Implementation of Machine Learning Algorithms for Prediction of Diabetes

Submitted as part of the assignment
In

CE802 Machine Learning
Submitted to

Dr. Vito De Feo
By

Tamanna
(PG21154802)

Comparative Study (910)

School of Computer Science and Electronic Engineering University of Essex
January 2022

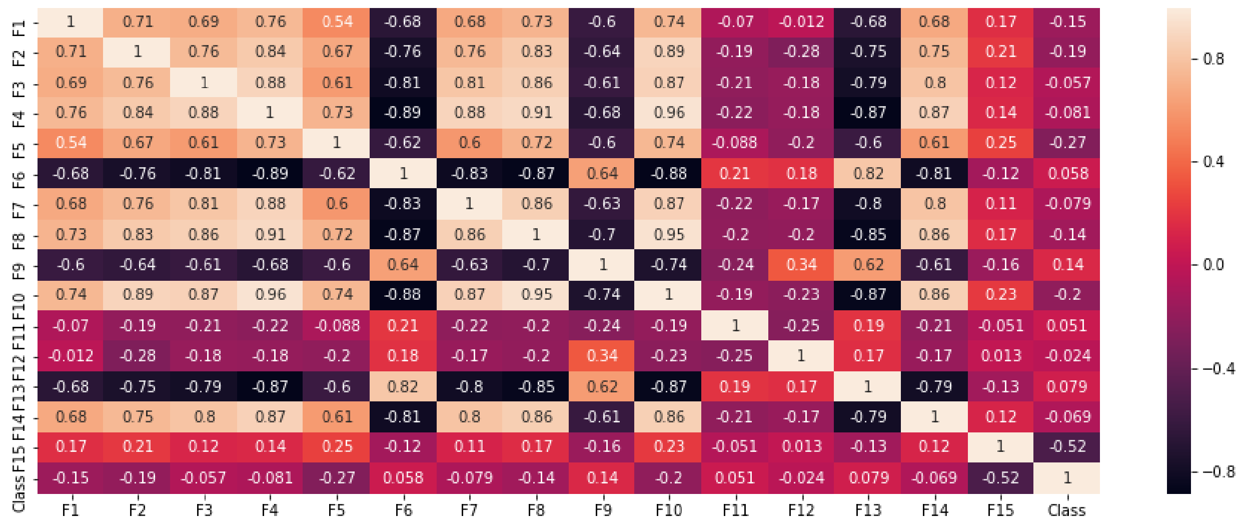
TABLE OF CONTENTS

2. Comparative Study- Task a

The data that is provided by the NHS, has 15 features that are associated with it. There are 1500 rows of data that has been provided for these features. Each feature is unique and represents a physical parameter that is used to test whether the patient or the person in question is affected by diabetes. In pilot case study we have used decision trees, k nearest neighbors (KNN), Xg boost, and finally prediction on a hold-out set.

2.1 Data Pre-Processing

First we have pre-processed the data to identify the null values. On identifying the null values, those null values have been replaced by mean value and used in its place. Since the dataset has a considerable number of features, it might be useful to do feature selection to reduce the high dimensionality of the dataset and reduce training and see if the prediction error on the testing set reduces.



	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	Class
F1	1	0.71	0.69	0.76	0.54	-0.68	0.68	0.73	-0.6	0.74	-0.07	-0.012	-0.68	0.68	0.17	-0.15
F2	0.71	1	0.76	0.84	0.67	-0.76	0.76	0.83	-0.64	0.89	-0.19	-0.28	-0.75	0.75	0.21	-0.19
F3	0.69	0.76	1	0.88	0.61	-0.81	0.81	0.86	-0.61	0.87	-0.21	-0.18	-0.79	0.8	0.12	-0.057
F4	0.76	0.84	0.88	1	0.73	-0.89	0.88	0.91	-0.68	0.96	-0.22	-0.18	-0.87	0.87	0.14	-0.081
F5	0.54	0.67	0.61	0.73	1	-0.62	0.6	0.72	-0.6	0.74	-0.088	-0.2	-0.6	0.61	0.25	-0.27
F6	-0.68	-0.76	-0.81	-0.89	-0.62	1	-0.83	-0.87	0.64	-0.88	0.21	0.18	0.82	-0.81	-0.12	0.058
F7	0.68	0.76	0.81	0.88	0.6	-0.83	1	0.86	-0.63	0.87	-0.22	-0.17	-0.8	0.8	0.11	-0.079
F8	0.73	0.83	0.86	0.91	0.72	-0.87	0.86	1	-0.7	0.95	-0.2	-0.2	-0.85	0.86	0.17	-0.14
F9	-0.6	-0.64	-0.61	-0.68	-0.6	0.64	-0.63	-0.7	1	-0.74	-0.24	0.34	0.62	-0.61	-0.16	0.14
F10	0.74	0.89	0.87	0.96	0.74	-0.88	0.87	0.95	-0.74	1	-0.19	-0.23	-0.87	0.86	0.23	-0.2
F11	-0.07	-0.19	-0.21	-0.22	-0.088	0.21	-0.22	-0.2	-0.24	-0.19	1	-0.25	0.19	-0.21	-0.051	0.051
F12	-0.012	-0.28	-0.18	-0.18	-0.2	0.18	-0.17	-0.2	0.34	-0.23	-0.25	1	0.17	-0.17	0.013	-0.024
F13	-0.68	-0.75	-0.79	-0.87	-0.6	0.82	-0.8	-0.85	0.62	-0.87	0.19	0.17	1	-0.79	-0.13	0.079
F14	0.68	0.75	0.8	0.87	0.61	-0.81	0.8	0.86	-0.61	0.86	-0.21	-0.17	-0.79	1	0.12	-0.069
F15	0.17	0.21	0.12	0.14	0.25	-0.12	0.11	0.17	-0.16	0.23	-0.051	0.013	-0.13	0.12	1	-0.52
Class	-0.15	-0.19	-0.057	-0.081	-0.27	0.058	-0.079	-0.14	0.14	-0.2	0.051	-0.024	0.079	-0.069	-0.52	1

Fig 1: Heat Map

The heatmap shows the accuracy count. The values shown in positive display higher accuracy.

Approach

In this paper the random state has been chosen as 42, which is used for initializing the internal random number generator, which will decide the splitting of data into train and test indices. The data is split into 1050 and 450 for all the 15 features.

2.2 Classification Algorithms

In classification models use training dataset and will identify how to pre-eminent map example of data to definite class labels. The training dataset needs to be sufficiently representative of the problem and need to have many instances of every class label.

2.3 Decision Trees

Decision trees are a simple classification algorithm that can handle feature interaction. Decision trees are non-parametric; therefore they can deal with mathematics and/or data sets, duplicated attributes, and noisy data. For this model, we created a confusion matrix with varied accuracy, precision, and f1-score values to demonstrate how well our model operates[1].

```
accuracy for decision tree model 0.81
classification Report for decision tree model
              precision    recall  f1-score   support

       False       0.81      0.85      0.83       162
        True       0.81      0.77      0.79       138

    accuracy                           0.81       300
   macro avg       0.81      0.81      0.81       300
weighted avg       0.81      0.81      0.81       300
```

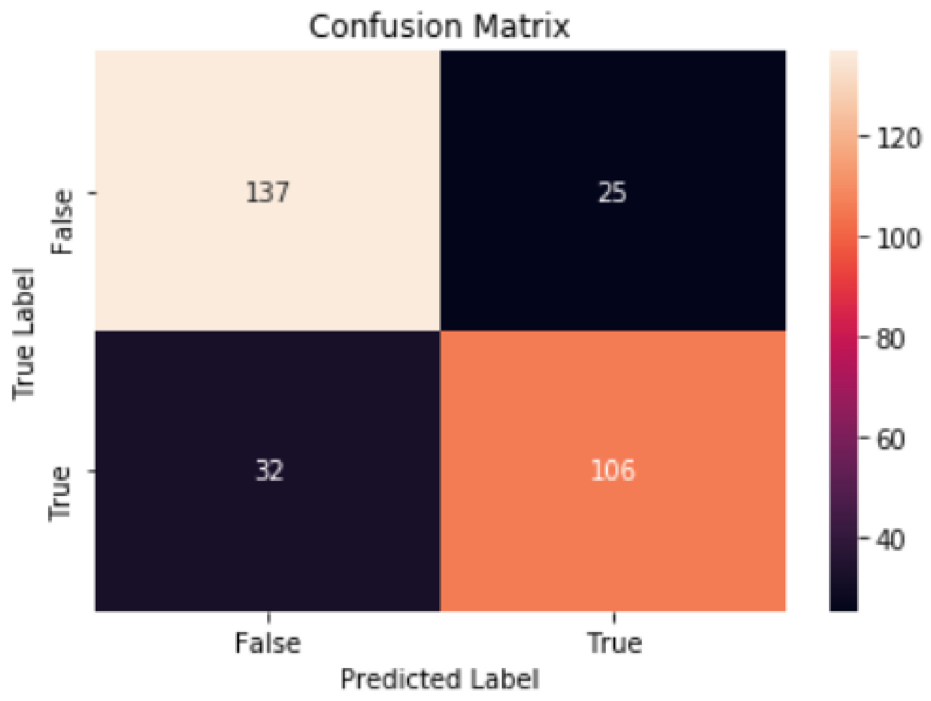


Fig 2: Confusion Matrix for Decision Trees

2.4 K-Nearest Neighbor

The K-Nearest Neighbor algorithm is a technique that works on an instance-by-instance basis. These algorithms are generally known as lazy learners since all training data are kept and new classifiers are not constructed until an unseen sample is presented. As a result, training is rapid, and the theory is simple to understand and apply. At the same time, it is capable of dealing with noisy training data. However, its performance is better when the target labels are more than two and may vary depending on the size of the dataset; the value of k must be chosen, so cross validation can be computationally expensive; and the algorithm is susceptible to the local structure of the data as well as irrelevant feature[2].

```
accuracy for  knn model 0.61
classification Report for knn tree model
              precision    recall  f1-score   support

       False       0.64      0.65      0.64       162
        True       0.58      0.57      0.57       138

    accuracy                           0.61       300
   macro avg       0.61      0.61      0.61       300
weighted avg       0.61      0.61      0.61       300
```

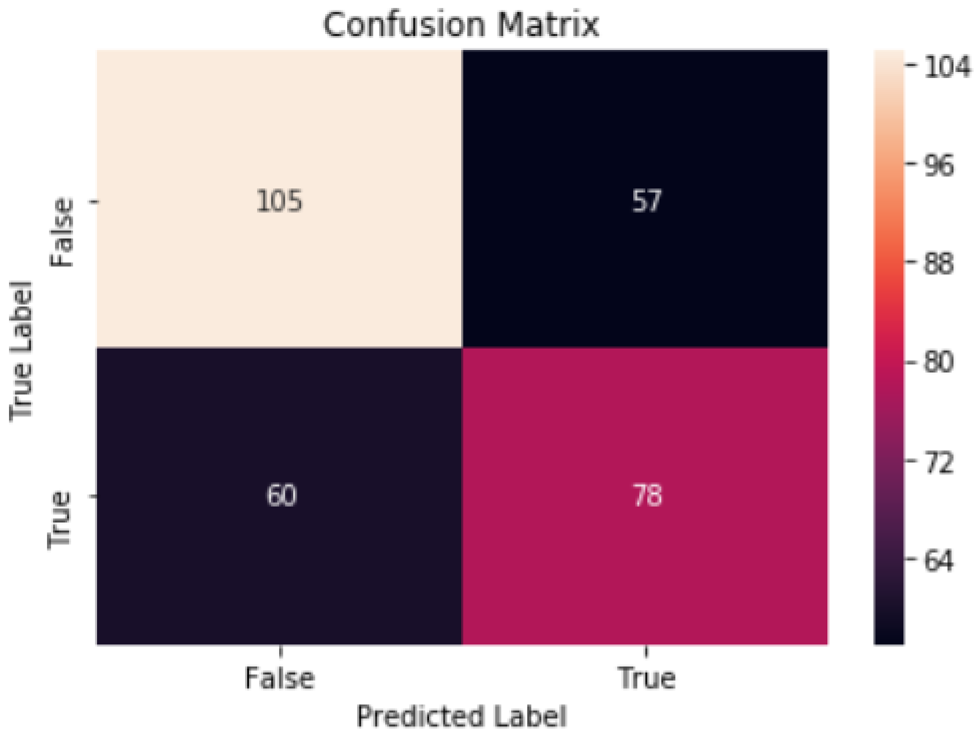
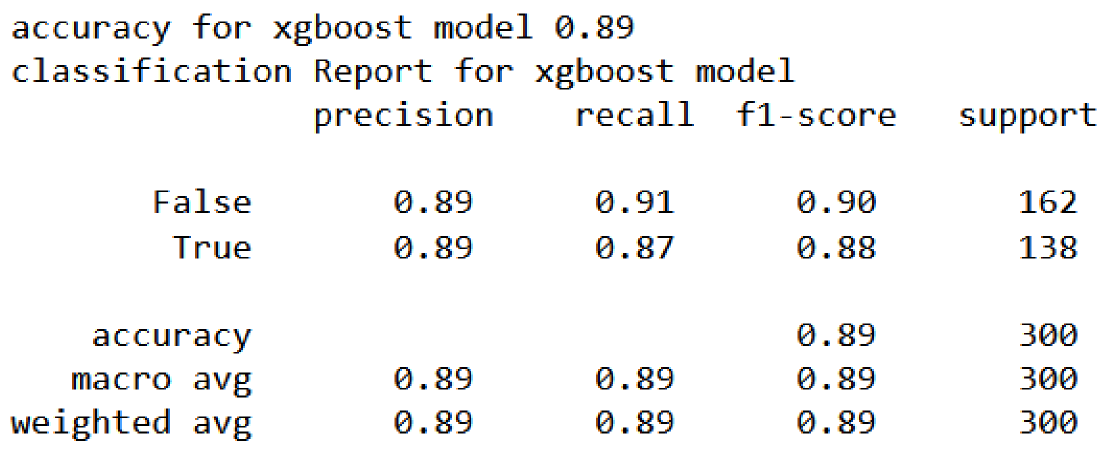


Fig 3: Confusion Matrix for KNN

2.5 XGBoost

XGBoost is an effective strategy for creating supervised regression models. A loss function and a regularization term are included in the objective function. It describes the difference between actual and expected values, i.e. how much the model findings stray from reality. The accuracy for the model is 0.8777[5].



```
accuracy for xgboost model 0.89
classification Report for xgboost model
                precision    recall  f1-score   support

       False       0.89      0.91      0.90       162
        True       0.89      0.87      0.88       138

    accuracy                           0.89       300
   macro avg       0.89      0.89      0.89       300
weighted avg       0.89      0.89      0.89       300
```

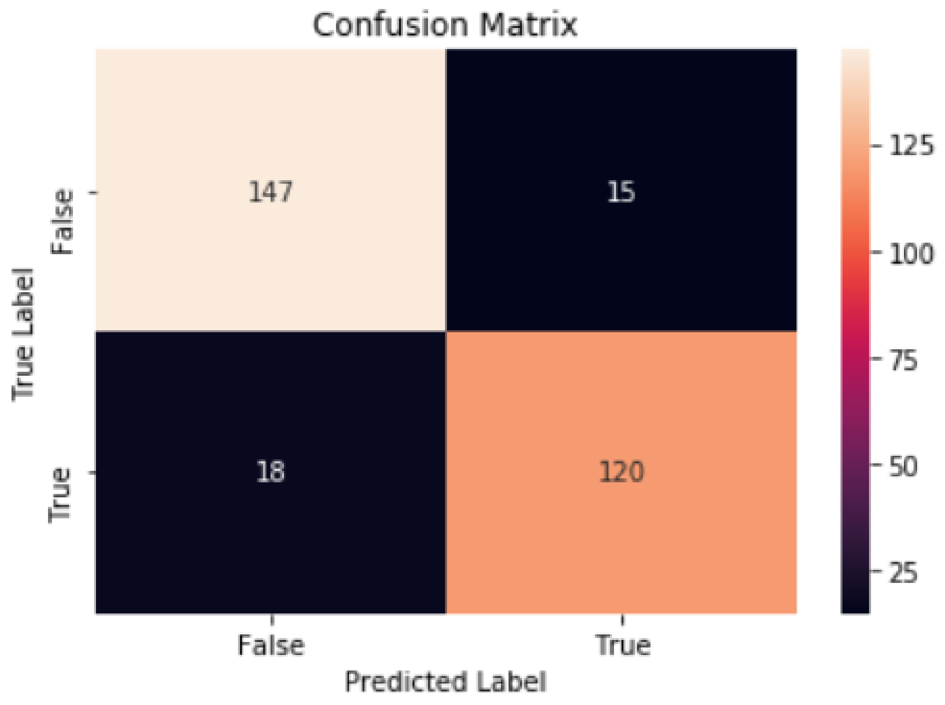
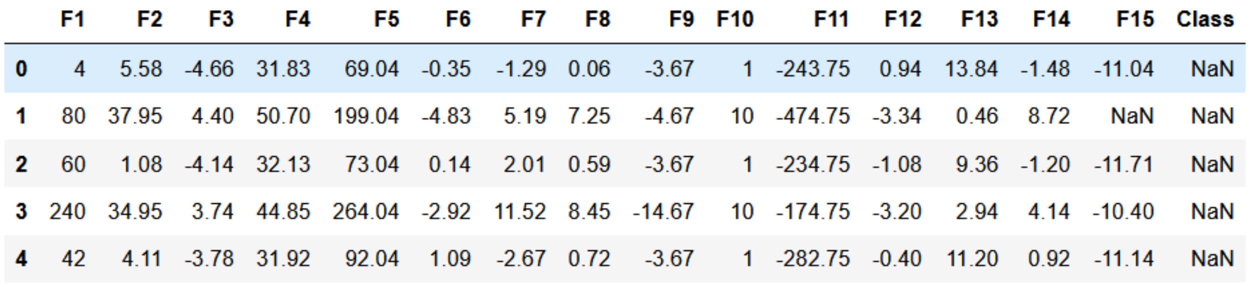


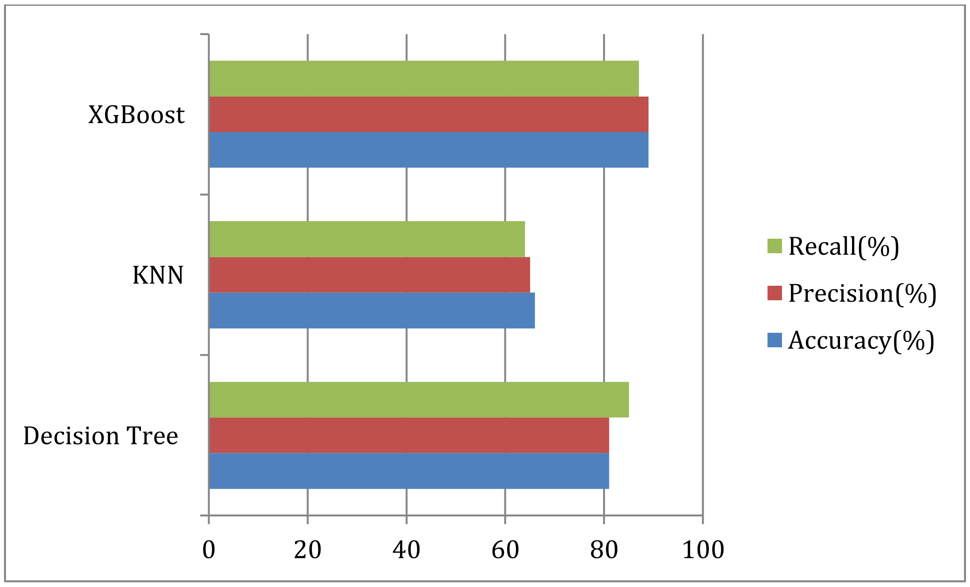
Fig 4: Confusion Matrix For XgBoost

Prediction On a hold-out set
After training the model we have tested the model against the train split data which has been displayed for 5 headers.



	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	Class
0	4	5.58	-4.66	31.83	69.04	-0.35	-1.29	0.06	-3.67	1	-243.75	0.94	13.84	-1.48	-11.04	NaN
1	80	37.95	4.40	50.70	199.04	-4.83	5.19	7.25	-4.67	10	-474.75	-3.34	0.46	8.72	NaN	NaN
2	60	1.08	-4.14	32.13	73.04	0.14	2.01	0.59	-3.67	1	-234.75	-1.08	9.36	-1.20	-11.71	NaN
3	240	34.95	3.74	44.85	264.04	-2.92	11.52	8.45	-14.67	10	-174.75	-3.20	2.94	4.14	-10.40	NaN
4	42	4.11	-3.78	31.92	92.04	1.09	-2.67	0.72	-3.67	1	-282.75	-0.40	11.20	0.92	-11.14	NaN

2.6 Results The data has been trained and tested and out of the three methods Decision Trees, K-nearest neighbors and Xgboost Classifier the best scores for accuracy are for Xgboost Classifier. The reported accuracy of Xgboost Classifier on validation data is 89% which is highest as compared to others.



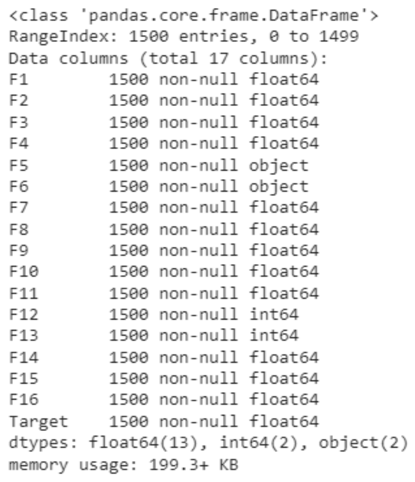
Graph 1: Comparison of Implementation results

3. Additional Case Study-Task a

Our pilot case study classification models are deployed for prediction of whether a patient is diabetic (YES) or not (NO). In this additional case study our task is to deploy a new model which will predict the quantity of drug for diabetic patients should take in a day. In this case study we have a dataset of related features of each patient to predict the everyday amount of drug each day the patient will take. In this study linear regression is performed to predict the quantity of drug per day. Here for prediction Algorithms used are Linear regression, Random Forest Regression and XGBoost regression and finally prediction on a hold-out test set.

3.1 Preprocessing on Dataset

After preprocessing all the values present in the dataset were taken without filtering out null values. Data Standardization/Scaling: This term in Data preprocessing is also termed as scaling, where the scale suggests that to convert the data set in the form where every attribute/column converts into the same scale.

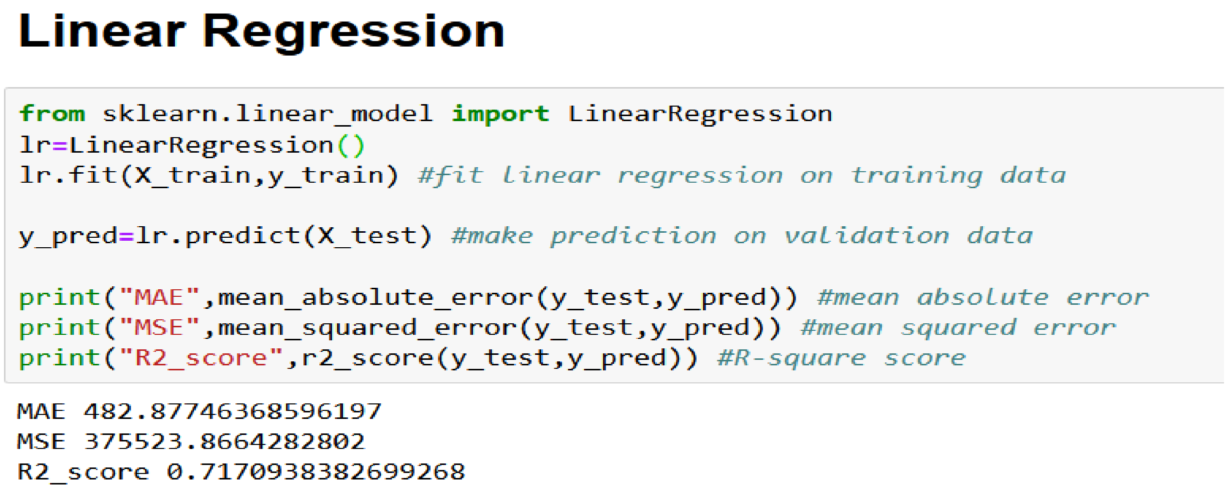


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 17 columns):
F1        1500 non-null float64
F2        1500 non-null float64
F3        1500 non-null float64
F4        1500 non-null float64
F5        1500 non-null object
F6        1500 non-null object
F7        1500 non-null float64
F8        1500 non-null float64
F9        1500 non-null float64
F10       1500 non-null float64
F11       1500 non-null float64
F12       1500 non-null int64
F13       1500 non-null int64
F14       1500 non-null float64
F15       1500 non-null float64
F16       1500 non-null float64
Target    1500 non-null float64
dtypes: float64(13), int64(2), object(2)
memory usage: 199.3+ KB
```

Fig 6: Values of the dataset

3.2 Linear Regression

Linear Regression is a supervised Machine Learning model that finds the best linear line fitting between the variables that are independent and dependent, i.e. it discovers the linear link between any two variables[6].



Linear Regression

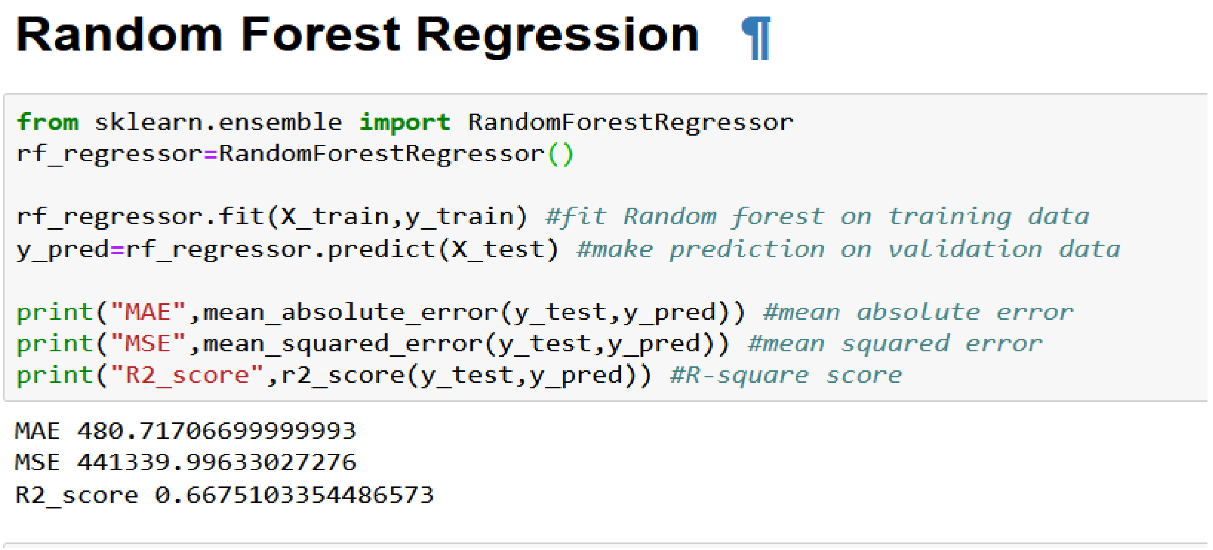
```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()
lr.fit(X_train,y_train) #fit linear regression on training data

y_pred=lr.predict(X_test) #make prediction on validation data

print("MAE",mean_absolute_error(y_test,y_pred)) #mean absolute error
print("MSE",mean_squared_error(y_test,y_pred)) #mean squared error
print("R2_score",r2_score(y_test,y_pred)) #R-square score
```

```
MAE 482.87746368596197
MSE 375523.8664282802
R2_score 0.7170938382699268
```

Fig 7: MAE, MSE and R2 score for Linear Regression

3.3 Random Forest Regression

A Random Forest is an ensemble approach that can do both regression and classification problems by combining several decision trees using a technique known as Bootstrap and Aggregation, sometimes known as bagging. The core idea is to use numerous decision trees to determine the final output rather than depending on individual decision trees[6].



Random Forest Regression ¶

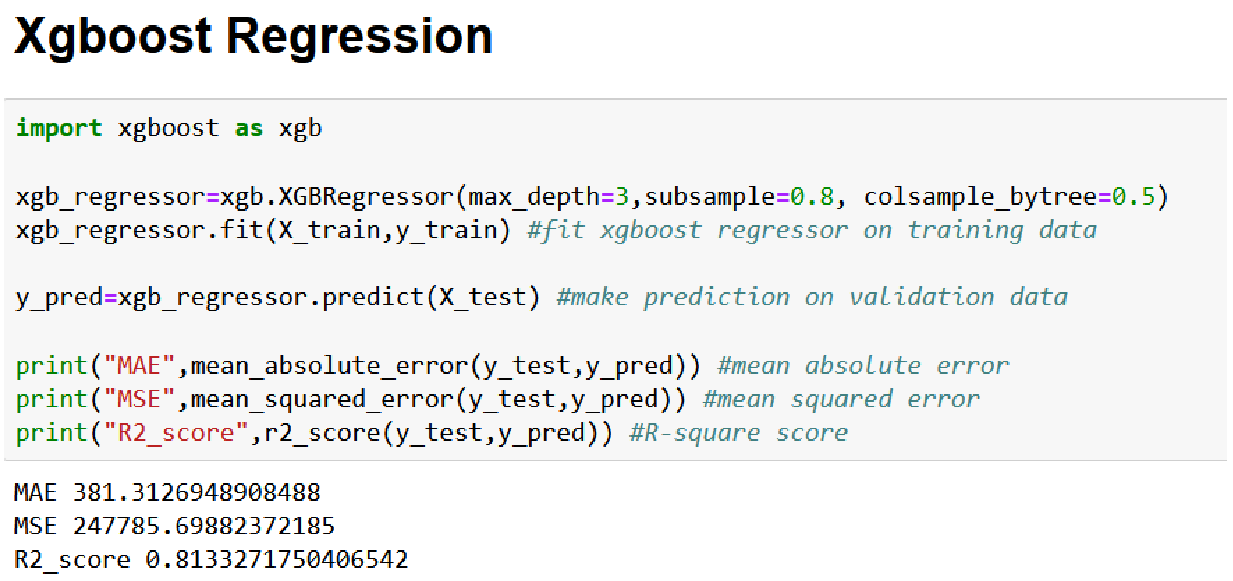
```
from sklearn.ensemble import RandomForestRegressor
rf_regressor=RandomForestRegressor()

rf_regressor.fit(X_train,y_train) #fit Random forest on training data
y_pred=rf_regressor.predict(X_test) #make prediction on validation data

print("MAE",mean_absolute_error(y_test,y_pred)) #mean absolute error
print("MSE",mean_squared_error(y_test,y_pred)) #mean squared error
print("R2_score",r2_score(y_test,y_pred)) #R-square score
```

```
MAE 480.71706699999993
MSE 441339.99633027276
R2_score 0.6675103354486573
```

Fig8: MAE, MSE and R2 score for Random Forest Regression

3.4 Xgboost Regression

This Gradient boosting decision tree was implemented in this study, which is an iterative decision algorithm combination of a multiplicity of decision trees composed to predict the final decision. As compared to linear regression which will only help in linear regression but Xgboost helps in both types of regression problem linear and nonlinear regression[7].



Xgboost Regression

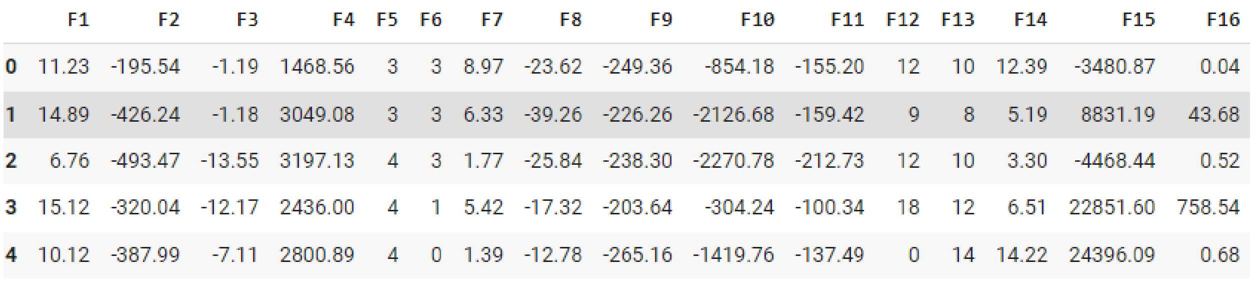
```
import xgboost as xgb

xgb_regressor=xgb.XGBRegressor(max_depth=3,subsample=0.8, colsample_bytree=0.5)
xgb_regressor.fit(X_train,y_train) #fit xgboost regressor on training data

y_pred=xgb_regressor.predict(X_test) #make prediction on validation data

print("MAE",mean_absolute_error(y_test,y_pred)) #mean absolute error
print("MSE",mean_squared_error(y_test,y_pred)) #mean squared error
print("R2_score",r2_score(y_test,y_pred)) #R-square score
```

```
MAE 381.3126948908488
MSE 247785.69882372185
R2_score 0.8133271750406542
```

Fig9: MAE, MSE and R2 score for XgBoost Regression

Prediction on a hold-out set

After training the model we have tested the model against the train split data which has been displayed for 5 headers.



	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16
0	11.23	-195.54	-1.19	1468.56	3	3	8.97	-23.62	-249.36	-854.18	-155.20	12	10	12.39	-3480.87	0.04
1	14.89	-426.24	-1.18	3049.08	3	3	6.33	-39.26	-226.26	-2126.68	-159.42	9	8	5.19	8831.19	43.68
2	6.76	-493.47	-13.55	3197.13	4	3	1.77	-25.84	-238.30	-2270.78	-212.73	12	10	3.30	-4468.44	0.52
3	15.12	-320.04	-12.17	2436.00	4	1	5.42	-17.32	-203.64	-304.24	-100.34	18	12	6.51	22851.60	758.54
4	10.12	-387.99	-7.11	2800.89	4	0	1.39	-12.78	-265.16	-1419.76	-137.49	0	14	14.22	24396.09	0.68

3.5 Results & Discussion : The above implemented model was tested on various parameters such as MAE (Mean absolute error), MSE(Mean Squared Error) and R2 score.

Mean Absolute Error: MAE uses the average of absolute errors for a collection of predictions and observations to calculate the magnitude of errors for the entire group. MAE is often referred to as the L1 loss function[8]. Which is calculated as : $(1/n) * \Sigma|y_i - x_i|$

Where Σ: Greek Symbol for Summation

y_i: Actual Value of the ith observation

x_i: Calculated value for the ith observation

n: Total number of observations

MSE (Mean Squared Error): It is defined as the average of the square of the difference between the original values from the predicted values.

$$MSE = 1/N\{(\Sigma_{j=1}{}^{N})(predicted\ -\ input)^2$$

R2 Score :The R2 score is an important metric for determining the performance of a regression-based machine learning model. This also implies that the closer the r squared score is to 1, the better trained the model is.

R square is calculated by using the following formula : $R^2 = 1 - (ss_{res}/SS_{tot})$

Where $SS_{res}$is the residual sum of squares and SS_{tot} is the total sum of squares. In additional comparative study after implementation of three regression algorithms Linear regression, Random forest classifier and Xgboost Regression. For the implementation data has been trained and tested and out of the three methods Xgboost Regression has the best scores. Which are also described in table below:

Performance parameters	**Xgboost Regression**	**Random Forest Regression**	**Linear Regression**
MAE	381.31269489	480.717	482.877
MSE	247785.69	441339.99	375523.86
R2_score	0.81332717	0.66	0.7170938

Table 1 : Implementation Results

Conclusion

In this study, our results specify machine learning algorithms are gradually becoming appropriate for use in medical daily practice. In the first task we implement classification algorithms to predict whether a person is diabetic or not using three classification algorithms, implementation results show that extreme boosting algorithm is best and in additional comparative study we implement a model to predict the quantity of drug given to each diabetic patient. Here we use regression because it helps to predict the output of the dependent variable for entities where some descriptive variable is available. The Regression has been performed using Linear regression, Random Forest and Xgboost. The Xgboost is the best regression model with 81% accuracy. Therefore, these techniques facilitate powerful tools for making patient's quality life.

.

REFERENCES

[1]Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[2]https://scikitlearn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

[3]https://scikitlearn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

[4]https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

[5]https://scikitlearn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

[6]https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[7]https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRGradient.html

[8] U. M. Butt, S. Letchmunan, M. Ali, F. H. Hassan, A. Baqir, and H. H. R. Sherazi, "Machine Learning Based Diabetes Classification and Prediction for Healthcare Applications," *J. Healthc. Eng.*, vol. 2021, 2021, doi: 10.1155/2021/9930985.