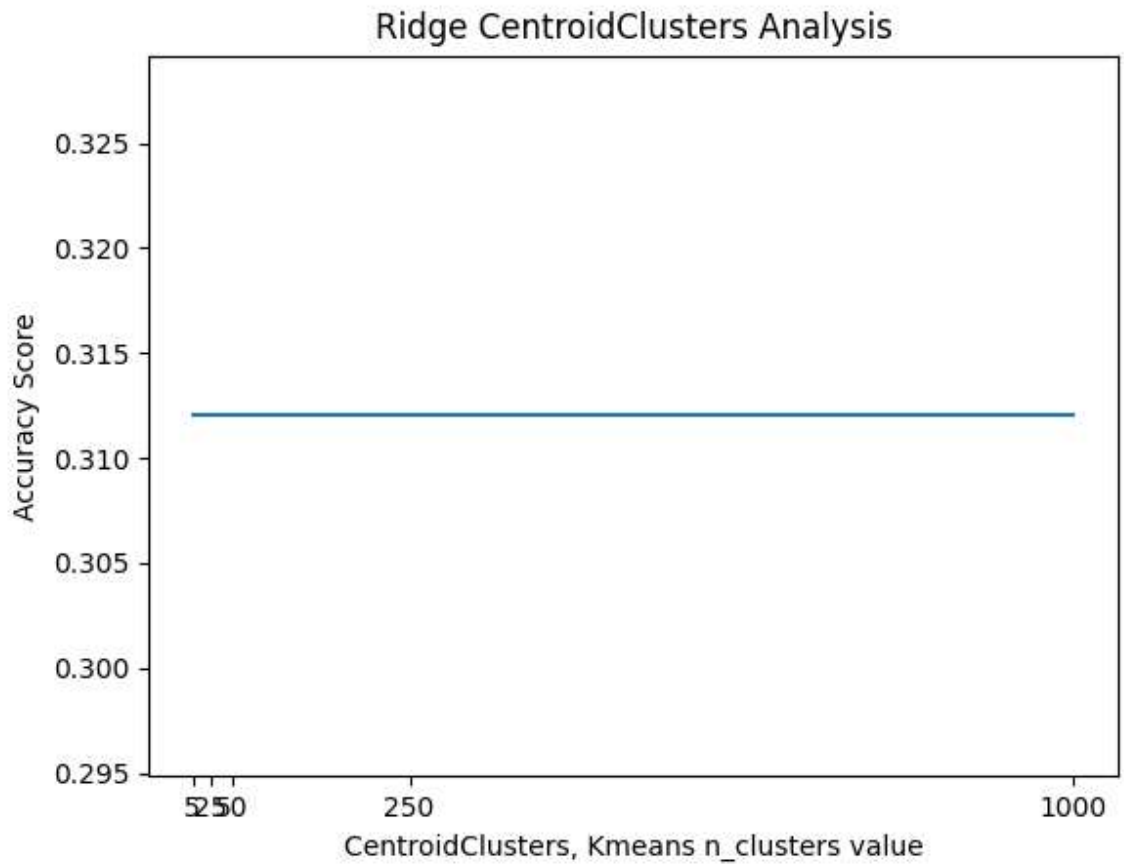


Table. A.21 - Ridge Best Model Train and Test Results

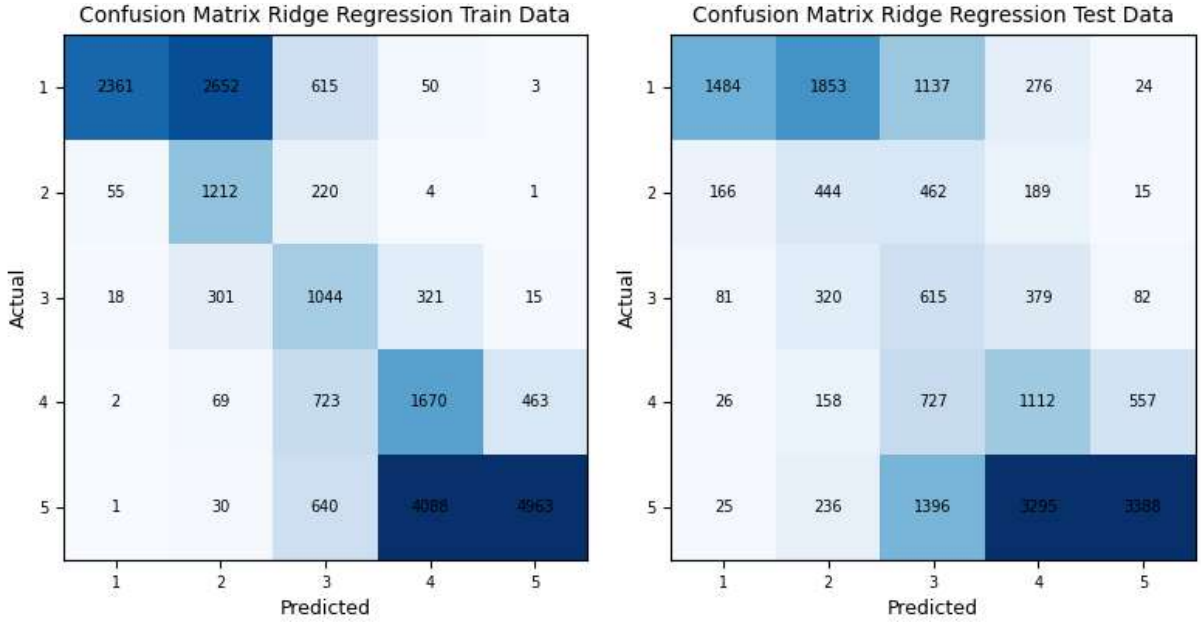
Train

tokenizer	vectorizer	sampler	alpha	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
lemmatized	TFIDF Vectorized	Smote, k_neighbors = 1000	0.3	0.523	0.750	0.523	0.562	0.892	0.700

Test

tokenizer	vectorizer	sampler	alpha	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
lemmatized	TFIDF Vectorized	Smote, k_neighbors = 1000	0.3	0.382	0.643	0.382	0.435	0.510	1.393

Fig. A.22 - Ridge ClusterCentroids cc cluster comparison



Best ElasticNet L1/L2 Regression Results

Table A.23 - ElasticNet RandomOverSampler, RandomUnderSampler and No sampling

tokenizer	vectorizer	sampler	Param elastic net alpha	Param elastic net l1 ratio	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	rOs	0.001	0.3	0.318	0.318	0.318	0.318	0.343	1.406
lemmatized	Count Vectorized	None	0.001	0.5	0.296	0.296	0.296	0.296	0.451	1.577

Fig A.24 - ElasticNet Parameter Comparison

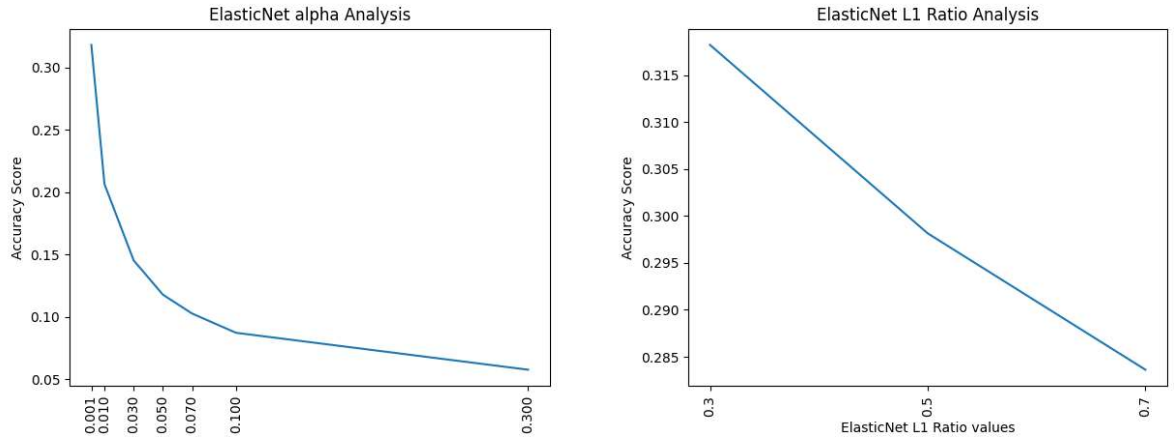


Table. A.25 - ElasticNet Smote Sampler – K Neighbors Comparison

tokenizer	vectorizer	sampler	Smote k neighbors	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
lemmatized	Count Vectorized	smote	1000	0.237	0.237	0.237	0.237	0.323	1.489

Fig A.26 - ElasticNet Smote K Neighbor Comparison

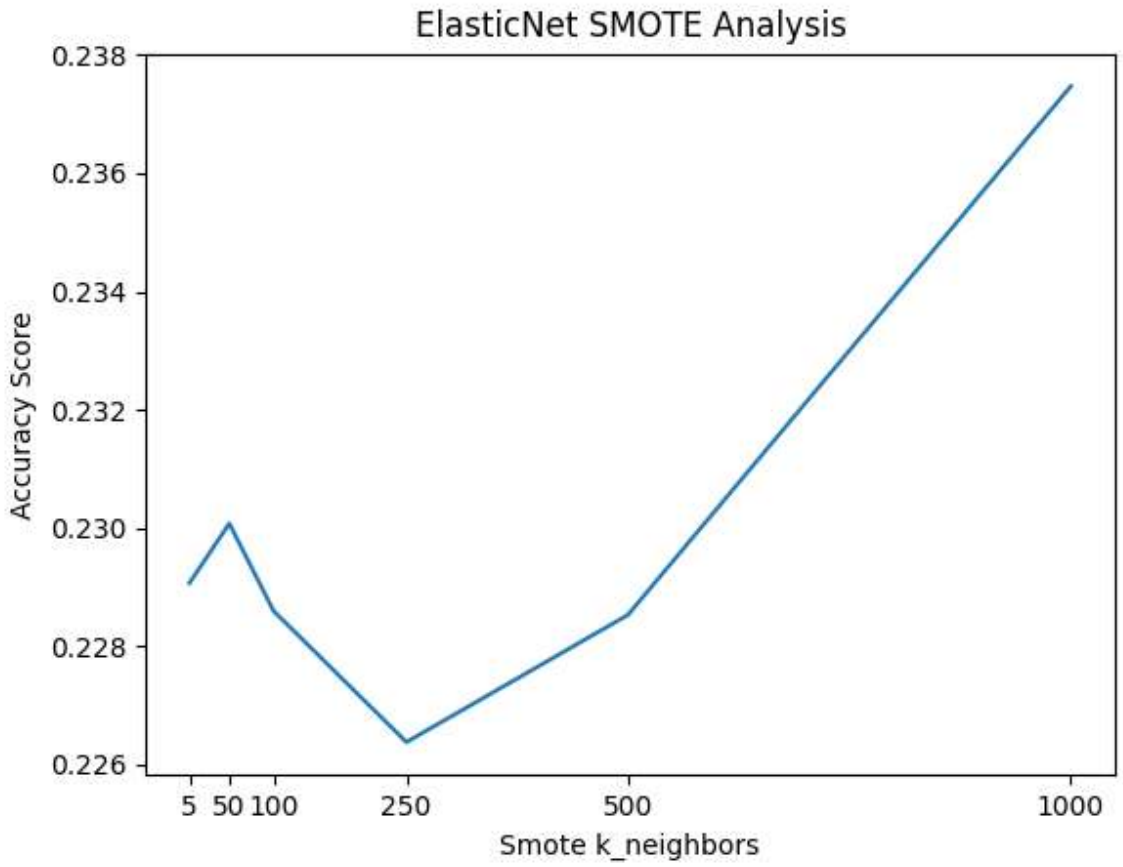


Table. A.27 - ElasticNet ClusterCentroids – cc clusters comparison

tokenizer	vectorizer	sampler	Cc clusters	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	cc	5	0.235	0.235	0.235	0.235	-4.396	1.890

Fig. A.28 - ElasticNet ClusterCentroids cc cluster comparison

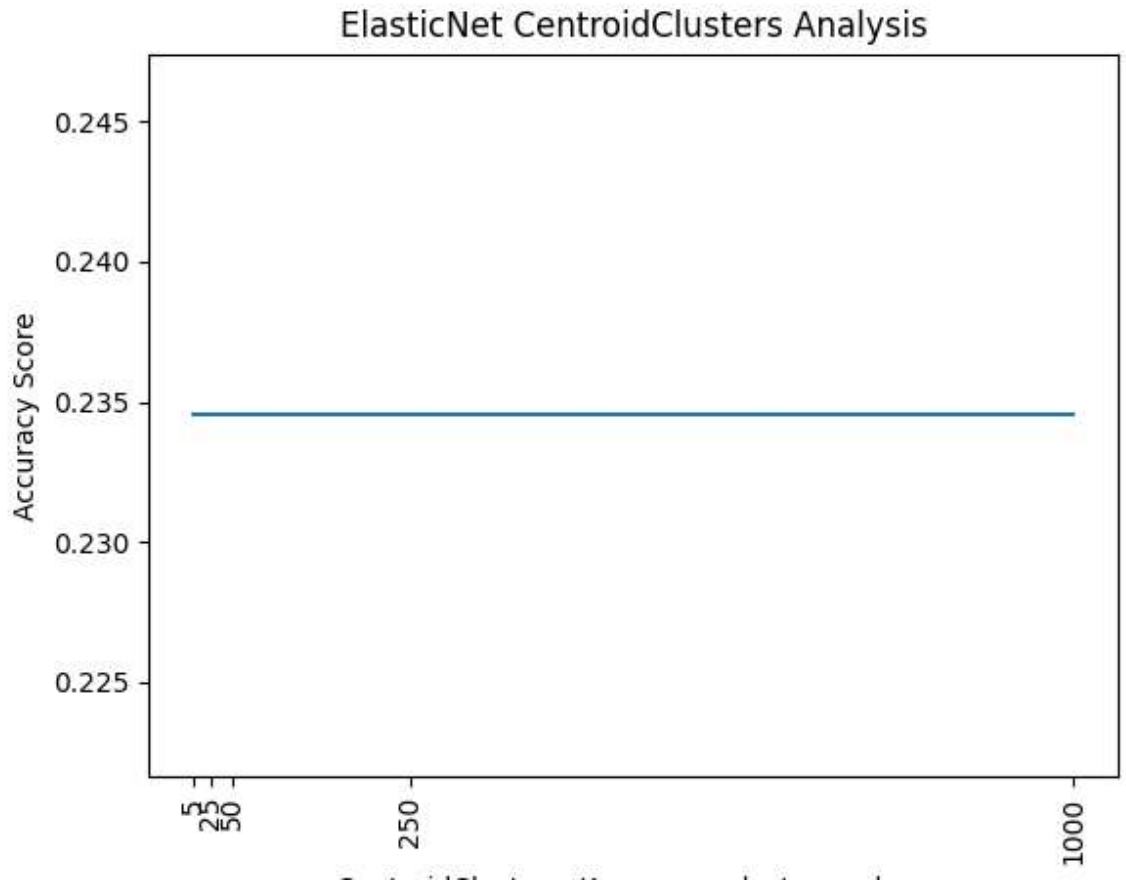


Table A.29 - ElasticNet Best Model Train and Test Results

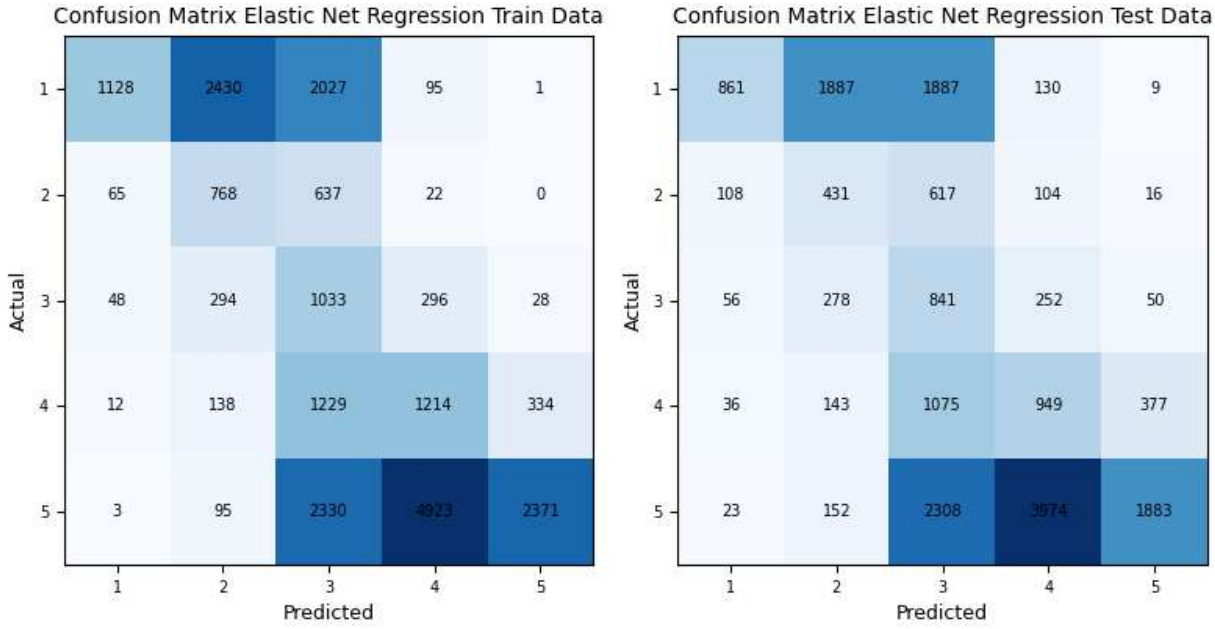
Train

tokenizer	vectorizer	sampler	Param elastic net alpha	Param elastic net l1 ratio	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	rOs	0.001	0.3	0.303	0.680	0.303	0.331	0.507	1.415

Test

tokenizer	vectorizer	sampler	Param elastic net alpha	Param elastic net l1 ratio	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vectorized	rOs	0.001	0.3	0.269	0.615	0.269	0.300	0.423	1.641

Fig. A.30 - ElasticNet Best Model Confusion Matrices



Best HistGradientBoostingRegressor (HGBR) Regression Results

Table. A.31 - HGBR RandomOverSampler, RandomUnderSampler and No sampling

tokenizer	vectorizer	sampler	Param gbr learning rate	Param gbr max depth	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	TFIDF Vectorized	rOs	0.5	1000	0.499	0.499	0.499	0.499	0.429	1.112
stemmatized	TFIDF Vectorized	None	0.2	1000	0.410	0.410	0.410	0.410	0.529	1.351

Fig A.32 - HGBRParameter Comparison

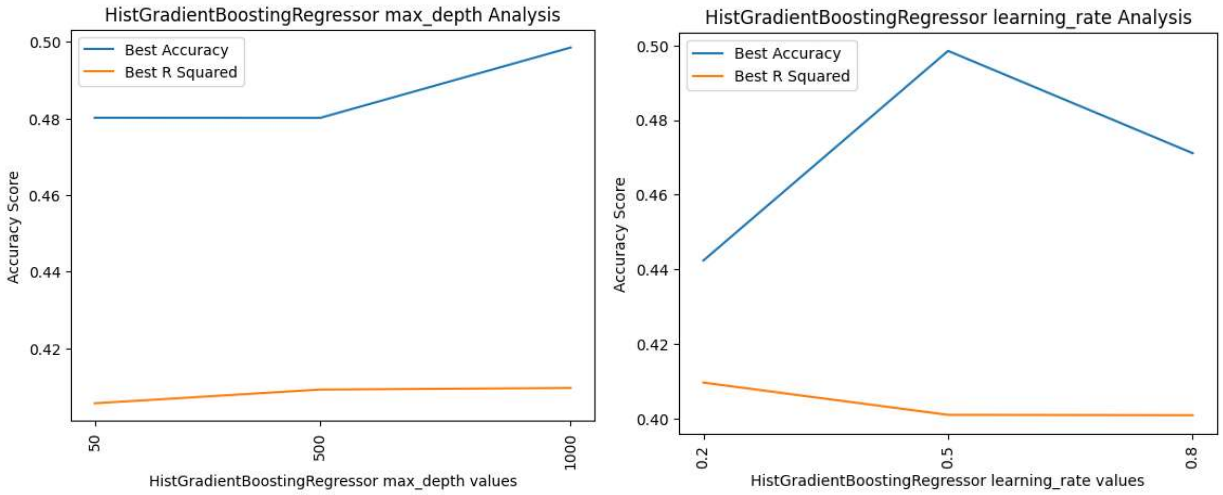


Table. A.33 - HGBR Smote Sampler – K Neighbors Comparison

tokenizer	vectorizer	sampler	Smote k neighbors	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
lemmatized	Count Vectorized	smote	50	0.486	0.486	0.486	0.486	0.393	1.188
lemmatized	TFIDF Vectorized	smote	250	0.443	0.443	0.443	0.443	0.414	1.184
lemmatized	TFIDF Vectorized	smote	100	0.446	0.446	0.446	0.446	0.414	1.176

Fig A.34 - HGBR Smote K Neighbor Comparison

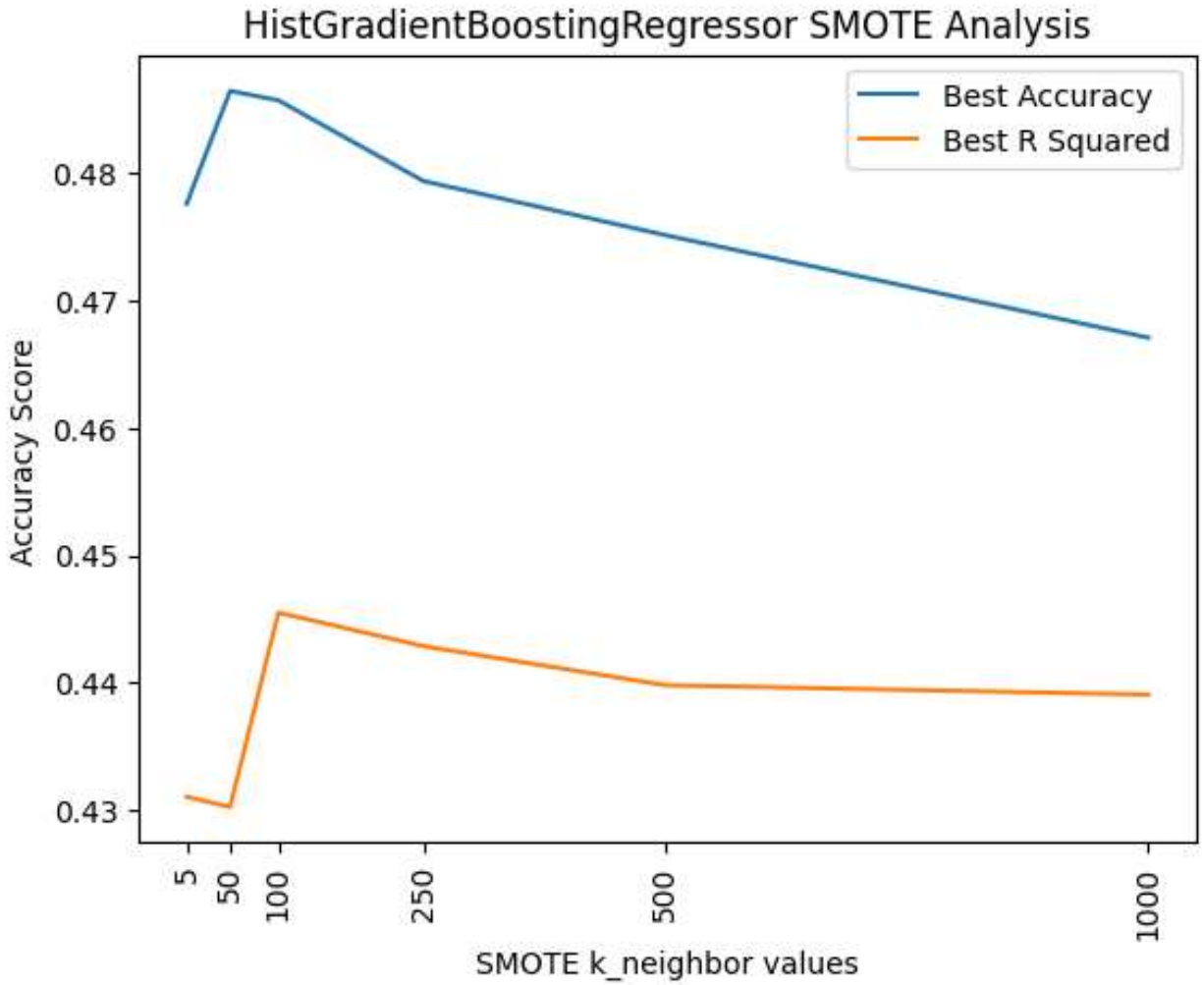


Table. A.35 - HGBR ClusterCentroids – cc clusters comparison

tokenizer	vectorizer	sampler	Cc clusters	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
stemmatized	TFIDF Vectorized	cc	1000	0.284	0.284	0.284	0.284	-5.478	2.255
lemmatized	Count Vectorized	cc	1000	0.269	0.269	0.269	0.267	-4.349	1.888

Fig. A.36 - HGBR ClusterCentroids cc cluster comparison

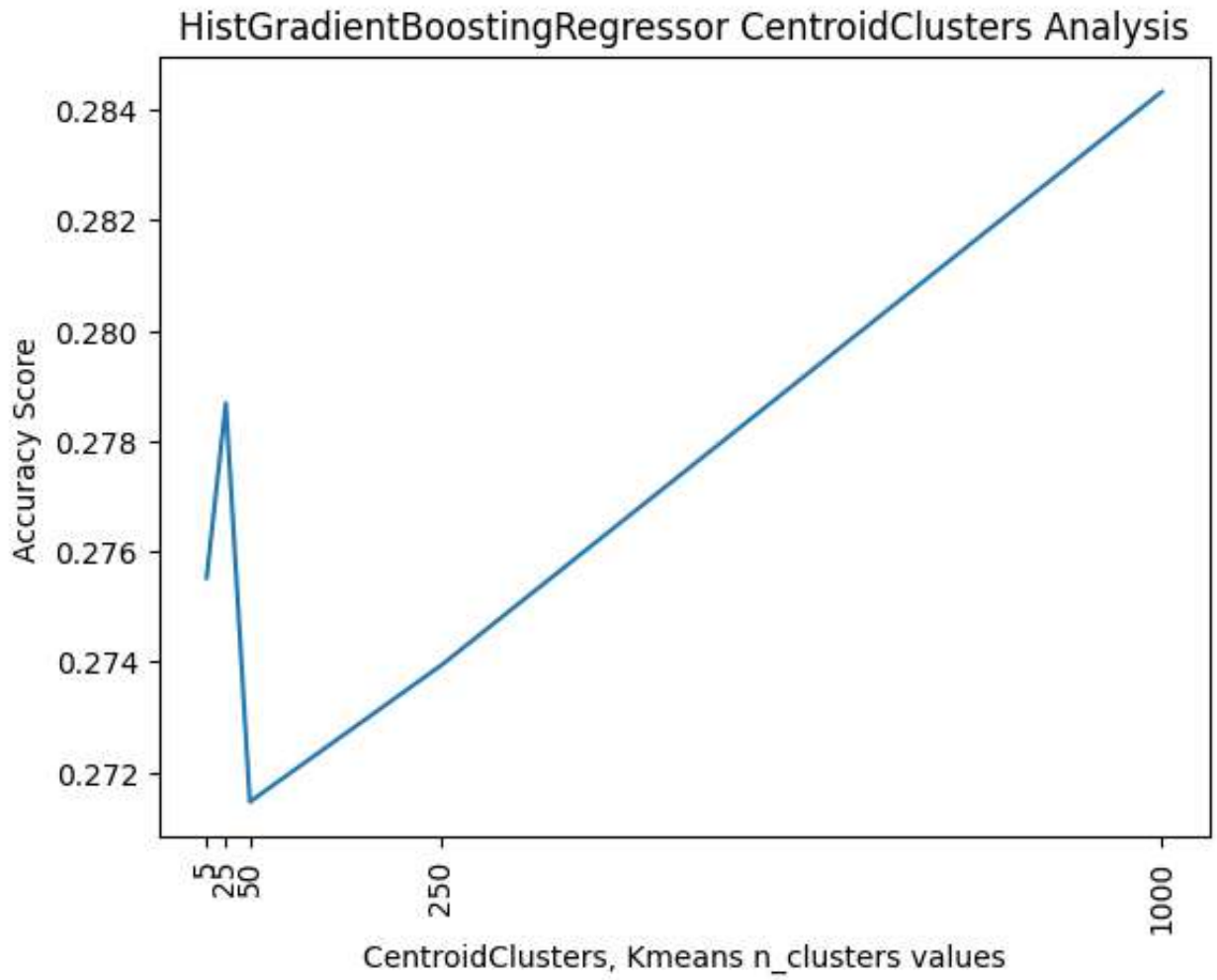


Table. A.37 - HGBR Best Model Train and Test Results

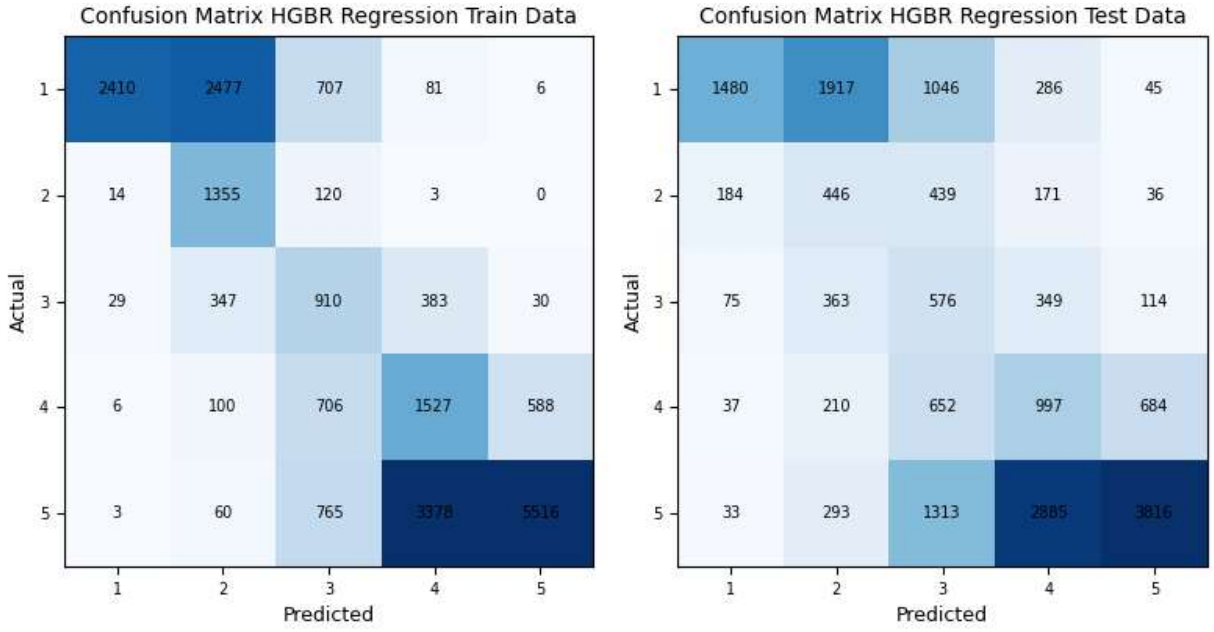
Train

tokenizer	vectorizer	sampler	Param gbr learning rate	Param gbr max depth	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	TFIDF Vectorized	None	0.5	None	0.604	0.740	0.604	0.642	0.563	0.746

Test

tokenizer	vectorizer	sampler	Param gbr learning rate	Param gbr max depth	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	TFIDF Vectorized	None	0.5	1000	0.593	0.727	0.593	0.632	0.508	0.829

Fig. A.38 - HGBR Best Model Comparison



Appendix B – Full Classification Results

Best LogisticRegression Results

Table B.1 - LogisticRegression Preprocessing initial exploration, no Sampling or Gridsearch

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.649	0.58	0.65	0.80	0.59	0.66	0.48
Lemmatized	TFIDF Vectorizer	0.668	0.61	0.67	0.82	0.61	0.68	0.51
Stemmatized	Count Vectorized	0.648	0.58	0.65	0.80	0.59	0.66	0.49
Stemmatized	TFIDF Vectorizer	0.665	0.60	0.66	0.82	0.61	0.68	0.51

Fig B.2 - Logistic Regression Confusion Matrices

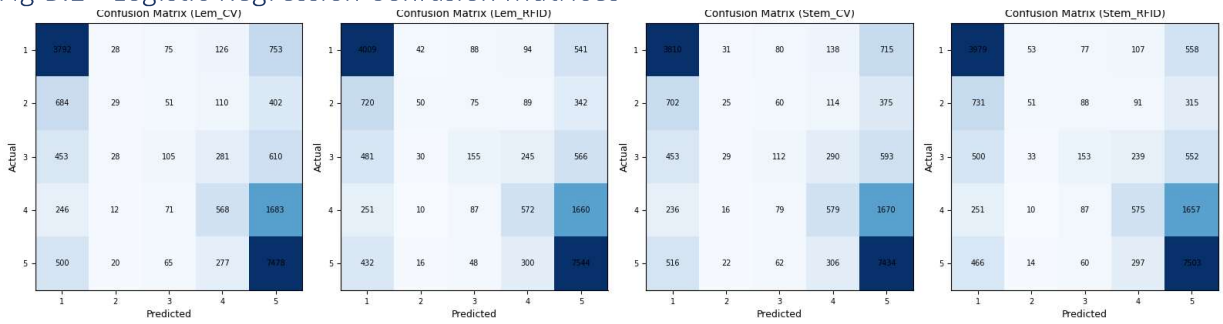


Table B.3 - LogisticRegression Undersampled

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.602	0.72	0.60	0.85	0.64	0.71	0.50
Lemmatized	TFIDF Vectorizer	0.597	0.73	0.60	0.87	0.64	0.72	0.51
Stemmatized	Count Vectorized	0.596	0.72	0.60	0.85	0.64	0.71	0.49
Stemmatized	TFIDF Vectorizer	0.593	0.73	0.59	0.87	0.64	0.71	0.50

Fig. B.4 - Logistic Regression undersampled Confusion Matrices

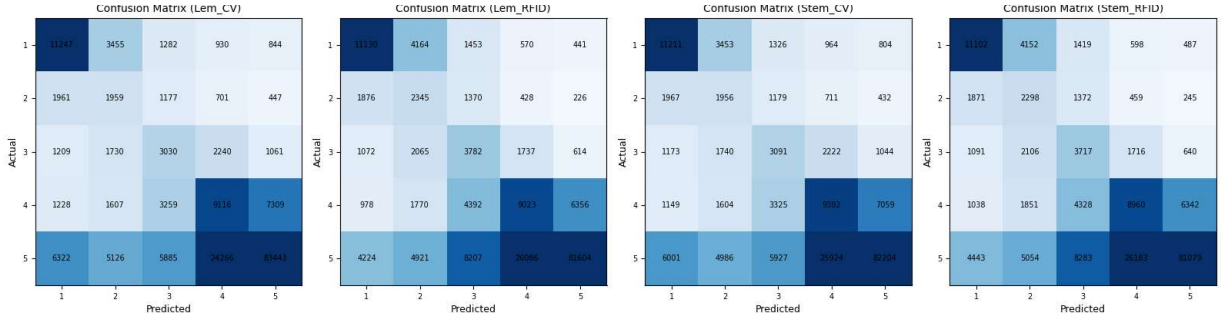


Table B.5 - Logistic Regression Word2Vec Preprocessing, RandomOverSampler

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specifi city	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
None	Google Word2Vec	0.526	0.60	0.53	0.70	0.56	0.57	0.34

Bets Params were:

- C = 100,
- l1_ratio = 0.3,
- penalty = elasticnet

Fig. B.6 - Logistic Regression oversampled Word2VecConfusion Matrices

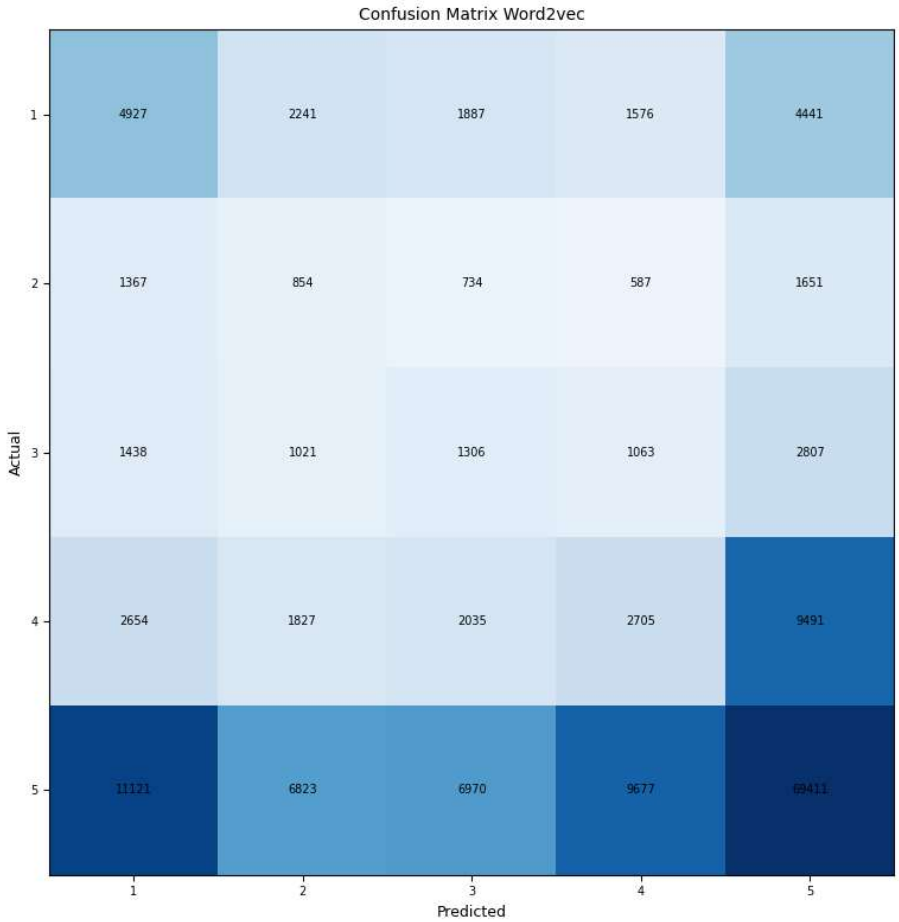


Table B.7 - Logistic Rgression Model Train and Test Results

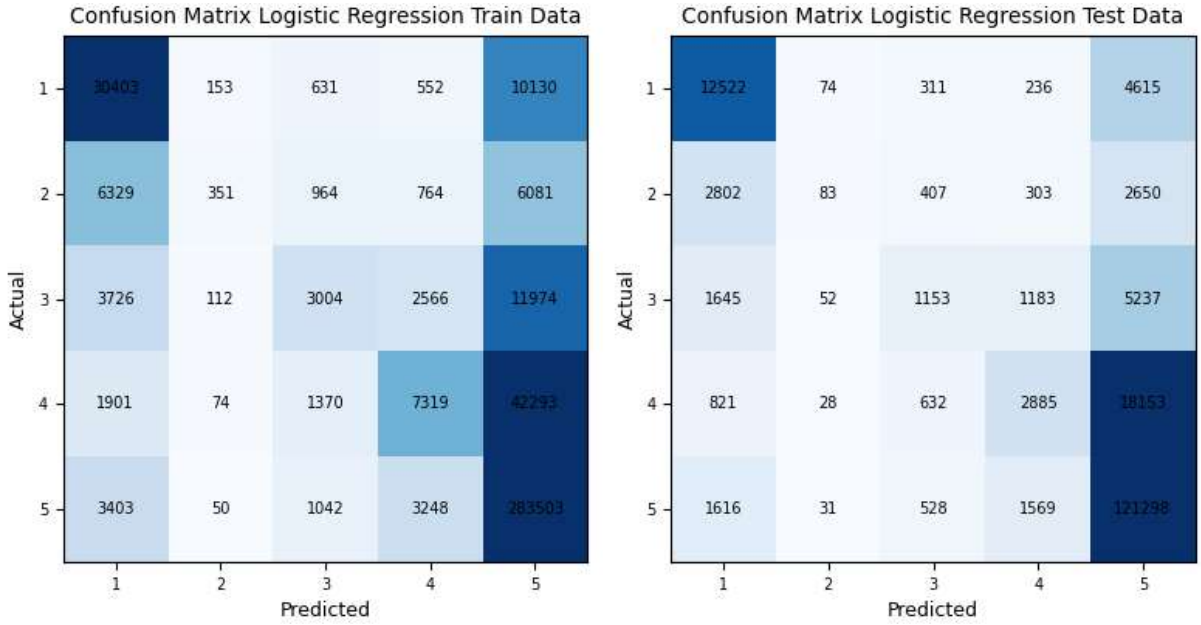
Train

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	TFIDF	None	0.769	0.720	0.769	0.715	0.408	1.004

Test

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	TFIDF	None	0.762	0.703	0.763	0.707	0.376	1.053

Fig. B.8 - Logistic Best Model Confusion Matrices



Best Support Vector Machine Results

Table B.9 - SVM Preprocessing initial exploration, no Sampling or Gridsearch

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specifi city	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.650	0.61	0.65	0.77	0.57	0.62	0.45
Lemmatized	TFIDF Vectorizer	0.670	0.62	0.67	0.80	0.59	0.65	0.49
Stemmatized	Count Vectorized	0.649	0.59	0.65	0.77	0.56	0.61	0.45
Stemmatized	TFIDF Vectorizer	0.669	0.63	0.67	0.80	0.59	0.65	0.49

Fig B.10 - SVM Confusion Matrices

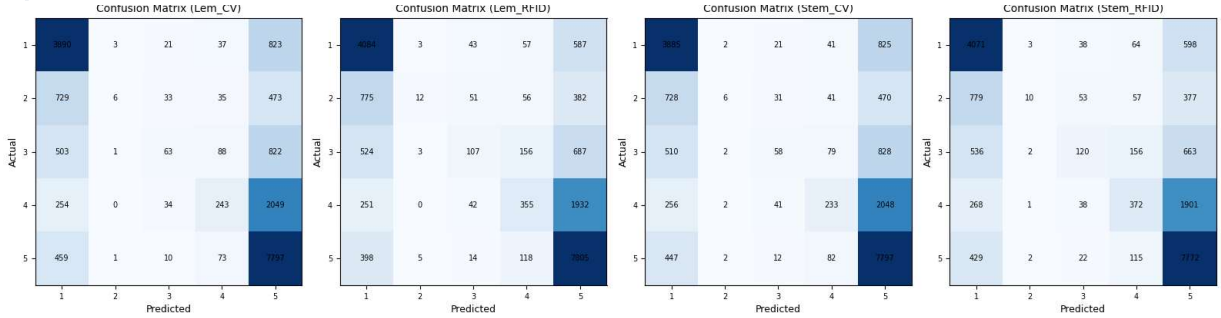


Table B.11 - SVM Undersampled

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specifi city	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.584	0.73	0.58	0.87	0.64	0.71	0.50
Lemmatized	TFIDF Vectorizer	0.594	0.74	0.59	0.88	0.64	0.72	0.51
Stemmatized	Count Vectorized	0.584	0.73	0.58	0.87	0.64	0.71	0.50
Stemmatized	TFIDF Vectorizer	0.594	0.74	0.59	0.88	0.64	0.72	0.51

Fig B.12 - SVM Undersampled Confusion Matrices

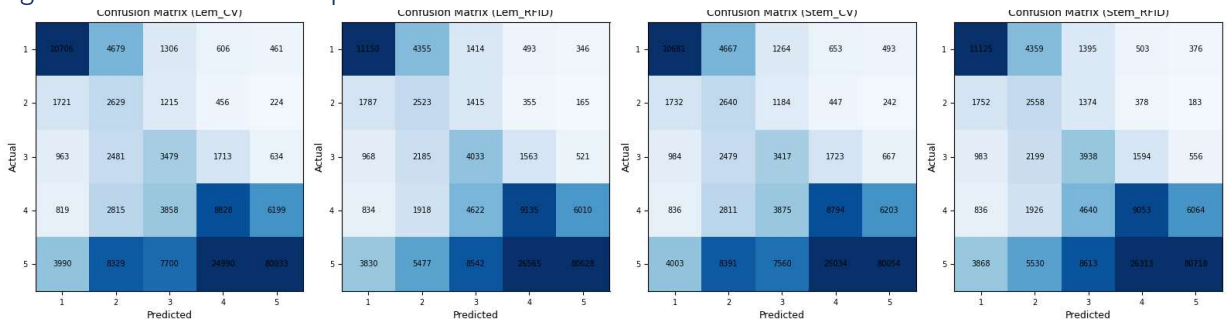


Table B.13 - SVM Classification Model Train and Test Results

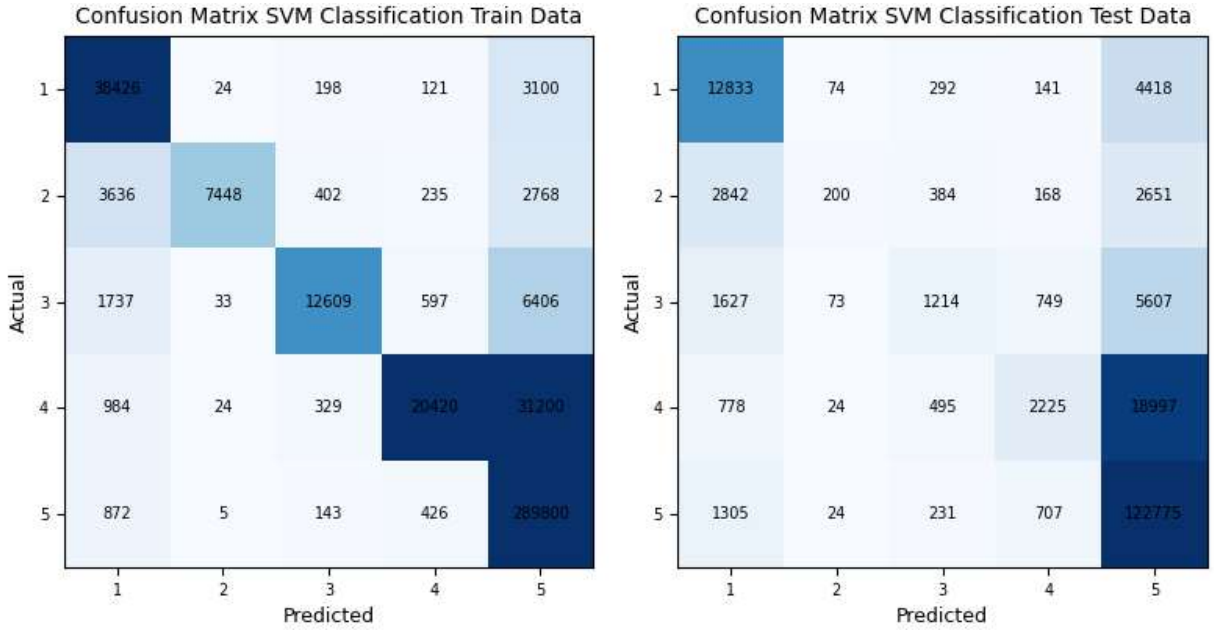
Train

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	TFIDF	None	0.874	0.882	0.814	0.856	0.762	0.403

Test

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	TFIDF	None	0.770	0.725	0.77	0.709	0.410	0.996

Fig. B.14 - SVM Classification Best Model Confusion Matrices



Best K Nearest Neighbour Results

Table B.15 - KNN Preprocessing initial exploration, no Sampling or Gridsearch

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specifi city	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.516	0.44	0.52	0.73	0.46	0.53	0.33
Lemmatized	TFIDF Vectorizer	0.290	0.48	0.29	0.75	0.17	0.20	0.05
Stemmatized	Count Vectorized	0.525	0.45	0.53	0.74	0.47	0.54	0.34
Stemmatized	TFIDF Vectorizer	0.298	0.47	0.30	0.75	0.19	0.23	0.06

Fig B.16 - KNN Confusion Matrices

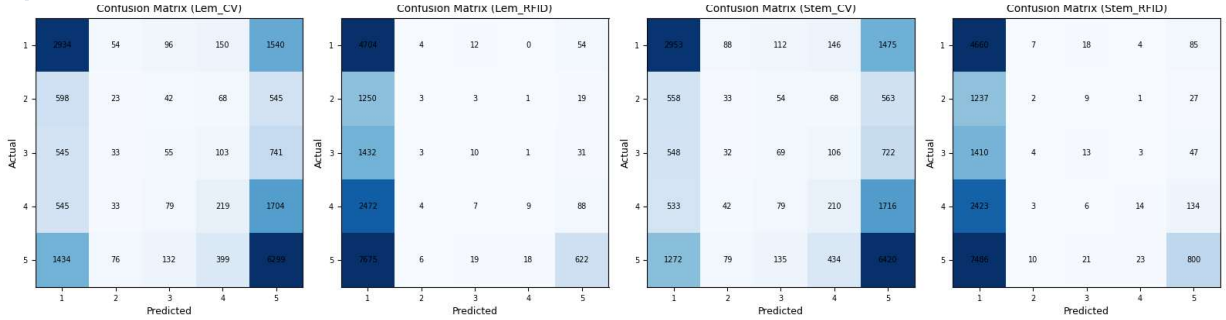


Table B.17 - KNN Undersampled

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specifi city	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.487	0.66	0.49	0.81	0.54	0.62	0.38
Lemmatized	TFIDF Vectorizer	0.248	0.67	0.25	0.89	0.27	0.42	0.18
Stemmatized	Count Vectorized	0.496	0.66	0.50	0.81	0.55	0.63	0.39
Stemmatized	TFIDF Vectorizer	0.261	0.67	0.26	0.89	0.29	0.44	0.19

Fig B.18 - KNN Undersampled Confusion Matrices

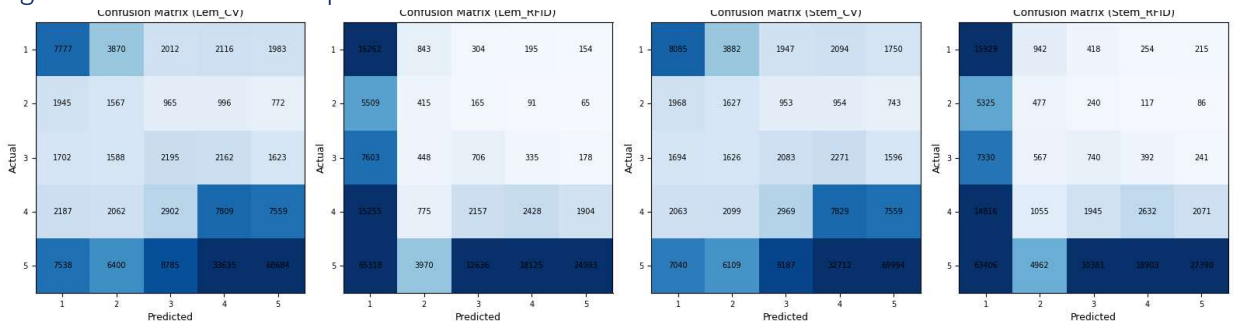


Table B.19 - KNN Classification Model Train and Test Results

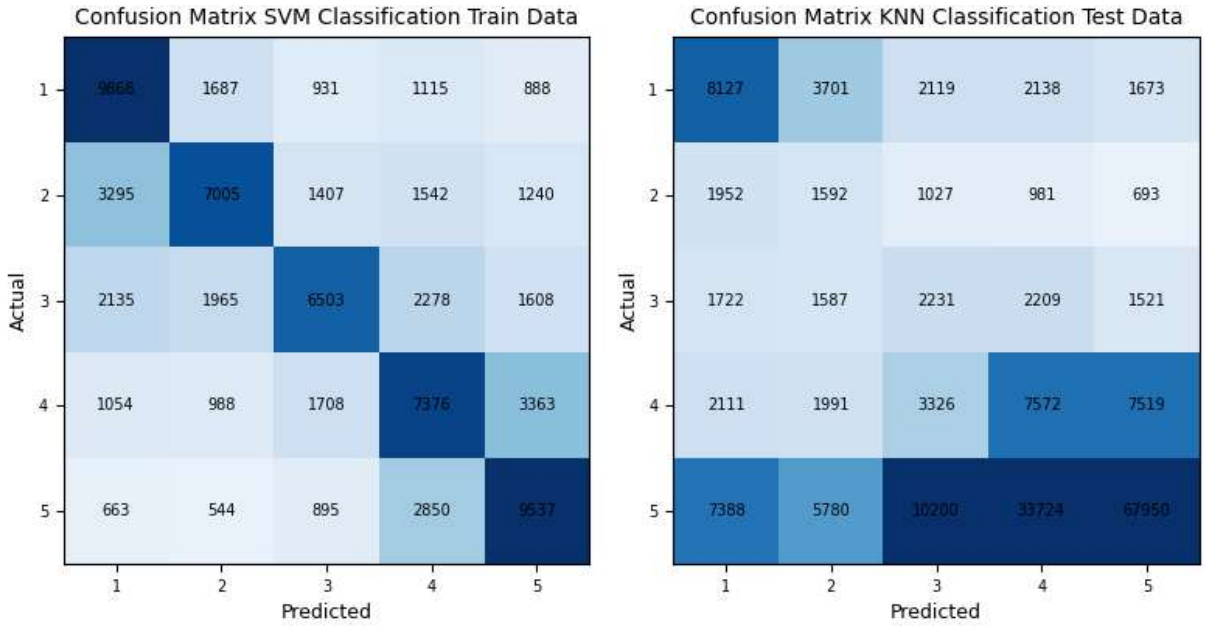
Train

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
stemmatized	Count Vector	rUs	0.556	0.556	0.556	0.552	0.232	1.537

Test

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
stemmatized	Count Vector	rUs	0.484	0.660	0.484	0.541	-0.214	2.050

Fig. B.20 - KNN Classification Best Model Confusion Matrices



Best Decision tree Classifier Results

Table B.21 - Decision Tree Preprocessing initial exploration, no Sampling or Gridsearch

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specifi city	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.518	0.50	0.52	0.80	0.51	0.61	0.39
Lemmatized	TFIDF Vectorizer	0.515	0.50	0.52	0.80	0.50	0.61	0.38
Stemmatized	Count Vectorized	0.516	0.50	0.52	0.80	0.51	0.61	0.39
Stemmatized	TFIDF Vectorizer	0.505	0.49	0.50	0.80	0.50	0.60	0.38

Fig B.22 - Decision Tree Confusion Matrices

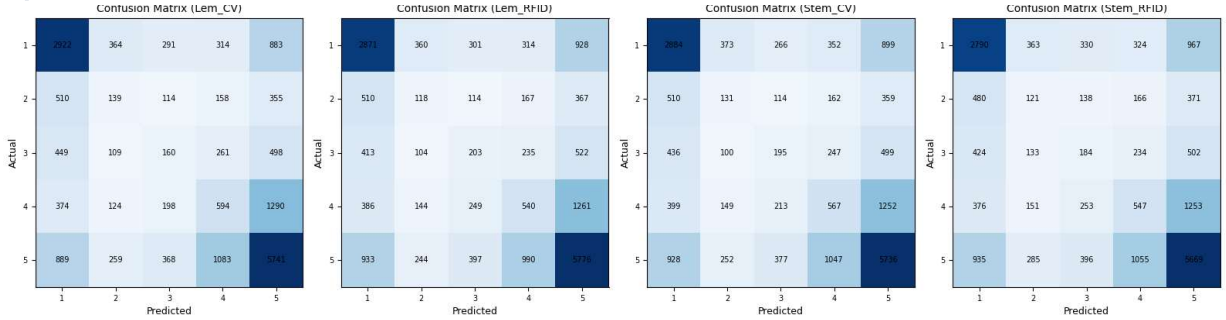


Table B.23 - Decision Tree Undersampled

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specifi city	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.464	0.68	0.46	0.84	0.53	0.62	0.38
Lemmatized	TFIDF Vectorizer	0.466	0.68	0.47	0.85	0.53	0.63	0.38
Stemmatized	Count Vectorized	0.463	0.68	0.46	0.84	0.53	0.62	0.38
Stemmatized	TFIDF Vectorizer	0.466	0.68	0.47	0.84	0.53	0.63	0.38

Fig B.24 - Decision Tree Undersampled Confusion Matrices

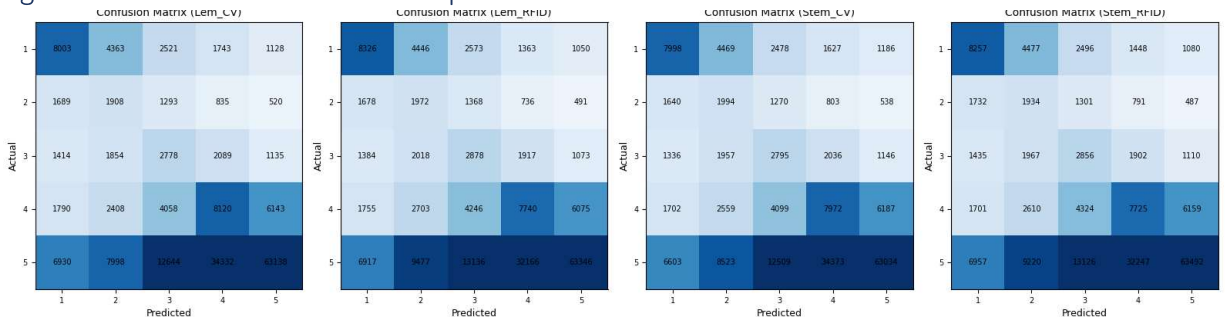


Table B.25 - Decision Tree Classification Model Train and Test Results

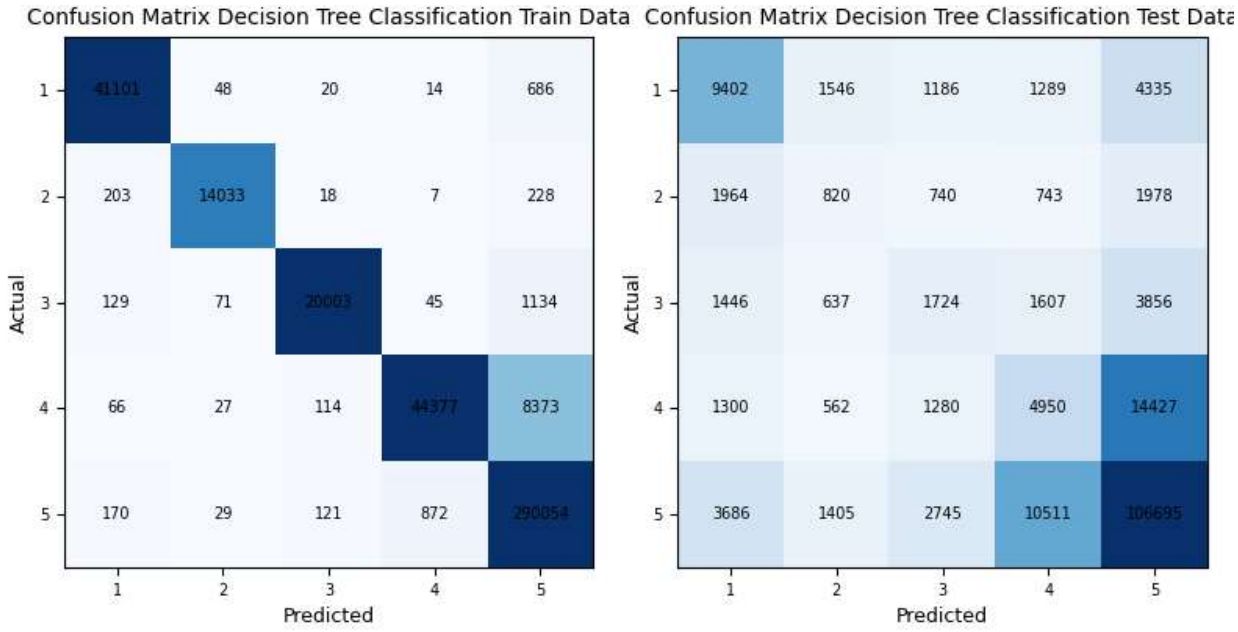
Train

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vector	None	0.971	0.971	0.971	0.970	0.954	0.076

Test

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vector	None	0.683	0.663	0.683	0.673	0.158	1.421

Fig. B.26 - Decision Tree Classification Best Model Confusion Matrices



Best Random Forest Classifier Results

Table B.27 - Random Forest Classifier Preprocessing initial exploration, no Sampling or Gridsearch

Train

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Lemmatized	TFIDF Vectorizer	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Stemmatized	Count Vectorized	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Stemmatized	TFIDF Vectorizer	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Test

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.639	0.65	0.64	0.75	0.54	0.57	0.42
Lemmatized	TFIDF Vectorizer	0.636	0.66	0.64	0.75	0.54	0.56	0.41
Stemmatized	Count Vectorized	0.637	0.64	0.64	0.75	0.54	0.57	0.41
Stemmatized	TFIDF Vectorizer	0.638	0.68	0.64	0.75	0.54	0.57	0.41

Fig B.28 - Random Forest Confusion Matrices

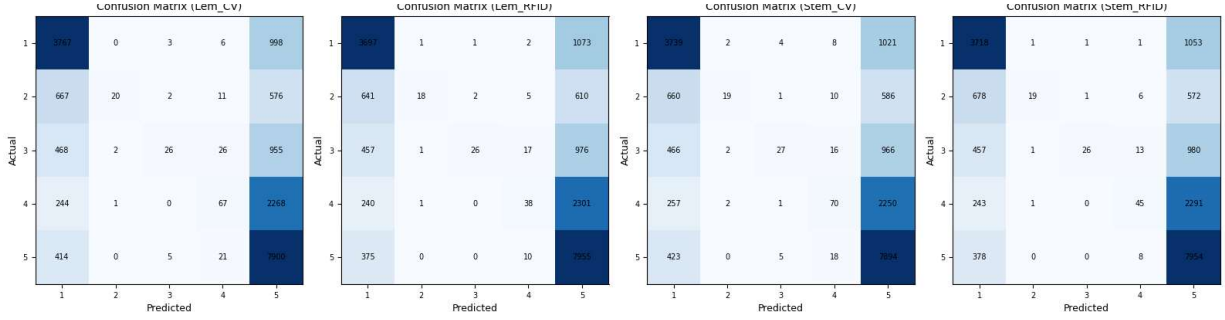


Table B.29 - Random Forest Classifier Undersampled

Train

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.96	0.96	0.96	0.99	0.96	0.98	0.95
Lemmatized	TFIDF Vectorizer	0.96	0.96	0.96	0.99	0.96	0.98	0.95
Stemmatized	Count Vectorized	0.96	0.96	0.96	0.99	0.96	0.97	0.95
Stemmatized	TFIDF Vectorizer	0.96	0.96	0.96	0.99	0.96	0.97	0.95

Test

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.577	0.71	0.58	0.85	0.62	0.70	0.48
Lemmatized	TFIDF Vectorizer	0.570	0.71	0.57	0.86	0.62	0.70	0.48
Stemmatized	Count Vectorized	0.568	0.71	0.57	0.86	0.62	0.69	0.47
Stemmatized	TFIDF Vectorizer	0.566	0.71	0.57	0.86	0.61	0.70	0.48

Fig B.30 - Random Forest Undersampled Confusion Matrices

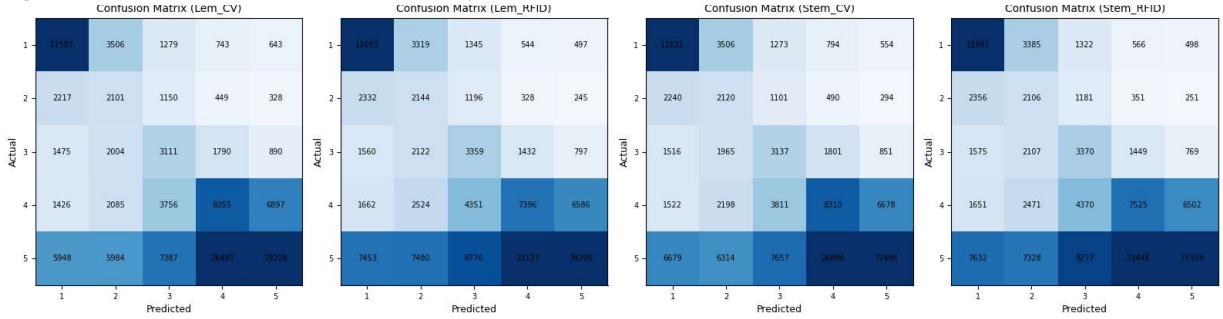


Table B.31 - RandomForest Word2Vec Preprocessing, RandomOverSampler

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy	Test mean mse
None	Google Word2Vec	0.651	0.55	0.65	0.36	0.57	0.26	0.07	2.31

Best Parameters were:

- max_depth = None,
- min_samples_leaf = 1,
- min_samples_split = 2,
- n_estimators = 200

Fig. B.32 Logistic Regression oversampled Word2VecConfusion Matrices

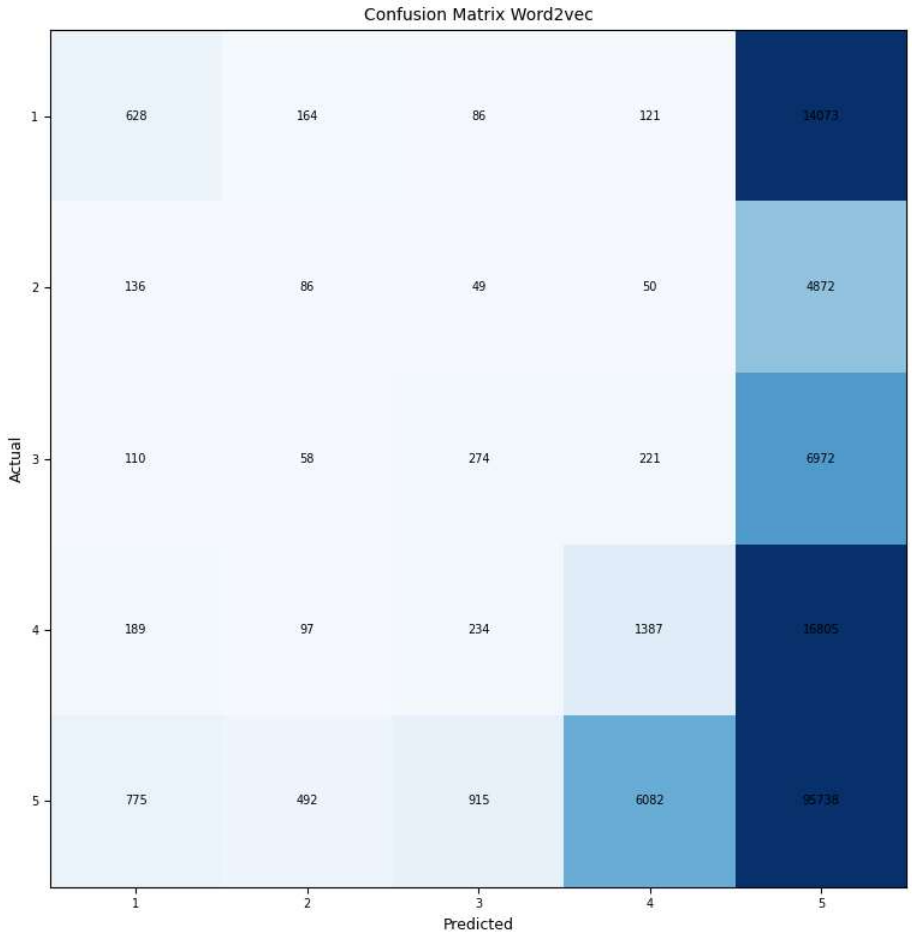


Table B.33 - Random Forest Classification Model Train and Test Results

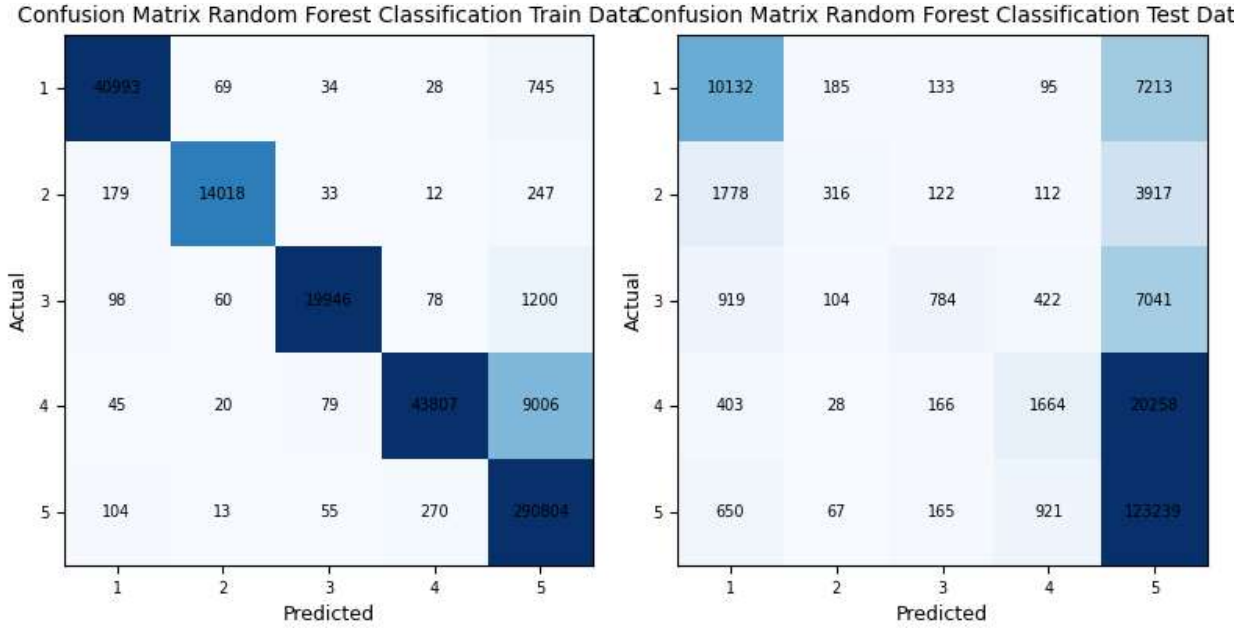
Train

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vector	None	0.971	0.971	0.971	0.970	0.955	0.076

Test

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
lemmatized	Count Vector	None	0.753	0.708	0.753	0.684	0.267	1.24

Fig. B.34 - Random Forest Classification Best Model Confusion Matrices



Best Naive Bayes Classifier Results

Table B.35 - Naïve Bayes Preprocessing initial exploration, no Sampling or Gridsearch

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.625	0.56	0.63	0.80	0.58	0.65	0.47
Lemmatized	TFIDF Vectorizer	0.593	0.45	0.59	0.68	0.49	0.45	0.29
Stemmatized	Count Vectorized	0.619	0.56	0.62	0.81	0.58	0.66	0.47
Stemmatized	TFIDF Vectorizer	0.593	0.59	0.59	0.69	0.49	0.45	0.30

Fig B.36 - Naïve Bayes Confusion Matrices

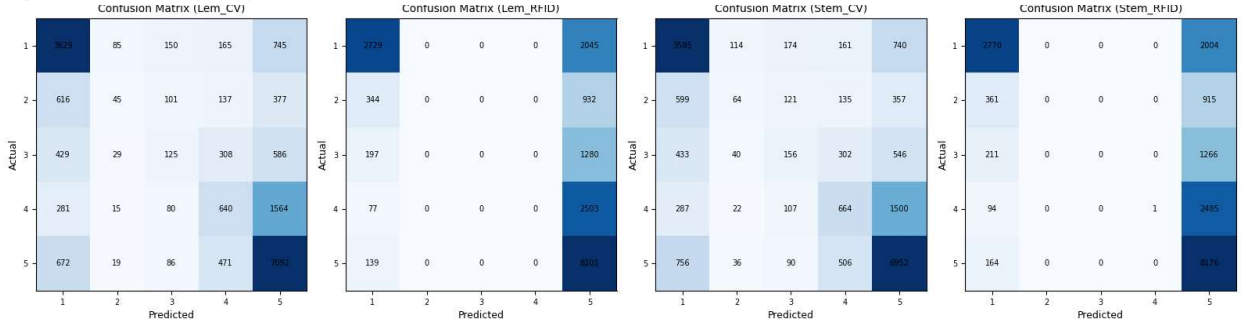


Table B.37 - Naïve Bayes Undersampled

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.628	0.71	0.63	0.83	0.66	0.71	0.51
Lemmatized	TFIDF Vectorizer	0.563	0.71	0.56	0.87	0.61	0.69	0.47
Stemmatized	Count Vectorized	0.630	0.71	0.63	0.83	0.66	0.71	0.50
Stemmatized	TFIDF Vectorizer	0.560	0.71	0.56	0.86	0.61	0.69	0.47

Fig B.38 - Naïve Bayes Undersampled Confusion Matrices

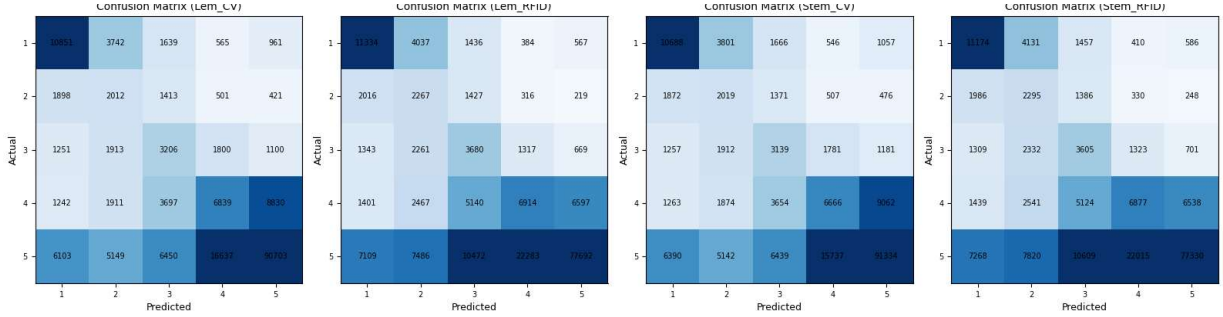


Table B.39 - Naïve Bayes Classification Model Train and Test Results

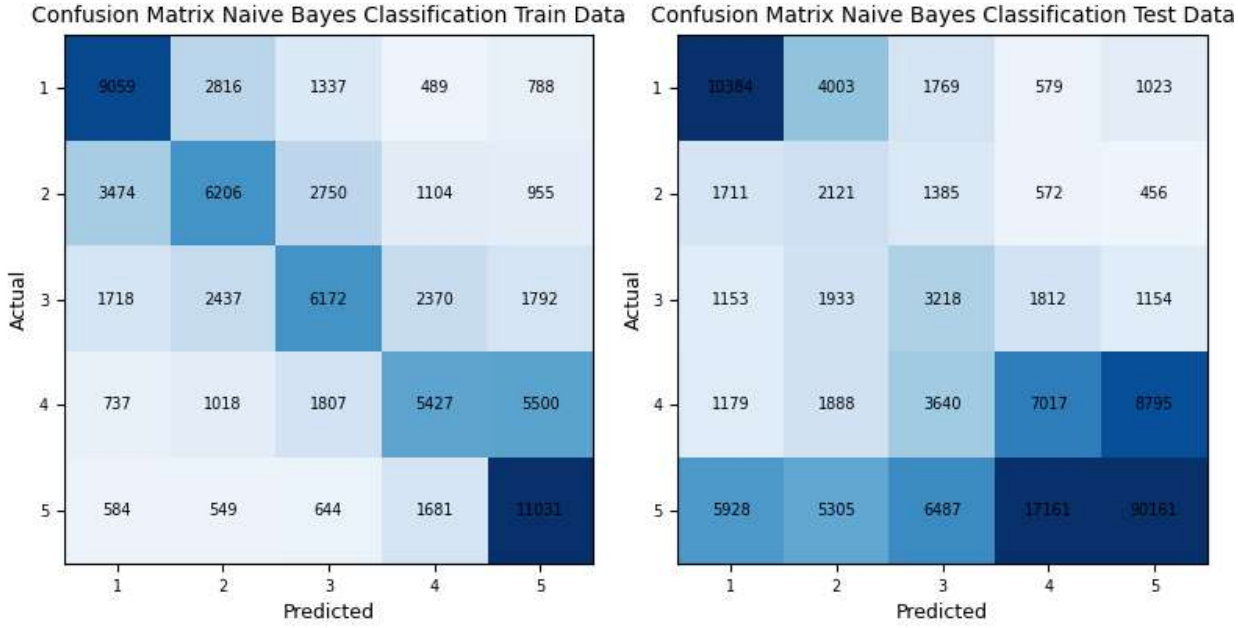
Train

tokenizer	vectorizer	sampler	Mean test accuracy	Mean test precision	Mean test recall	Mean test f1 score	Mean test r2 score	Mean test mse
stemmatized	Count Vector	rUs	0.523	0.517	0.523	0.514	0.311	1.378

Test

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
stemmatized	Count Vector	rUs	0.624	0.711	0.624	0.659	0.111	1.501

Fig. B.40 - Naïve Bayes Classification Best Model Confusion Matrices



Best HistGradientBoostingClassifier Results

Table B.41 - HistGradientBoostingClassifier Preprocessing initial exploration, no Sampling or Gridsearch

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specificity	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.661	0.60	0.66	0.81	0.61	0.68	0.50
Lemmatized	TFIDF Vectorizer	0.659	0.60	0.66	0.82	0.61	0.68	0.51
Stemmatized	Count Vectorized	0.659	0.60	0.66	0.81	0.60	0.67	0.50
Stemmatized	TFIDF Vectorizer	0.661	0.60	0.66	0.82	0.61	0.68	0.51

Fig B.42 - HistGradientBoostingClassifier Confusion Matrices

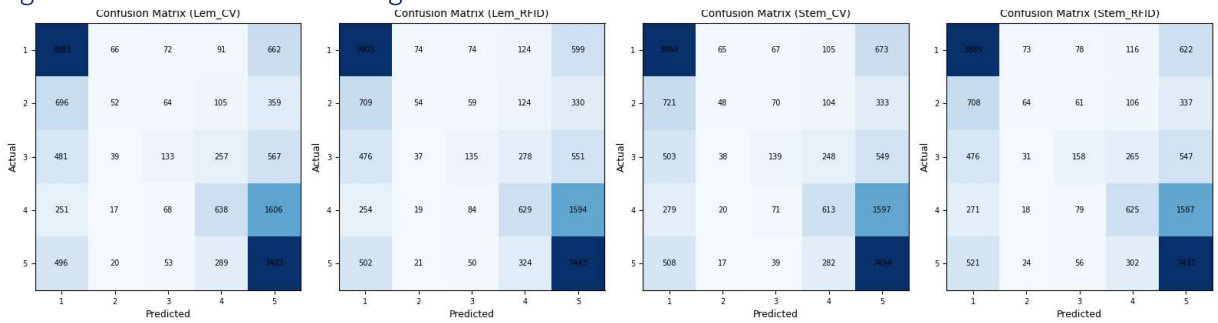


Table B.43 - HistGradientBoostingClassifier Undersampled

tokenizer	vectorizer	Mean test accuracy	Mean test precision	Mean test recall	Mean test specifi city	Mean test f1 score	Geo-metric Mean error	Index of balanced Accuracy
Lemmatized	Count Vectorized	0.600	0.73	0.60	0.86	0.65	0.72	0.50
Lemmatized	TFIDF Vectorizer	0.597	0.73	0.60	0.87	0.65	0.72	0.50
Stemmatized	Count Vectorized	0.600	0.73	0.60	0.86	0.65	0.72	0.51
Stemmatized	TFIDF Vectorizer	0.595	0.73	0.60	0.87	0.64	0.71	0.50

Fig B.44 - HistGradientBoostingClassifier Undersampled Confusion Matrices

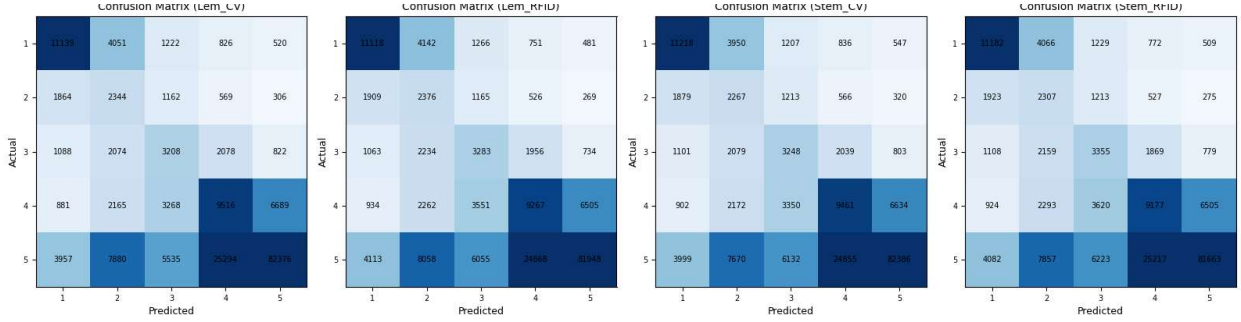


Table B.45 - Best Histogram Gradient Boosting Classification Model Train and Test Results

Train

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
stemmatized	Count Vector	None	0.782	0.752	0.782	0.745	0.407	1.00

Test

tokenizer	vectorizer	sampler	**Mean test accuracy**	**Mean test precision**	**Mean test recall**	**Mean test f1 score**	**Mean test r2 score**	**Mean test mse**
stemmatized	Count Vector	None	0.742	0.682	0.742	0.698	0.299	1.18

Fig. B.46 - Histogram Gradient Boosting Classification Best Model Confusion Matrices