PROJECT REPORT-Activity prediction for chemical compounds
GROUP NO-20

Introduction:

In this assignment , we were tasked with developing a predictive model using training set of chemical compounds represented by SMILES together with activity label. The features are available in the open source toolkit, RDKit.(1)

Feature selection:

The features are chosen from the packages and sub packages of RDKit API. Then we create a data frame with the input values from training file and the features corresponding to it. Then an initial analysis on the distribution and the type of the data was done. Initially considered these features NumAtoms, HeavyAtomCount, CalcExactMolWt, fr_Al_COO, HsNumAtoms. The correlation matrix is identified for the features and correlated features which are more than 0.9 are eliminated. Two or more than two features are correlated if they are close to each other in linear space. if two or more than two features are mutually correlated, they convey redundant information to the model and hence only one of the correlated features should be retained to reduce the number of features. This would also reduce the training time effectively. Then we added the morgan fingerprints in the form of bit vectors.

Feature distributions:

```
NumAtoms
0.0    10140
1.0     9191
2.0     5956
3.0    13870
4.0     7198
5.0    14496
6.0     6730
7.0    11967
8.0    13163
9.0    11284
dtype: int64
```

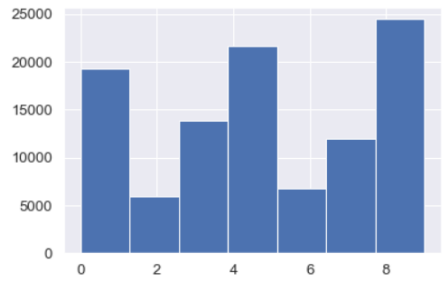

Figure 1. NumAtoms

```
CalcExactMolWt
41.984857      1
60.032363      1
60.068748      1
74.059246      1
74.084398      1
              ..
1389.901796    1
1421.748941    1
1446.434951    1
1619.575870    1
1766.302831    1
Length: 45927, dtype: int64
```

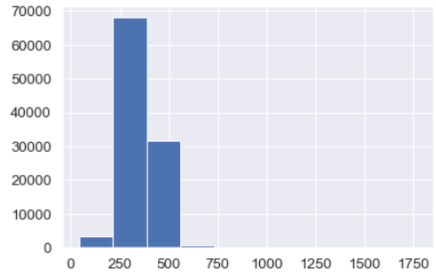

Figure 2. CalcExtractMolWt

```
fr_Al_COO
0    100666
1      3065
2       236
3        17
4        11
dtype: int64
```

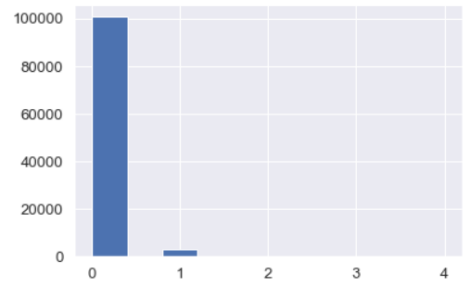

Figure 3.Fr_Al_COO

```
HsNumAtoms
2      1
3      3
7      1
8      3
9      3
      ..
196    2
199    1
203    1
209    1
219    1
Length: 155, dtype: int64
```

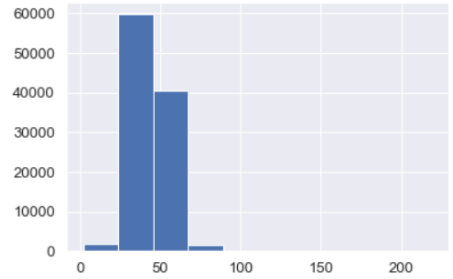

Figure 4. HsNumAtoms

```
ACTIVE
0.0    103700
1.0       295
dtype: int64
```

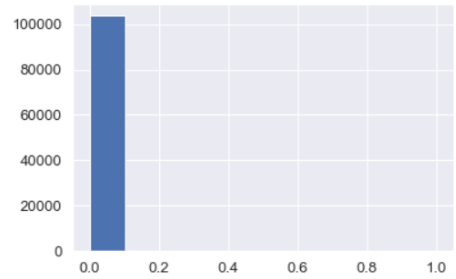

Figure 5. ACTIVE

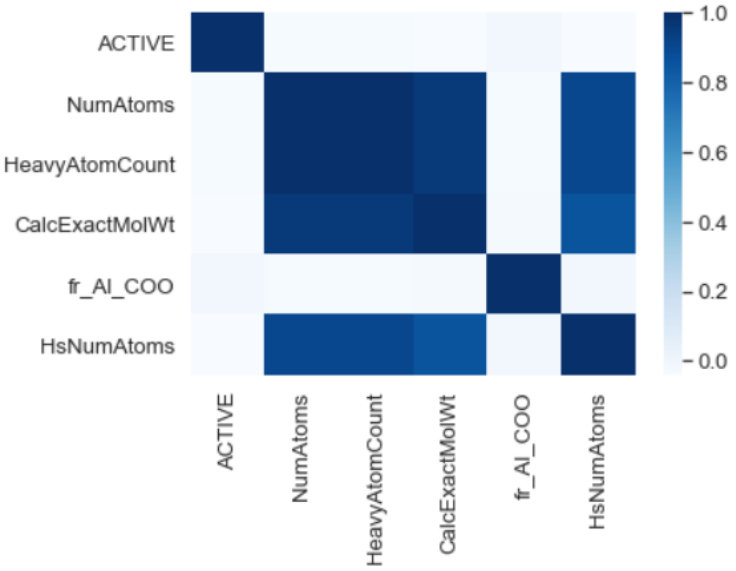

Figure 6. Heat map of correlation

Splitting data:

Then we split the data into training dataset ,validation dataset and holdout dataset. The training data set is 70 % , the validation data set is 21% and the holdout data set is 9% of the input training data. This approach is a pessimistic approach to training the model. The holdout set is not used for training the model and is classified as unseen. The training data set is used for training the model and the validation data set is used during model selection and tuning the hyper parameters. The hold out set is only used at the very end for evaluating the estimated AUC. In order to split the data we use train_test_split function from sklearn module(2). we also do a stratified split so that we preserve the percentage of sample for each class label.

Preprocessing the data:

Then the data is prepared for preprocessing. The non binary features undergo imputation, normalisation and discretisation. We fit with the training data and then apply it on both the training dataset ,the validation dataset and the holdout data set. The min max normalisation is done on the data and the no of bins in discretisation is 10.Since there are multiple attributes and attributes have values on different scales, it may lead to poor data models while performing training. So they are normalised to bring all the attributes on the same scale. We discrete the features to fit the bins and reduce the impact of small fluctuations in data.

Model Building:

The value of the class labels is binary . It is either 0 or 1. Then we train the data using different models using the training data and then predict the AUC using the validation dataset. We determine the AUC of each model classifier. We try to identify the AUC of the models with the fingerprints as well without the finger prints. Then we compare the values. We initially train and determine AUC using the default parameter values of the classifier.

	RandomForest	Decision Tree	KNN
Without FingerPrint Vector	0.7130990774769369	0.7212604657221695	0.7009527211864507
With FingerPrint Vector	0.831973101959128	0.5639992973508549	0.7191387021517643

We can observe from the above table that RandomForest model gives the best AUC with finger print vector features.

Crossvalidation:

We have to do crossvalidation to estimate the skill of the model on new data. It is to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model. Shuffle the training dataset randomly and then we split the data into k=10 groups. One group is chosen as the test

data and the rest of the groups are chosen as the training data. Then we fit the model using training data set and evaluate it with the test set. Then the different performance AUC for the classifiers are noted along with the average AUC. We also evaluate the generalization efficiency of the model. The average AUC stands at 0.7340223012890854 which we felt was pretty good.

Determining Overfitting and Underfitting:

So to determine if there was any overfitting or underfitting we monitor training accuracy and the testing accuracy across various parameter values. There is no significant difference in training accuracy and testing accuracy so there is no overfitting. If the training accuracy is good and testing accuracy is bad it is considered overfitting. It occurs due to low bias and high variance. Under fitting has poor training and testing accuracy. It occurs due to high bias and low variance. Hence we assumed to rule out overfitting and underfitting problems.

Tuning the parameter:

In order to tune the parameter , we tried doing a grid search. Grid search is an automatic way to tune the parameters and the observed the right parameters to apply for training the model. we tried different values of no of trees n_estimators [50,100,150,200] and max_depth [5,10,15,20]. We finally ended up with the best AUC value at parameter values of max_depth=10,n_estimators=200, random_state=0 and class_weight='balanced'. The AUC was 0.8968980803467455.

Over sampling the data:

There was a class imbalance that was observed in the training data. We felt that this might influence the result. So in order to reduce the class imbalancing we can either use under sampling or over sampling. under sampling resulted in loss of data. Hence we applied random oversampling and SMOTE oversampling. Random oversampling involves randomly duplicating examples from the minority class and adding them to the training dataset.SMOTE works by selecting examples that are close in the feature space, drawing a line between the examples in the feature space and drawing a new sample at a point along that line. The Oversampling resulted in slight reduction of AUC to 0.8794996190694242. Also, This may increase the likelihood of overfitting.(3) . Hence we decided not to go with oversampling the data.

AUC of holdout set:

Now the holdout set which we have not used so far, is predicted with the tuned hyper parameters for the classifiers . The AUC is around 0.8758190400861027 which is slightly lesser than the previously predicted AUC of 0.8968980803467455 with the validation data set.

Final predictions:

Then the model is trained with the tuned hyper parameters with the training dataset and then it is used to predict the data set in the test file whose activity is to be predicted. We calculate the probability of prediction of 1.0 as the activity label and then export the result to the text file.

References

1. https://www.rdkit.org/
2. https://scikit-learn.org/stable/index.html
3. survey of predictive modelling under imbalanced distributions https://arxiv.org/abs/1505.01658