Intelligent Systems - Predicting Biodegradability of Chemicals

Report

May 17, 2023

1 Introduction

This is a report on the second seminar Assignment for Intelligent Systems class. The task of this assignment was to use a given data set and analyze it to predict whether a chemical is readily biodegradable or not. The data set consists of 41 features and a binary target class, with 1055 instances.

2 Exploration

This section consists of data set analysis. The goal here was to get the general idea of the features, which ones matter more and which less, which features are more or less correlated to each other etc.

2.1 Basic information

As mentioned above, the data set consists of 1055 instances, which are further divided into two sets: training and test set. The training set includes 846 instances - 564 with target class 1 and 282 with target class 2. Thus, the representation of each target class is not completely balanced, as class 1 has twice as many examples representing it.

Some rows are also missing some data, which is a problem we solved by replacing such values by the mean value of the column. Another way of dealing with this would be to simply discard such rows, but since our data set relatively small, that would mean potentially losing a lot of examples. Of course, by using the mean we are assuming, that that is the most neutral value.

The feature space is visualized in 3 different ways: parallel plot, heat map and 2 dimensional plot of the feature space reduced with t-SNE. The parallel plot shows that instances of class 2 resemble the slight "outliers" when it comes to values of the features, as the values are often on the outer edge of the values that the instances of the target class 1 have. Visually those instances almost represent a border in some features. The heat map, where each column represents a feature, shows the variance of the features - the ones with low variance will have

a constant color throughout the column, the ones with high variance will change the color. The heat map has been divided into columns of 100 to allow easier analysis. The third visualization is a t-SNE reduced set, which shows 4 visual clusters, but unfortunately they do not separate the classes, as seen with the coloring.

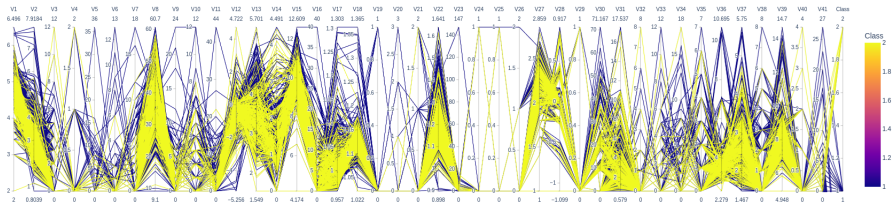


Figure 1: Feature space of the data set

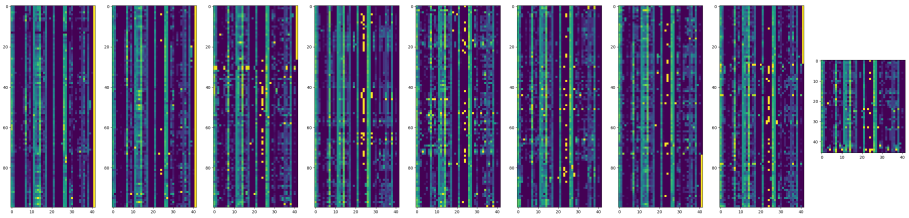


Figure 2: Feature space heat map

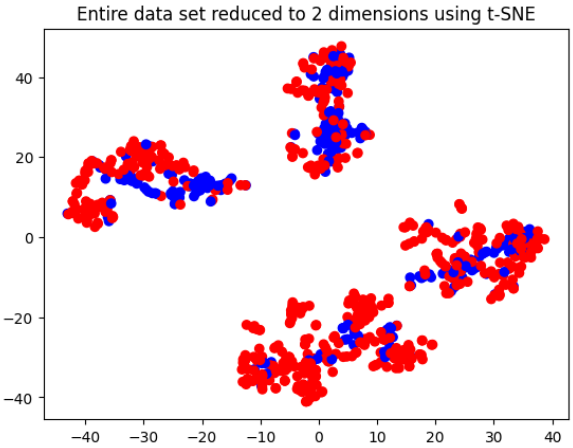


Figure 3: Feature space reduced to 2 dimensions with t-SNE

2.2 Feature analysis

Upon generating a correlation matrix and analyzing the relationships between features and the target, the results show that none of the features are directly correlated to the target class, whereas the analysis of the relationships between different features has shown that some of the features are very closely related to each other. Later, when dealing with feature selection, some of the highly correlated features were removed, to speed up the computation time.

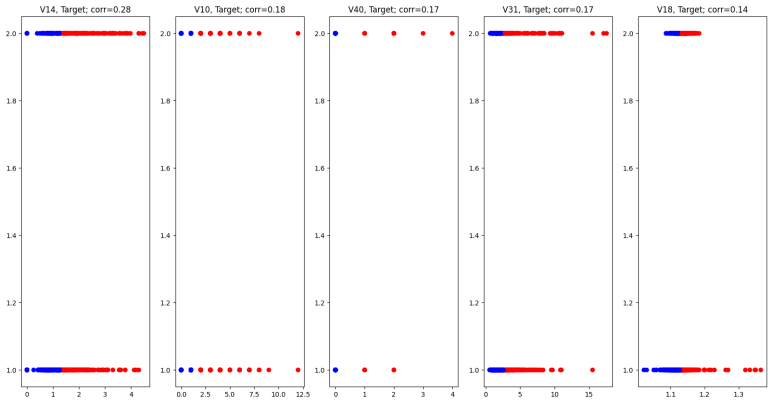


Figure 4: Feature-target relations, colored by the mean value of the feature (y-axis: target class; x-axis: feature value)

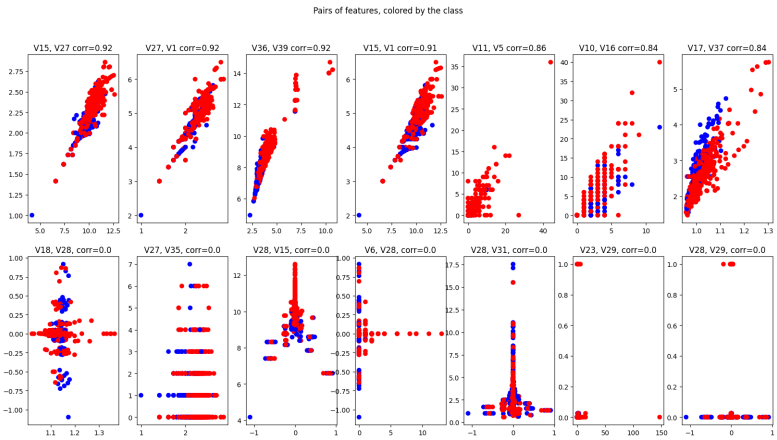


Figure 5: Feature pairs with highest and lowest correlations

3 Modeling

After getting a feeling for the data set, we constructed a couple of models, ranging from linear to nonlinear. First we implemented the majority classifier, so that we could compare the other models to it. All the models were built with the 'scikit-learn' Python library. To get a better feel for the models' success, we plotted the predicted data for each model, using 2 features and predicted class, which we colored by the actual target class.

3.1 Feature selection

Each model was trained and tested on two sets of data - one was the original data set, and the other was a reduced data set, where some features were removed. In attempt to speed up the computation and make the classification easier/better we used a few methods - removing features with high correlation to another features, removing features with low inner variance, PCA dimension reduction and selecting k best features. The first two methods have improved the model accuracy, whereas PCA and best k features were not as successful, so they were removed, since the scores were significantly worse. The methods used removed about 10-15 features from the dataset.

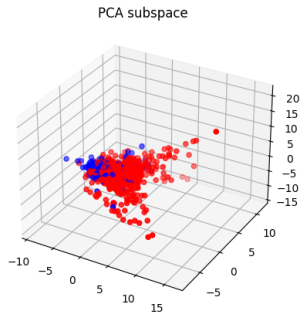
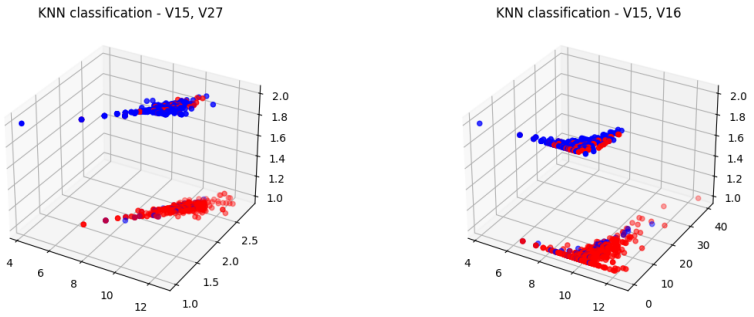


Figure 6: Feature space reduced to 3 dimensions with PCA

3.2 K Nearest Neighbors

The first actual model we implemented was KNN - K Nearest Neighbors. The decision for this was based on the fact that KNN is a relatively simple algorithm that can yield very good results, and that it does not make any assumptions about the data, while at the same time it works well with a high dimensional data. The default value for 'k' is set to 5. With that being the only parameter, and the data not being preprocessed at all, this model achieved accuracy of 75%. After removing highly correlated features and features with low inner variance, that score went up to 79%, and that was further improved by changing the k to be 8. The highest accuracy this model achieved was 86%.

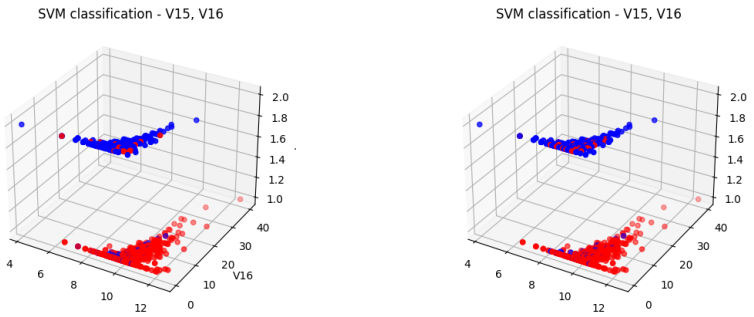


(a) (b)

Figure 7: (a) KNN on original data set; (b) KNN on data set with feature selection

3.3 Support Vector Machine

The second model was SVM - Support Vector Machine. Since it is a powerful model when it comes to linearly separable data and it is not sensitive to high dimensional data, we thought it would yield good results. We tried 3 variants but only 2 were actually useful - SVM with linear kernel and SVM with RBF (nonlinear) kernel. The polynomial kernel performed poorly. The model outperformed the KNN model and reached accuracy of 84% with default parameters (83% with RBF kernel). With feature selection the model scored up to 90%.



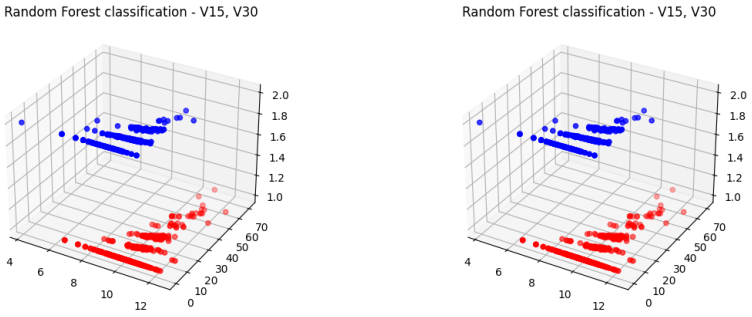
(a) (b)

Figure 8: (a) SVM on original data set; (b) SVM on data set with feature selection

3.4 Random Forest Classifier

Another model that we opted for was the random forest classifier, as it is able to handle imbalanced classes well and also doesn't have a problem dealing with

multiple dimensions, but also because we can use its feature importance scores to determine which features are more important. As before, we left the hyper parameters (mostly) set to defaults as they seemed to be the best option. The results of this model were comparable to the results of the SVM - the average accuracy was about 85% for both sets of data.



(a) (b)

Figure 9: (a) Random forest on original data set; (b) Random forest on data set with feature selection

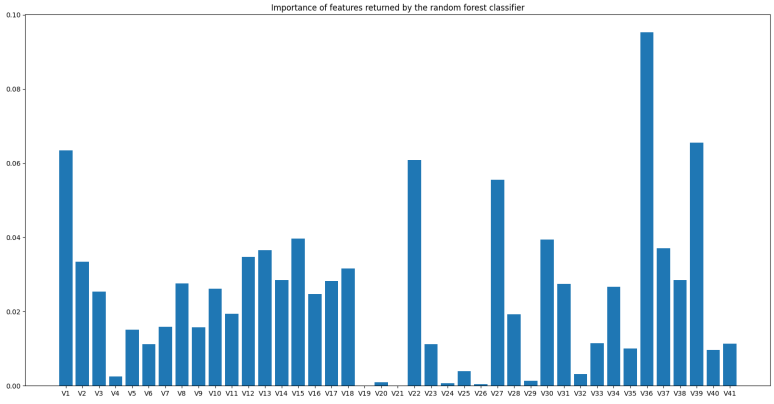
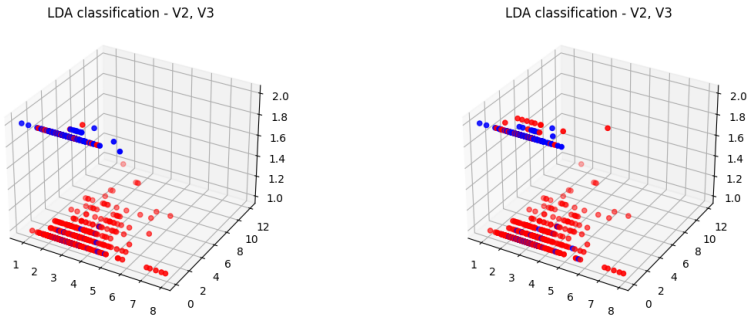


Figure 10: Feature importance scores returned by the random forest classifier

3.5 Linear Discriminant Analysis Classifier

Linear Discriminant Analysis is a useful model when it comes to linear separation, so it was used to try to classify the data. It proved to be a better model than KNN and reached 80% accuracy with the original data set. When using the reduced set, the accuracy reached 82%.

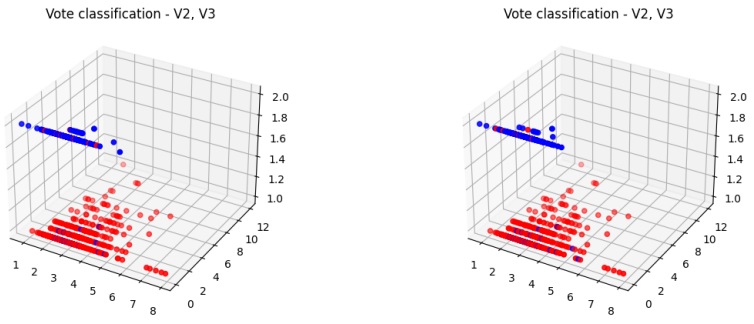


(a) (b)

Figure 11: (a) LDA on original data set; (b) LDA on data set with feature selection

3.6 Voting Classifier

Lastly the models were combined using the voting classifier, which combined the probabilities of all the models above and classified into the appropriate class. This model was one of the best performing ones.



(a) (b)

Figure 12: (a) Vote classifier on original data set; (b) Vote classifier on data set with feature selection

4 Evaluation

4.1 Cross validation and results

To better evaluate the models, each model has been 5-fold cross validated 10 times and the results were averaged.

Each model has been scored using 4 metrics: F1, Precision, Recall and AUC. The results were then averaged and displayed together with the standard deviation of each score. The overall best model was the support vector machine with linear kernel, very closely followed by the random forest classifier.

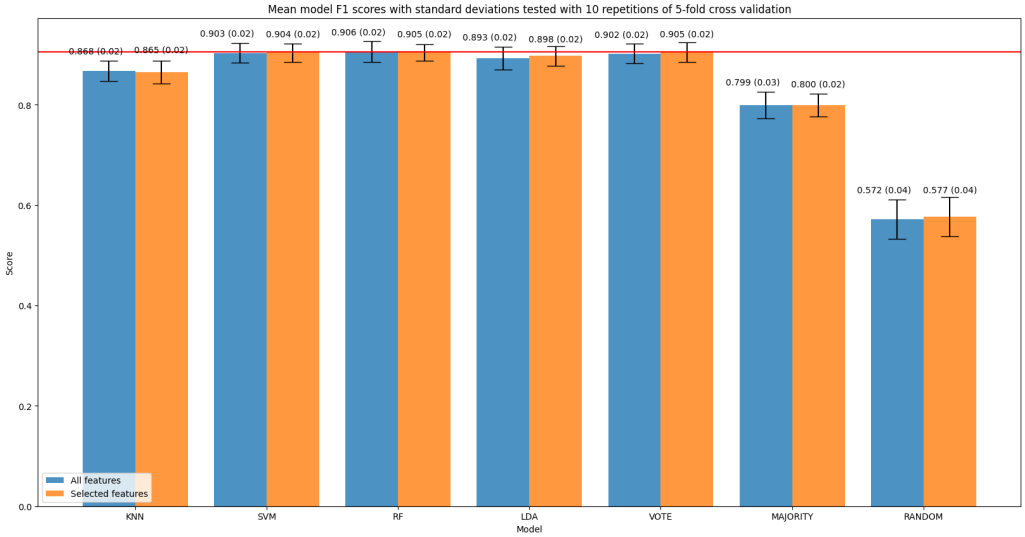


Figure 13: Mean model F1 scores

TODO - compare results, metrics