LEARNING SYSTEMS PROJECT REPORT

Ajmal Hussain

Shankara Narayanan Bangalore Ramalingam

Introduction:

Regression:

1.1 Estimate cetane number of Diesel fuel

In this task we are told to predict the cetane number of the diesel fuel, the dataset consist of 3 matrices and they were the following cnTrainX, cnTrainY and cnTestX . The Dataset consisted of 401features and 245 data points, out of which 133 were for training and validation and rest 112 were test data.

cnTrainX: Contains IR spectrum for each sample.

cnTrainY: Output value cetane number of each diesel fuel.

CnTestX: Contains the IR Spectrum of each sample.

1.2 Modelling the need for cooling in a H2O2 process

In this task we are asked to make a construct a model with all the variables or some of it to model the valve opening. Like the previous dataset is a "mat" file which is loaded using loadmat() and then converted into pandas.

The dataset contains 65 features and has more than 7000 datapoints and like previous the data has 3 matrices, it has training (XtrainDS,YtrainDS) and test(XtestDS) data. This dataset also contains time data which is present in the first column, and this should be ignored as a feature.

1.3 Predicting power load

This is a non-linear regression task which is like previous 1.2 task. The main goal of the task is to predict the power load of Puget Sound Power & Light Co. the following day at 8 in the morning. While creating the model we had to consider the data from different seasons. The dataset consists of 15 features, which were in 3 matrices and this were powerTrainInput, powerTrainOutput and powerTestInput. The dataset also consists of a powerTrainDate columns in it.

Classification

1.4 Thyroid disease (Classification)

In this task we have to check that a person is normal. hypothyroid or hyperthyroid based on some given particular measurements. This is classification task because we have to predict discrete value rather than continues values. The total number of observations is 7200 with 21 features. There is only 5000 samples available for train the model and rest of 2200 will be used for testing.

1.5. Breast cancer (Classification)

In This task we have to train a model that will predict that a patient has benign or malignant breast cancer base on some features that are collected from a Fine Needle Aspiration (FNA). The diagnosis test is done in the following way: The breast mass is taken by FNA and then it scaled on microscope slide. After this its highlight the cellular nuclei .Well differentiated cells are scanned by digital camera and a frame grabber board. . The total number of observations are 569 samples with 30 features there is a cancerWTrain.mat file which containing three metrices cancerTrainX (30×400), cancerTrainY (1×400), and cancerTestX (30×169).the out put has only 1=malign or 0 =benign

1.6. Electrocardiograms (Classification)

In This task we have to predict that a patient suffer from transmural ischemia or not based on some given ECG data. ECG has ten different channels. Each channel has 26 feature it means every subject has 26 x 12= 312 features . A lot of information given by channels so we have to extract the important features like 19–26 for each channel. These correspond to inputs 19–26, 45– 52, 71–78,and so on.

The total number of observations is 300.the 200 samples will be used for training the model and rest of 100 observation will be used for testing . The file ECGITtrain.mat contain three metrices which contain the s inputECGITtrain (200 × 312), outputECGITtrain (200 × 1, i.e. a column vector), and inputECGITtest (100 × 312).

State-of-the-art (Regression)

Regression in one of the relevant problems in machine learning, this is because there are many approaches. The current work represents a comparison of a large amount of popular machine learning algorithms, which number to a total of 77, which belong to 19 families.

The main objective of the paper is to provide a "road-map" for researchers to who want to solve regression problems. In machine learning we are trying to predict the next output in the data. The topic is widely studied by statisticians, and they provide different approaches to the problem. Linear regression, ridge regression, multivariable adaptive regression splines, least and partial least squares regression. Now there are many more algorithms which are being developed and called as universal algorithm, wherein they can be used in both regression and classification, the example as follows neural networks, support vector machines, regression trees and rules, bagging and boosting ensembles, random forests etc.

The current work group is quantitative comparison of very large collection of regression techniques which are intended for the to provide the reader with 1. A list of currently available regression algorithms and the families they are classified into 2. A brief description of each of the algorithms and the list of references for each approach along with the technical details, such as hyperparameter tuning list and recommended values. 3. The rank of each model available based on the performance peed, identifying the best performing approach and the performance level which can be expected for it. 4. The code to run all the regression models compared in the study.

State-of-the-art (Classification): Supervised classification is one of the tasks most frequently used by so called machine learning. There are several techniques have been developed based on machine learning. Supervised learning mainly used to develop a concise model of the class labels in terms of predict output. The classifier is used to give class label to the testing sample where the value of predictor feature is given but the class label is unknown i.e. output. There are different number of applications for machine learning the most preferable of which is predictive data mining. The machine learning algorithm used the same feature for every sample in datasets. The feature may be continuing, categorical, or binary. If the observations are provided with known labels then the learning is called supervised learning because in supervised learning, we have some label data from train our model. In other hand where the data is given without label class it refers to unsupervised learning. Numerous ML applications involve tasks that can be set up as supervised. In this task we will concentrate on techniques that are necessary to perform this task. This work is link to the classification problem in which the label class has only discrete unsorted value. After understanding the strength and limitations of each method the possibility of two or more algorithms together to solve a problem should be evaluated. The objective is to calculate the accuracy of one algorithm complement the weakness of other algorithms. If we used only one algorithm to find the best model it might be strange that how can we compare the model because if we want to compare our model, we should have another 1 or more model so we can make comparison between them. We will use the different algorithm for train the best classifier that will predict the accurate class label on given sample. We will use K-nearest neighbors (KNN), Artificial Neural Network (ANN) and Decision Tree (DT). After completing each task there will be one classifier that will have the best accuracy as compare to the other algorithms.

General issues of supervised learning algorithms

To be very first in supervised learning we must collect the datasets. If an expert is available, then he/she could suggest that which features are the most important as in task 1.6 you mentioned that features 19–26 for each channel is important. These correspond to inputs 19–26, 45– 52, 71–78, and so on. It can affect the accuracy of classifier. If not, then the simplest method is that of "brute-force," which means measuring everything available in the hope that the right (informative, relevant) features can be isolated. However, we dint need brute-force because we have no missing value. After getting the data there is may be need to normalization of the data because if they have much difference then it may affect our model, so we are using standard scaler for data preprocessing. After this We have to choose the algorithm.

Methodology:

Regression:

The goal of the regression task is to find the best model, which will be useful to predict the next best outcome from the dataset, this is done by calculating the best minimum mean squared error and root mean squared error. The following methods were used in the project to solve the problem

Pandas: Pandas is a fast, flexible and open source python library which is very useful for data analysis and manipulation. In this project we have used pandas due to its advantages in data representation.

The dataset is loaded into a Data frame, which were split into Training, and test Data.

Feature Selection: Feature selection is one of the core concepts of Machine Learning, which will help to increase the overall performance of the model. The feature selection is used here to remove the features which do affect the overall mean squared error of the model. We do this by presenting the score of each feature.

Train_test_split: We are using the train_test_split function from the scikit learn library to split the data into train and validation, this is done by splitting the training data provided in the dataset.

StandardScaler: The standard scaling is a common practice in machine learning models, we are using the standard scaler in order to prevent the data from misbehaving which will result in bad models (or not accurate ones.)

KNN Regression: KNN is a regression algorithm which is based on K-nearest neighbors. The target is predicted by local interpolation of the targets associated of the nearest neighbor in the training set.

Hyperparameter tuning: Parameters (K)

Linear Regression: The linear regression model is one of the most used models in machine learning, and it is also the most basic. This model provides good performance for most linear data problems.

RandomForest regressor: RandomForest is ensemble method which is used in regression, with the use of multiple decision trees and technique called Bootstrap Aggregation, commonly known as bagging.

Hyperparameter tuning: Parameters (n_estimators, max_depth min_samples_split, min_samples_leaf)

Lasso Regression: Lasso regression is a type of linear regression that uses shrinkage. Shrinkage is where data values are shrunk towards a central point, like the mean.

Hyperparameter Tuning: Parameters ('alpha')

Ridge regression: This model solves a regression model where the loss function is the linear least squares function and regularization is given by the l2-norm. Also known as Ridge Regression or Tikhonov regularization.

Hyperparameter Tuning: Parameters ('alpha')

Support Vector regressor: Support Vector regression is a type of Support vector machine that supports linear and non-linear regression.

Hyperparameter Tuning: Parameters (Kernel=[linear,poly,rbf,sgz], param_grid=[C,gamma],cvFold=n)

MLP Regressor: It implements a multi-layer perceptron (MLP) that trains using backpropagation with no activation function in the output layer, which can also be using the identity function as activation function. Therefore, it uses the square error as the loss function, and the output is a set of continuous values.

Hyperparameters Tuning: Parameters (hidden_layer_sizes, activation, solver, shuffle, random_state, max_iter, momentum, early_stopping, validation_fraction)

Classification:

We used different method for training the model because there is no guarantee that 1 method will give you the best model. We are using the Following method to train our best model:

Loading datasets

Loading data from the mat file Xtrain, ytain, Xtest

Data Preprocessing

Used standard scalar for data preprocessing if needed .

Principal component analysis (PCA).

We are using the PCA for the visualization of data. PCA used to reduce the dimensionality of data. Principal component analysis (PCA) is a technique used to emphasize variation and bring out strong patterns in a dataset. It's often used to make data easy to explore and visualize.

import sklearn library for splitting the data in train and testing part

k-nearest neighbors algorithm (k-NN)

KNN is an algorithm that is used in classification. In KNN An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors.

- **Hyper parameter tuning (**K**)**
- **Train the model according to best parameter**
- **Calculate the accuracy**
- **Predict the test data**

Artificial neural network

Artificial neural network is a machine learning technique used for classification problems. ANN is a set of connected input output network in which weight is associated with. each connection. It consists of one input layer, one or more intermediate layer and. one output layer.

- **Hyper parameter tuning (**Hidden layer size , number of neurons**)**
- **Train the model according to best parameter**
- **Calculate the accuracy**
- **Predict the test data**

Decision Tree

A Decision Tree is a simple representation for classifying examples. It is a Supervised Machine Learning where the data is continuously split according to a certain parameter.**we have to give the parameters maximum depth minimum leaf sample to train our model**

- **Hyper parameter tuning (max depth , mini sample leaf)**
- **Train the model according to best parameter**
- **Calculate the accuracy**
- **Predict the test data**

Selection of Classifier

After train our model with different algorithm we will select our best model one of them which has best accuracy and then predict the test data

Data:

1.1 Estimate cetane number of Diesel fuel

The first dataset is split into 3 matrices as mentioned in the description, the dataset is loaded into a Pandas data frame, the cnTrainX had the dimension initially (401 rows × 133 columns) and cnTrainY had initial dimensions of (1 rows × 133 columns). In this we had to take the transpose of the dataset, in this case both cnTrainX and cnTrainY to prevent the shape error from occurring, while performing future operations. We even had to take the transpose of the cnTestX, as it will cause parity while trying to predict the data with the test data.

1.2 Modelling the need for cooling in a H2O2 process

In this dataset the initial dimension was 4466 rows × 65 columns for the XtrainDS dataset, YtrainDS also had a dimension of 4466 rows × 1 columns. In this dataset it was mentioned that we had to drop the first column of the dataset, which was the time column. This operation was performed in the XtrainDS and XtestDS dataset. As this dataset had to shape issues while working on it, there was no need to reshape or transpose the dataset.

1.3 Predicting power load

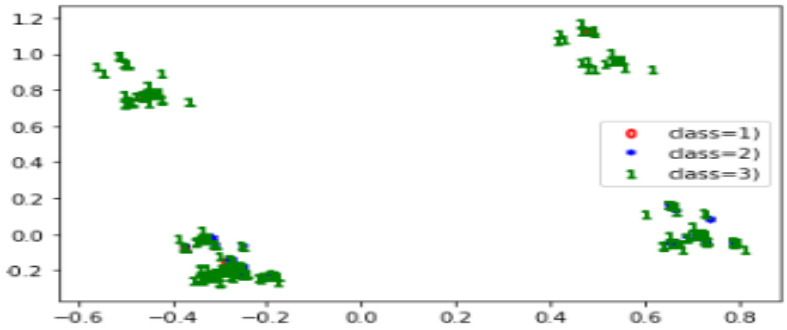
In this dataset we are using the powerTrainInput, powerTestInput and powerTrainOutput matrices, and in this dataset, there was the need for transpose as there was problem which stated that the datapoints was inconsistent, we had removed this error by taking the transpose of powerTrainInput, powerTrainOutput.

1.4 Thyroid disease

A thyroidTrain.mat file is given which has 3 different matrices. its included trainThyroidinput which contain 5000 obesrvations and 21 feature and second matrix trainThyroidOutput which contain 5000 rows and 3 column i.e out put of the observations in the form of (1,0,0), (0,1,0), or (0,0,1) Here you can see the data and data shape we don't need data preprocessing in this task

```
trainThyroidInput shape (5000, 21)
trainThyroidOutput shape (5000, 3)

small subset trainThyroidInput

[[0.28 0.   1.   0.   0.   0.   1.   0.   0.   0.   0.   0.   0.   0.   ]
 [0.45 0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   ]
 [0.77 0.   0.   0.   0.   0.   0.   0.   1.   1.   0.   0.   0.   0.   ]
 [0.72 0.   1.   0.   1.   0.   0.   0.   0.   0.   0.   0.   0.   0.   ]
 [0.78 0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   ]
 [0.74 0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   ]
 [0.48 0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   ]
 [0.51 0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   ]
 [0.45 1.   0.   0.   0.   1.   0.   0.   0.   0.   0.   0.   0.   0.   ]
 [0.46 1.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   0.   ]]

small subset of trainThyroidOutput

[[0 0 1]
 [0 0 1]
 [0 0 1]
 [0 0 1]
 [0 0 1]
 [0 0 1]
 [0 1 0]
 [0 0 1]
 [0 0 1]
 [0 0 1]]
```

We used PCA for the data visualization following is the scatter plot of the data



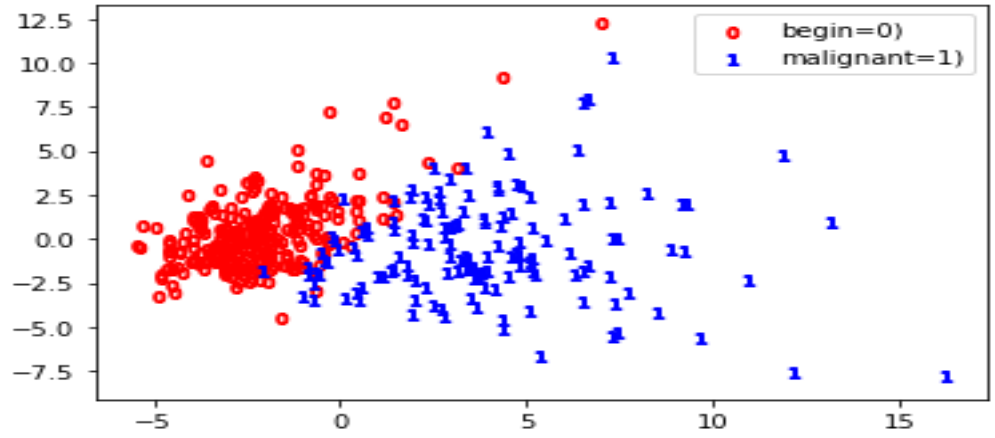
1.5 Breast Cancer: The total number of observations are 569 samples with 30 featuresthere is a cancerWTrain.mat file which containing three metrices cancerTrainX (30×400), cancerTrainY (1×400), and cancerTestX (30×169). The output has only 1=malign or 0 =benign

```
(30, 400)
[[1.453e+01 1.136e+01 1.768e+01 1.902e+01 2.048e+01]
 [1.398e+01 1.757e+01 2.074e+01 2.459e+01 2.146e+01]
 [9.386e+01 7.249e+01 1.174e+02 1.220e+02 1.325e+02]
 [6.442e+02 3.998e+02 9.637e+02 1.076e+03 1.306e+03]
 [1.099e-01 8.858e-02 1.115e-01 9.029e-02 8.355e-02]]
(1, 400)
[[0 0 1 1 1]]
```

Now we have to take the transpose of cancerTrainX and cancerTrainY then we also used to standard scaler for data preprocessing you can see the following output

```
(400, 30)
[[ 0.11460184 -1.24395724  0.07895931 -0.03005724  0.96753454]
 [-0.80287855 -0.39483967 -0.81669104 -0.72915924 -0.52140722]
 [ 1.02629371  0.35493824  1.06555778  0.88386702  1.07927501]
 [ 1.41412453  1.26555179  1.25835102  1.20509924 -0.40198459]
 [ 1.83668648  0.5252348   1.69842255  1.86301029 -0.87269132]]
(400,)
[0 0 1 1 1]
```

Now apply PCA for visualization here you can see the scatter plot of this data with PCA n_compunents=2



1.6 Electrocardiograms

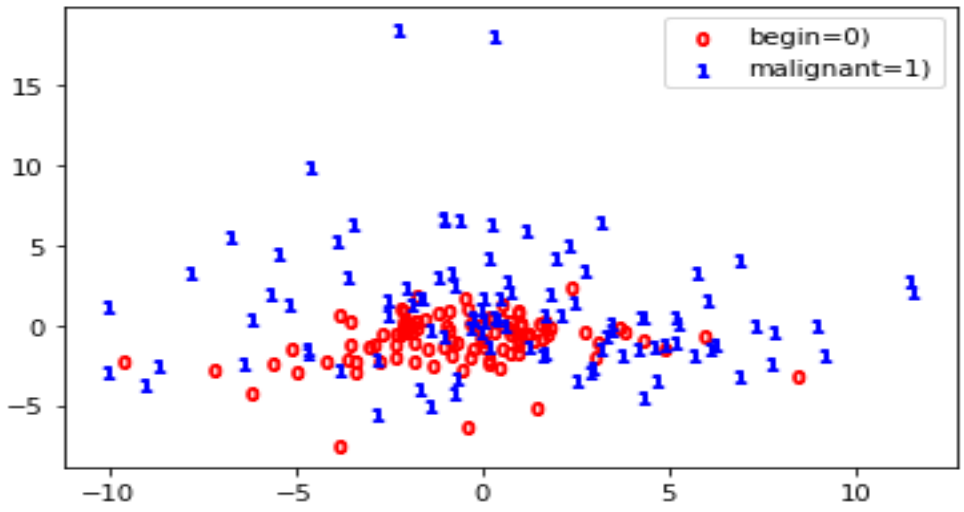
The total number of observations is 300.the 200 samples will be used for training the model and rest of 100 observation will be used for testing . The file ECGITtrain.mat contain three metrices which contain the s inputECGITtrain (200 × 312), outputECGITtrain (200 × 1, i.e. a column vector), and inputECGITtest (100 × 312).

```
(200, 312)
(200, 1)
[[ 0.         0.         0.         1.2637258  9.2601681]
 [14.910554   0.         0.         0.         0.       ]
 [ 1.0045998 13.89223    9.5149614  5.         4.2430653]
 [ 0.         8.1978235  0.         0.         0.       ]
 [ 0.         2.9922511  6.0993438  7.        16.430363 ]]
[1 0 0 0 0]
```

Now we have to extract the most important feature from the whole 312 feature. we will only extract 84 most important feature and perform standard scalar for because the values has much difference.following is the out put

```
(200, 84)
[[ 1.36779444  0.56835858 -0.58605598 -0.62999496 -0.63417844]
 [-0.98347552  0.38353729  0.57107342  3.01116756  2.51055967]
 [ 0.55362493 -0.75345378 -0.58605598 -0.62999496 -0.63417844]
 [-0.98347552  0.20257712  0.36283601  1.11921917  1.25660321]
 [ 0.48690019  1.00887809  0.92016061  2.20314582 -0.63417844]]
```

Now we will apply PCA for visualization .We used PCA n_compunents=2 and draw the scatter plot .you can see the output below



Results and their interpretation:

1.1 Estimate cetane number of Diesel fuel For all the regression models we have subjected them to 5 major regression algorithms, for the first algorithm we have used KNN regressor, Linear Regression, Random Forest regression, Lasso and Ridge regression. In this model we have found the MSE (mean squared error) and RMSE (root mean squared error) for all the models, some of the results are displayed below

```
regressor=KNeighborsRegressor(n_neighbors=10)
regressor.fit(X_train,y_train)
y_pred=regressor.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 1.6754999999999995
Mean Squared Error: 4.191834999999997
Root Mean Squared Error: 2.0473971280628476
```

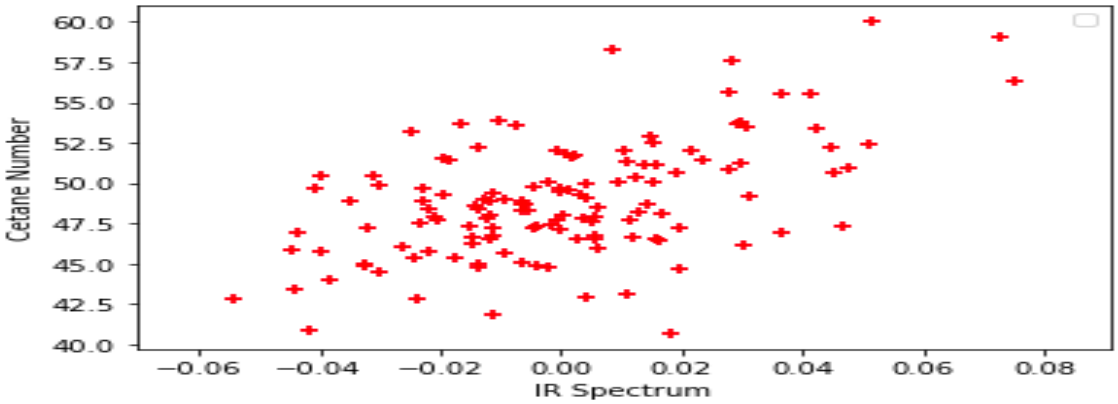
```
#Selecting the best algorithm
regressor=Ridge()
regressor.fit(X_train,y_train)
y_pred=regressor.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

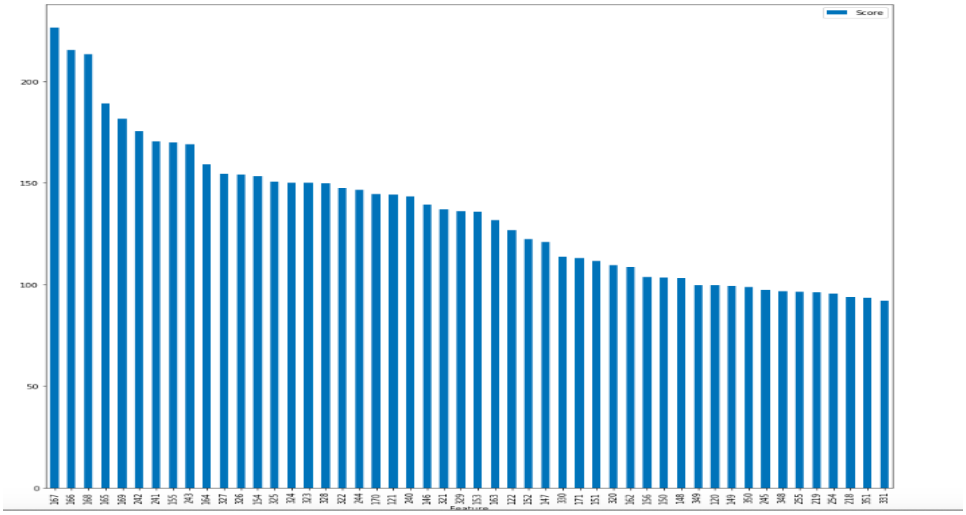
```
Mean Absolute Error: 1.6191600580998355
Mean Squared Error: 3.9419961647439608
Root Mean Squared Error: 1.9854460870907478
```

```
regressor=LinearRegression()
regressor.fit(X_train,y_train)
y_pred=regressor.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 1.8411918633482447
Mean Squared Error: 5.166192076608688
Root Mean Squared Error: 2.2729258845392843
```

The Datapoints for the first Dataset has been plotted, and also the most important features in the dataset has been plotted, since the dataset is has a lot of features, we are only taking the top





1.2 Modelling the need for cooling in a H2O2 process

For this dataset we have used 5 regression models again, which are namely KNN regressor, Linear Regression, Random Forest regression, Lasso and SVR. In this model we have found the MSE (mean squared error) and RMSE (root mean squared error) for all the models, some of the results are displayed below.

```
regressor=LinearRegression()
regressor.fit(X_train,y_train)
y_pred=regressor.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 6.0361746196131
Mean Squared Error: 77.94528410761411
Root Mean Squared Error: 8.82866264547548
```

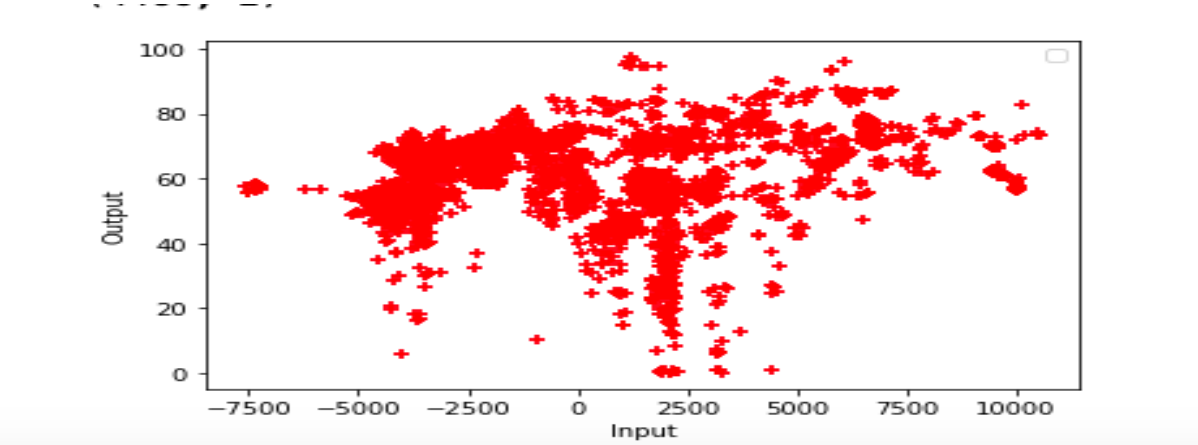
```
from sklearn import metrics
regressor=RandomForestRegressor(n_estimators = 350, random_state = 0)
regressor.fit(X_train,y_train)
y_pred=regressor.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 2.791439765458427
Mean Squared Error: 25.191203841532147
Root Mean Squared Error: 5.019083964383555
```

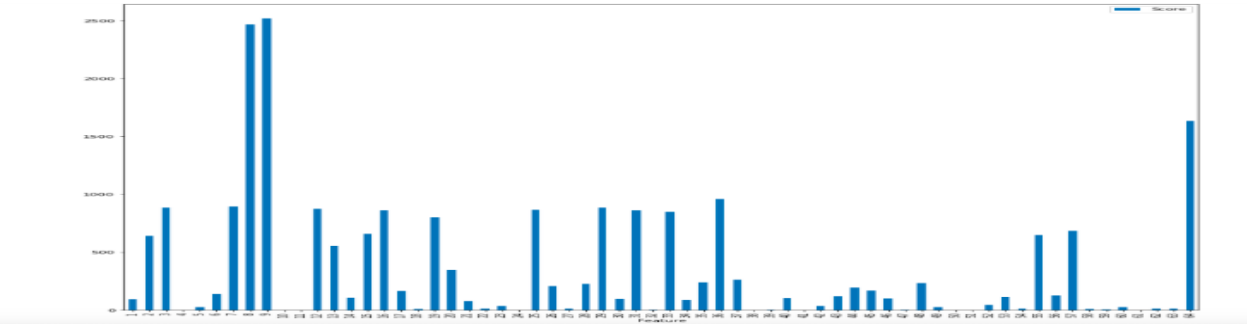
```
from sklearn import metrics
regressor=KNeighborsRegressor(n_neighbors=10)
regressor.fit(X_train,y_train)
y_pred=regressor.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 2.818876865671642
Mean Squared Error: 25.34048851679105
Root Mean Squared Error: 5.033933702065518
```

The Datapoints for the above dataset is given by



Feature score graph:

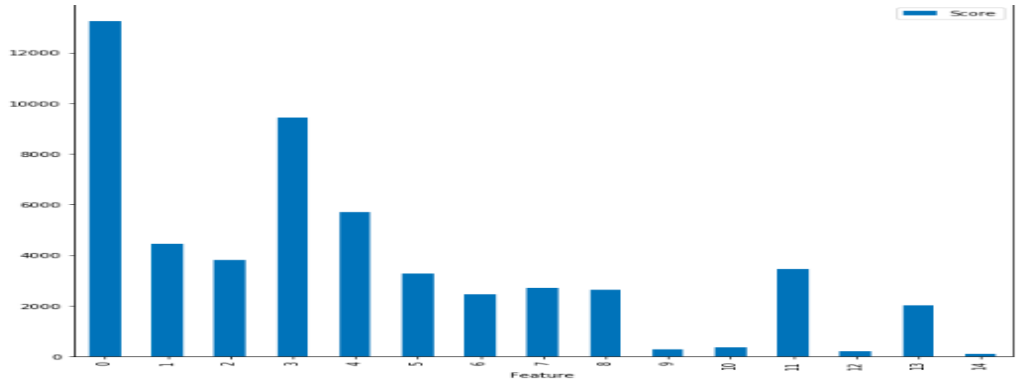


1.3 Predicting power load

For this dataset we have used the following models in the code and cross validation error score have been used to MLP regressor algorithm. The root mean squared error and the feature is given below

```
from sklearn import metrics
regressor=MLPRegressor(hidden_layer_sizes=(500, 200), solver="lbfgs", activation="relu")
regressor.fit(X_train,y_train)
y_pred=regressor.predict(X_test)
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absolute Error: 54.15154064129579
Mean Squared Error: 4769.057669365028
Root Mean Squared Error: 69.0583642245096
```



1.4 Thyroid disease

Artificial Neural Networks

To be very first I implemented Artificial Neural Network by using sklearn library then I split the data and train the model with different parameters and find the optimal best model

```
Accuracy with: 1 0.9666666666666667
Accuracy with: 2 0.9726666666666667
Accuracy with: 3 0.96
Accuracy with: 4 0.9526666666666667
The Best model is : (0.9726666666666667, MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
              beta_2=0.999, early_stopping=False, epsilon=1e-08,
              hidden_layer_sizes=(10, 10, 10), learning_rate='constant',
              learning_rate_init=0.001, max_iter=500, momentum=0.9,
              n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5,
              random_state=None, shuffle=True, solver='adam', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False))
```

Now we will train the model according to higher accuracy and predict the test data which is without class label you can see the subset output of test data following:

KNN Results

Now we implemented the KNN algorithm with different k values and find he optimal value of k with best accuracy you can see in following pic

```
accuracy when K = 1
0.91
accuracy when K = 3
0.92
accuracy when K = 5
0.9226666666666666
accuracy when K = 7
0.9193333333333333
accuracy when K = 9
0.916
accuracy when K = 11
0.9133333333333333
accuracy when K = 13
0.9113333333333333
accuracy when K = 15
0.9106666666666666
accuracy when K = 17
0.91
accuracy when K = 19
0.91
maximum accuracy  0.9226666666666666
optimal K value  5
```

Decision Tree

Now we implemented Decision tree with different parameters value like max depth and min sample leaf and find the optimal values and model following you can see the output of decision tree

```
accuracy when max and min : 3 5
accuracy with model 1 0.9906666666666667
accuracy when max and min : 10 4
accuracy with model 2 0.9906666666666667
accuracy when max and min : 10 5
accuracy with model 3 0.9906666666666667
accuracy when max and min : 12 1
accuracy with model 4 0.996
accuracy when max and min : 12 6
accuracy with model 5 0.9906666666666667

best accuracy with model 4 0.996

DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=12,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=100, splitter='best')
```

1.5 Breast cancer

To be very first I reshape the data with transpose and then apply stand scalar for preprocessing and then used PCA for visualization

KNN

We implemented KNN with gridsearchcv and find the optimal value of k and best accuracy score grid search cv showing the following result

```
Best Accuracy  0.9458333333333333
{'n_neighbors': 5}
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                     metric_params=None, n_jobs=None, n_neighbors=5, p=2,
                     weights='uniform')
optimal k=  5
```

ANN

We implemented KNN with gridsearchcv and find the optimal hidden layer size best accuracy score grid search cv showing the following result

```
Best parameters found:
 {'activation': 'relu', 'alpha': 0.0001, 'hidden_layer_sizes': (100,), 'learning_rate': 'constant', 'solver': 'adam'}
[1 1 0 0 1 0 0 0 0 0]
0.98125
```

Decision Tree

we implemented the decision tree with changing the value of max dept and min sample leaf to get the optimal value and best accuracy score

```
[1 1 0 0 1 0 0 0 0 1]
accuracy when max and min value: 3 5
0.90625
[1 1 0 0 1 0 0 0 0 0]
accuracy when max and min value: 10 4
0.925
[1 1 0 0 1 0 0 0 0 0]
accuracy when max and min value: 10 5
0.925
[1 1 0 0 1 0 0 0 0 1]
accuracy when max and min value: 2 2
0.90625
[1 1 0 0 1 0 0 0 0 0]
accuracy when max and min value: 12 1
0.9125
Best accuracy with parameters  0.925 10 5

DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=10,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=5, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=100, splitter='best')
```

1.6 Electrocardiograms

KNN

Now we implemented the KNN algorithm with different k values and find he optimal value of k with best accuracy you can see in following pic

```
accuracy when n = 1
Accuary: 0.775
accuracy when n = 3
Accuary: 0.8
accuracy when n = 5
Accuary: 0.825
accuracy when n = 7
Accuary: 0.775
accuracy when n = 9
Accuary: 0.825
accuracy when n = 11
Accuary: 0.8
accuracy when n = 13
Accuary: 0.8
accuracy when n = 15
Accuary: 0.825
accuracy when n = 17
Accuary: 0.825
accuracy when n = 19
Accuary: 0.825
0.825 19
```

Decision Tree

we implemented the decision tree with changing the value of max dept and min sample leaf to get the optimal value and best accuracy score

```
[0 0 1 1 0 1 0 0 0 0]
accuracy when max and min value: 3 5
0.75
[0 0 1 1 0 1 0 0 0 0]
accuracy when max and min value: 10 4
0.65
[0 0 1 1 0 1 0 0 0 0]
accuracy when max and min value: 10 5
0.625
[0 0 1 1 0 1 0 0 0 0]
accuracy when max and min value: 2 2
0.8
[1 0 0 1 0 1 0 0 0 0]
accuracy when max and min value: 12 1
0.75
best accuracy with optimal parameter 0.8 2 2
```

ANN

We implemented ANN and find the best parameters and best accuracy you can see the following:

```
Accuracy with model: 1 0.725
Accuracy with model: 2 0.75
Accuracy with model: 3 0.675
Accuracy with model: 4 0.75
The Best model is : (0.75, MLPClassifier(activation='relu', alpha=0.0001, batch_size='auto', beta_1=0.9,
              beta_2=0.999, early_stopping=False, epsilon=1e-08,
              hidden_layer_sizes=(50, 80, 100), learning_rate='constant',
              learning_rate_init=0.001, max_iter=500, momentum=0.9,
              n_iter_no_change=10, nesterovs_momentum=True, power_t=0.5,
              random_state=None, shuffle=True, solver='adam', tol=0.0001,
              validation_fraction=0.1, verbose=False, warm_start=False))
```

Discussion:

1.1 Estimate cetane number of Diesel fuel

For this dataset we can notice that the Lasso regression algorithm, provides the least root mean squared error, compared to all the models. Hence this is the best, compared to the rest.

1.2 Modelling the need for cooling in a H2O2 process

For this dataset we can notice that the Random Forrest regression algorithm, provides the least root mean squared error, compared to all the models. Hence this is the best, compared to the rest.

1.3 Predicting power load

For this dataset we can notice that the MLP regression algorithm, provides the least root mean squared error, compared to all the models. Hence this is the best, compared to the rest.

we implemented the three different algorithm and apply on 3 different classification task it gives different result on different data .When we change the value of hyper parameter it does effect on the accuracy of algorithm so we find the best 3 classifies for 3 task following is the output of final test data

1.4 Thyroid disease

We implement three algorithm and find one best model in decision tree with 99 accuracy and following is the test data prediction

```
2200
[[0 0 1]
 [0 0 1]
 [0 0 1]
 ...
 [0 1 0]
 [0 0 1]
 [1 0 0]]
```

1.5 Breast cancer

In This task we also implemented the same algorithm and find the best classifier i.e ANN the output of final test data is following

```
[0 0 1 0 0 1 0 0 1 0 1 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1 1 1 1 1 1 0
 0 0 0 1 1 1 1 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 1 1 1 0 1 0 0 0 1 0 1 1 0 0 1
 0 1 0 0 0 1 1 0 1 1 0 1 1 1 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1
 1 1 0 0 0 0 0 1 0 1 0 1 1 1 1 1 0 1 0 1 1 1 0 1 1 0 0 0 0 0 1 0 0 0 1 1 0
 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 1 1 0]
169
```

1.6 Electrocardiograms

In This task we also implemented the same algorithm and find the best classifier i.e ANN the output of final test data is following

```
[0 0 1 0 0 0 1 0 0 1 0 0 1 0 0 0 0 1 0 1 1 0 1 0 0 1 0 0 0 0 0 1 1 1 0 1 0
 1 0 0 0 0 0 0 0 0 1 0 0 1 0 1 1 1 0 1 1 1 0 1 0 0 1 1 0 0 1 0 1 1 1 1 0 0
 0 0 1 0 0 1 1 0 1 1 1 1 0 0 0 0 1 1 0 0 0 1 0 0 0 0]
100
Accuracy 0.825
```

References

M. Fernandez-Delgado M.S.Sirsat , E.Cernadas, S.Alawadi ,S.Barro , M.Febrero-bande .An extensive experimental survey of Regression method , University of Santiago de Compostela, Campus Vida, 15782, Santiago de Compostela, Spain

S. B. Kotsiantis · I. D. Zaharakis · P. E. Pintelas . Machine learning: a review of classification and combining techniques. Springer Science+Business Media B.V. 2007