CS 584: Data Mining (HW2)

Md. Ridwan Hossain Talukder
miner2 ID: Luffy
Rank on miner2: 13
Accuracy: 0.77

I. Problem Statement

In this homework we are trying to develop predictive models that can determine given a particular compound whether it is active (1) or not (0).

II. Methodology

The training data set has an imbalanced distribution of 78 actives (1) and 722 inactives (0). So before trying to fit the data to any model we need to do some kind of feature engineering, like feature reduction, feature selection or both. I have tried to do feature reduction, feature selection, tried different resampling techniques (over sampling and under sampling, combining both) and finally tried to use models from NaiveBayesClassifier, DecisionTreeClassifier and MLPClassifier from neural network to measure the performance in terms of f1_score.

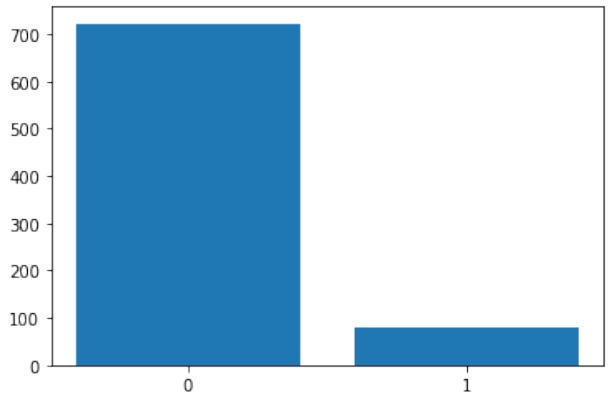

Fig. 1: Imbalanced distribution of dataset

A. Feature Engineering

1) Feature Selection: I have tried both Feature Selection (SelectKBestFeatures, SelectPercentile) and Feature Reduction eg: dimensionality reduction techniques (PCA and SVD) on the training data and used DecisionTreeClassifier, NaiveBayesClassifier (BernoulliNB, as BernoulliNB is designed for boolean features) and a neural network (MLPClassifier) on the reduced vector after feature selection or feature reduction technique. But the F1 score for using PCA was very low considering the F1 score I got using k best feature selection, on the other hand TruncatedSVD took forever to run so I came to a decision that feature selection technique might be good for this problem. As result I decided to go with the feature selection technique instead of the feature reduction technique. Except for the TruncatedSVD the running time for other technique was always below 1 min.

TABLE I: F1-Score for 3 models with reduced features using PCA and SelectKBestFeatures

	DTree	BNB	MLP
PCA	0.44	0.57	0.27
SelectKBest	0.63	0.77	0.62

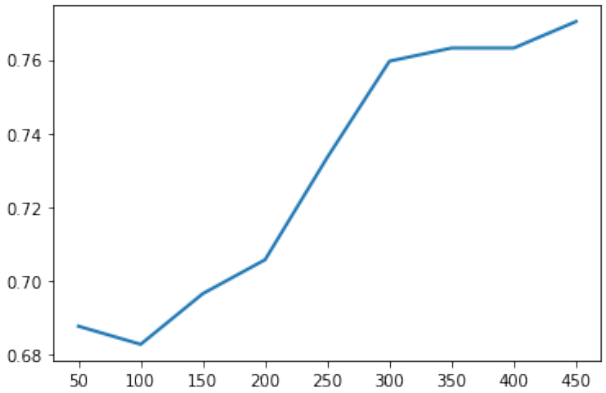

Fig. 2: F1 Score for different K value (Range 50-450)

SelectKBestFeatures selects the feature according to the k highest scores, for that I used the chi_square score function to select the k best features. This function "weeds out" the features that are most likely to be independent of class and, therefore, unimportant for classification because the chi-square test analyzes dependence between stochastic variables. I have also stratified my training data for validation, as this is an imbalanced classification problem we need to allocate the samples evenly based on sample classes so that training set and validation set have similar ratio of classes. So I used RepeatedStratifiedKFold for that purpose. I used 5 fold and 3 repetition to cross validate the f1_score and took the average. In Figure 2 as I can see that the curse skewed after some value 300. I have decided to use a value from range 250 to 350 to check which k value gives the best f1 score. And I got 250 is the last k value after which the f score got great hike. And it is supported by the precision-recall curve in figure 4 where the recall and precision is highest in some value between 300-350. So I have decided to experiment with K value of 200-350 for the next phase. Though it may be appealing for a higher

f1_score for a k value of 400 or higher but I tried that on miner and the f1_score was lower than any value below 300, so I think the model was over fitting for any value above 350.

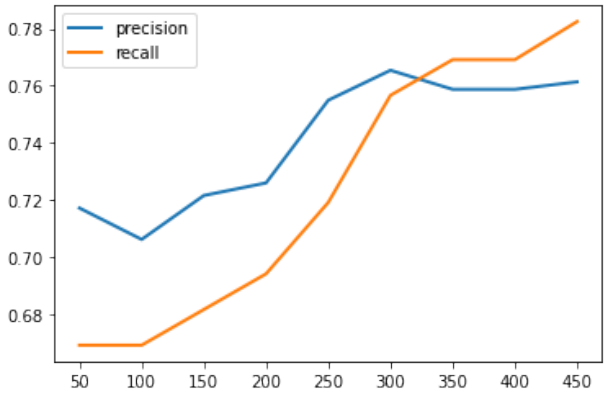

Fig. 3: Precision-Recall Curve for different k value

2) ReSampling: As this is an imbalanced dataset, resampling procedure might greatly help doing better classification. As the number of active records are very low in comparison to inactive records, I have used some undersampling techniques (RandomUnderSampler, TomekLinks, NeighbourhoodCleaningRule, RepeatedEditedNearestNeighbours, CondensedNearestNeighbour) to undersample the inactive(0) records and amongst them RandomUnderSampling gave the best f1 score. Though the CondensedNearestNeighbour seemed to have a better score it performed very poorly on miner. I also tried the oversampling techniques as well but that performed really good on train data but poor on test data.

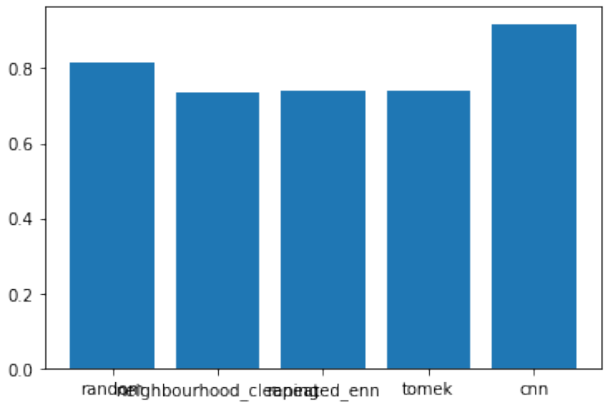

Fig. 4: Performence of Resampling Techniques

After Selecting the undersampling technique (RandomUnderSampler), model (BernoulliNB or DecisionTreeClassifier, we discarded NN as it was poorly performing on this type of feature selection method) and k value(200-350) for SelectKBestFeatures I performed the RepeatedStratifiedKFold again to see which combination gave the best f1_score. And as you can see in figure 5 DecisionTree though most of the time gave higher f1 score but performed poorly on miner with respect to the same combination for BernoulliNB that was because of over fitting. The number of sample became lower after resampling and feature got reduced due to feature selection technique it must have over fitted the data for the DecisionTreeClassifier and also submitting some of the other combination with decision tree classifier proves it true as the f1 score on miner for decision tree was always below 0.65, while BernoulliNB performed better for this kind of feature engineering. Without any feature engineering I got a 0.6 accuracy on miner2 while performing a resampling first (RandomUnderSampler) and then feature selection with SelectKBestFeatures(k=250) I got 0.77 f1_score on miner.

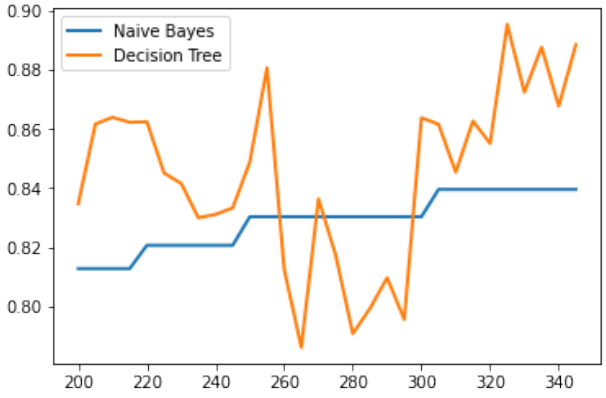

Fig. 5: Performence of DecisionTree and NaiveBayes(BernoulliNB) for different K value

III. Results

Our model gave an average f1 score of 82.1% on training data when we cross validated with RepeatedStratifiedKFold with k-fold(k=5). However, it performed less in miner(0.77 accuracy on miner2) on the test data (approximately as the percentage on the miner is approximate at the time of writing this report).

TABLE II: F1-Score for different combination with NaiveBayesClassifier

Combination	f1_Score
RandomUnder Sampling, KBest Feature Selection (K=250) and NaiveBayes	0.77
KBest Feature Selection (K=250) and NaiveBayes	0.74
RandomUnder Sampling, and NaiveBayes	0.62