Dataset 2 Diabetes

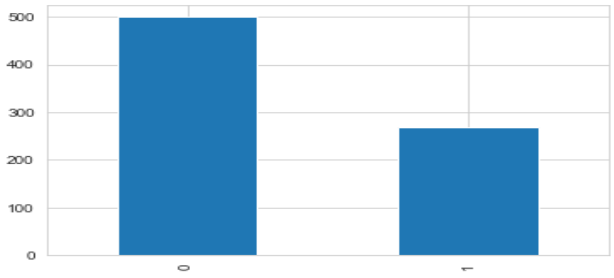
Advantage of this project: - The rules derived will be helpful for doctors to identify patients suffering from diabetes. Further predicting the disease early leads to treating the patient before it becomes critical.

DATASET DESCRIPTION

- This dataset has 769 samples of diabetic and healthy individuals.
- All patients here are females of at least 21 years of age.
- The dataset has total 9 attributes out of which 8 are independent variables and one is the dependent variable i.e. target variable which determines whether patient is having diabetes or not.

SUMMARY STATISTICS

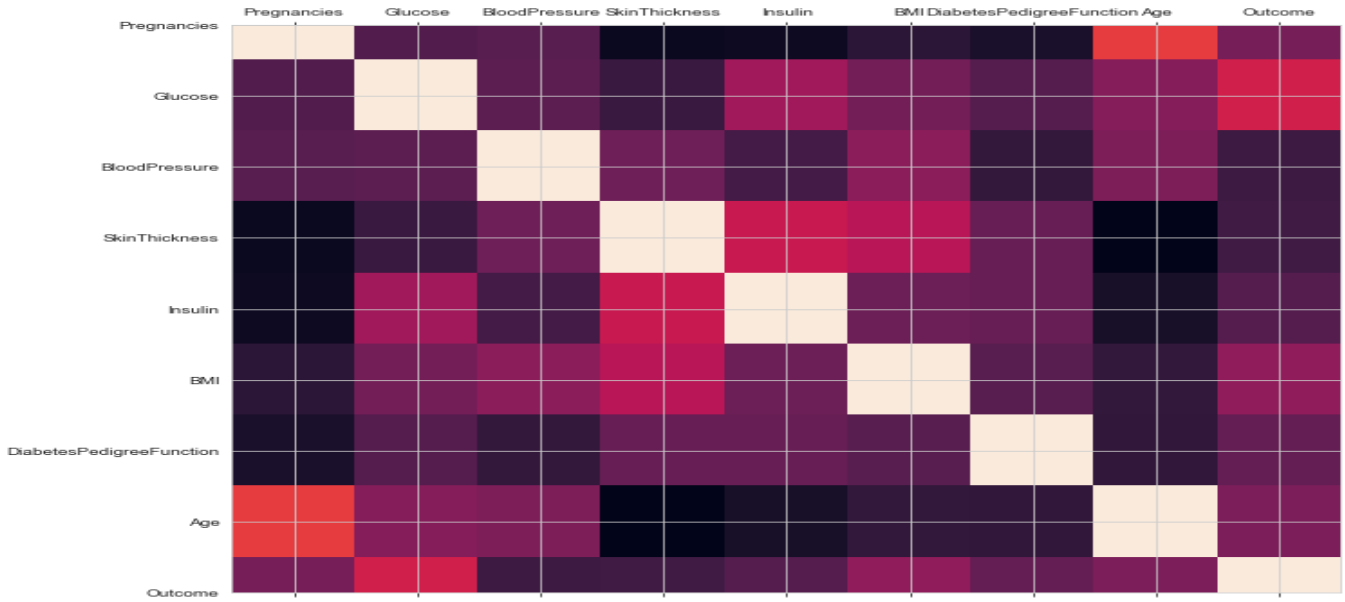
- The diabetes dataset is credited to UCI machine learning database repository. The dataset consists of 769 samples, out of which 500 are non-diabetic while 269 are diabetic people. So, we have 34.90% people in current data set who have diabetes and rest of 65.10% does not have diabetes. It is a good distribution True/False cases of diabetes in data.



- No null values are found in this data.
- But there can be lots of entries with 0 values. So, I replaced the extreme values with median values. It is advised to not use mean values as they are affected by outliers.
- Data was divided into training and testing data into 80:20 ratio. eighty percent was training data and twenty percent was testing data.

CORRELATION

After checking the correlation between the data, it is observed that yellow colour represents max correlation and blue colour represents min correlation. We can see none of variable have correlation with any other variables.



MODEL SELECTION

PART 2: SUPPORT VECTOR MACHINE

The SVM model performance can be improved by tuning parameter values. Some important parameter values that have an impact on the model performance are:

1. **kernel** – Various options are available such as linear, rbf, poly. The default used is rbf.
2. **gamma** - Kernel coefficient for 'rbf', 'poly' and 'sigmoid'. Higher the value of gamma, will try to exact fit the as per training data set i.e. generalization error and cause over-fitting problem.
3. **C** - Penalty parameter C of the error term. It also controls the tradeoff between smooth decision boundary and classifying the training points correctly.

A. Gaussian(rbf) Kernel

- Firstly, I trained this model by calling standard SVC () function without doing Hyper-parameter Tuning and saw its classification and confusion matrix.

```
[[100   0]
 [ 54   0]]
              precision    recall  f1-score   support

           0       0.65      1.00      0.79       100
           1       0.00      0.00      0.00        54

    accuracy                           0.65       154
   macro avg       0.32      0.50      0.39       154
weighted avg       0.42      0.65      0.51       154
```

I got **65% accuracy** by performing this model. Then I realized that that recall, f1-score, and precision for class 1 are always 0. It means that classifier is always classifying everything into a single class i.e. class 0! This means our model needs to have its parameters tuned.

- Then I tried to improve this model by using Grid Search for searching best parameters for good results. Then I added refit = True and verbose =5 as an argument for better results.

I selected these because: -

Refit = True: - it runs the same loop with cross-validation, to find the best parameter combination. Once it has the best combination, it runs fit again on all data passed to fit (without cross-validation), to build a single new model using the best parameter setting.

Verbose = 5: - verbose is declared to understand the text output better.

Then I inspected the best parameters found by GridSearchCV in the best_params_ attribute, and the best estimator in the best_estimator_ attribute. **{'C': 1, 'gamma': 0.0001}**

After that again I saw classification_report and confusion matrix.

```
[[91  9]
 [26 28]]
              precision    recall  f1-score   support

           0       0.78      0.91      0.84       100
           1       0.76      0.52      0.62        54

    accuracy                           0.77       154
   macro avg       0.77      0.71      0.73       154
weighted avg       0.77      0.77      0.76       154
```

Then I got almost **77.27% Accuracy** and 28 are the number of observations that are correctly classified.

B. Linear kernel: The accuracy is 79.22% without tuning the parameters and with tuning the parameters.

The best parameters are **{'C': 0.1, 'gamma': 1, 'kernel': 'linear'}** and 33 observations are correctly classified.

```
[[89 11]
 [21 33]]
              precision    recall  f1-score   support

           0       0.81      0.89      0.85       100
           1       0.75      0.61      0.67        54

    accuracy                           0.79       154
   macro avg       0.78      0.75      0.76       154
weighted avg       0.79      0.79      0.79       154

Accuracy 79.22077922077922
Accuracy Score of linear = 0.792208
```

C. Sigmoid Kernel: The accuracy is same i.e. 64.93%. The number of observations classified correctly is 100. All these 100 observations are classified in class 0 which means that these customers did not churn.

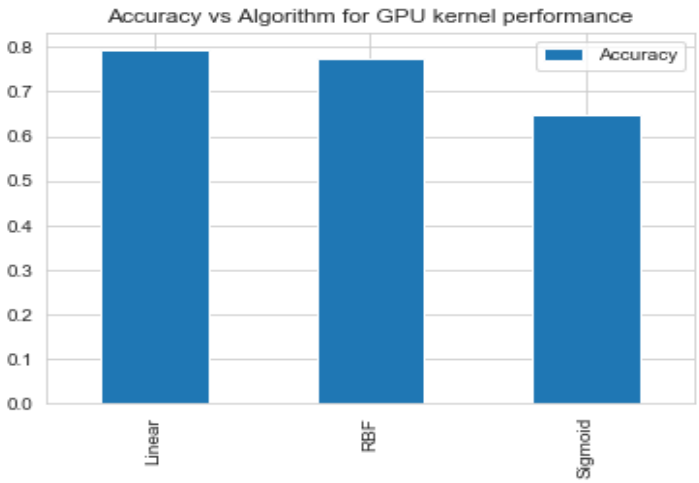
```
[[100   0]
 [ 54   0]]
              precision    recall  f1-score   support

           0       0.65      1.00      0.79       100
           1       0.00      0.00      0.00        54

    accuracy                           0.65       154
   macro avg       0.32      0.50      0.39       154
weighted avg       0.42      0.65      0.51       154

Accuracy 64.93506493506493
```

Comparison between kernels



In this dataset, Linear kernel with tuning parameters and k-fold cross validation performed the best with an accuracy of 79.22%. The above chart shows the results with grid search that is used for hyperparameter tuning models and it gave the higher accuracy when compared with same algorithm with default parameters.

PART 3: DECISION TREE

Attribute selection is used for selecting the splitting criterion that partitions the data into the best possible manner. **Information gain, Gini index, max depth, splitter are some of the popular selection measures.**

DATASET 2 DIABETES

First observation by default parameters shows that Misclassified samples are: 49 with an accuracy of 68.18%

```
Misclassified samples: 49
[[79 21]
 [28 26]]
              precision    recall  f1-score   support

           0       0.74      0.79      0.76       100
           1       0.55      0.48      0.51        54

    accuracy                           0.68       154
   macro avg       0.65      0.64      0.64       154
weighted avg       0.67      0.68      0.68       154

Accuracy 68.18181818181817
```

Then I performed grid search algorithm and found these results: -

Misclassified samples: 31 and Accuracy: 80%

```
DecisionTreeClassifier(class_weight=None, criterion='entropy', max_depth=5,
max_features=None, max_leaf_nodes=9,min_impurity_decrease=0.0, min_impurity_split=Non
e, min_samples_leaf=1, min_samples_split=2,random_state=None, splitter='best')
```

These are top parameters that are found for best results.

```
Misclassified samples: 31
Accuracy: 0.80
[[86 14]
 [17 37]]
              precision    recall  f1-score   support

           0       0.83      0.86      0.85       100
           1       0.73      0.69      0.70        54

    accuracy                           0.80       154
   macro avg       0.78      0.77      0.78       154
weighted avg       0.80      0.80      0.80       154
```

PART 4: GRADIENT BOOSTING

DATASET 2 DIABETES

I used **XGBoost classifier** for performing boosting. For pruning purpose, I used grid search algorithm and found best results with these parameters.

```
{'learning_rate': 0.01, 'max_features': 4, 'n_estimators': 300}
```

With XGboost, I got an accuracy of 77% always although training of data was very fast as compared to Gradient Boosting Classifier.

```
 ---GBC---
GBC AUC = 0.73
              precision    recall  f1-score   support

           0       0.79      0.88      0.83       100
           1       0.72      0.57      0.64        54

    accuracy                           0.77       154
   macro avg       0.76      0.73      0.74       154
weighted avg       0.77      0.77      0.77       154

[[88 12]
 [23 31]]
GBC accuracy is 0.77
```

But when I used **Gradient Boosting Classifier** and performed Grid Search on it, the accuracy increased to 79% which shows that in this dataset, Gradient Boosting works better.

```
{'learning_rate': 0.01, 'max_features': 8, 'n_estimators': 500}
```

```
 ---GBC---
GBC AUC = 0.75
              precision    recall  f1-score   support

           0       0.81      0.89      0.85       100
           1       0.75      0.61      0.67        54

    accuracy                           0.79       154
   macro avg       0.78      0.75      0.76       154
weighted avg       0.79      0.79      0.79       154

[[89 11]
 [21 33]]
GBC accuracy is 0.79
```

PART 5: CROSS VALIDATION

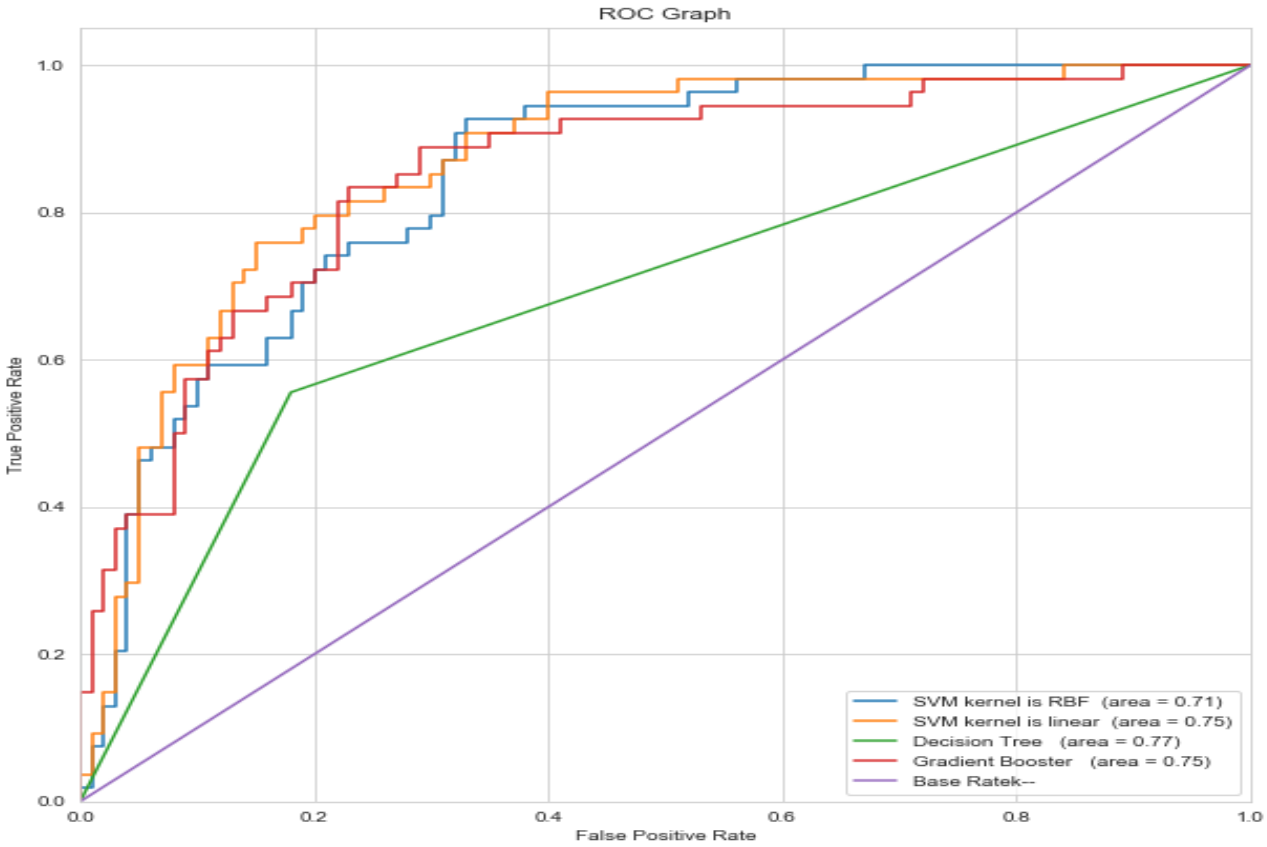
DATASET 2 DIABETES

Then I performed cross validation for all the algorithms and found these results: -

Accuracy (SVC Sigmoid): 0.649487
Accuracy (SVC linear): 0.753077
Accuracy (SVC rbf): 0.741026
Accuracy (Decision Tree): 0.727179
Accuracy (Gradient booster): 0.793077

By performing cross validation, maximum accuracy of 79.3% is achieved by Gradient Boosting algorithm because it only did cross validation. But when I performed grid search, then the best result is achieved by Decision Tree with an accuracy of 80% that is because grid search performs hyper tuning of parameters and cross validation both.

ROC Curve Algorithm Comparison for Diabetes Dataset



Higher the AUC, better the model is at distinguishing between diabetes patients and non-diabetes patients. So, the best result is achieved by Decision Tree with an AUC value of 0.77. Sigmoid kernel performed the worst with an AUC = 0.50 that is this model was unable to distinguish between class 0 and class 1.

DISCUSSSIONS

Results: Machine learning model performance is relative and ideas of what score a good model can achieve only make sense and can only be interpreted in the context of the skill scores of other models also trained on the same data. So, for me the best results for diabetes dataset is 80%. Diabetes results could have been much better with more learning and by trying more parameters and by more algorithms.

Cross validation: In general, there is a trade-off between accuracy and generalisation. The more accurate your classifier is on your training data, the less it will probably generalise (depends on your training data). This is called overfitting and cross validation tries to avoid that. Therefore, you mix training and test data on each fold. This method affected the results greatly in the dataset.

Pruning: I performed pruning by using grid search algorithm by setting the range of cost function and gamma and it showed after performing combinations which parameters are best for getting best results. By specifying max depth of the model, we can set when to stop further pruning. Maximum depth refers to the length of the longest path from a root to a leaf.

Kernel choice: I randomly choose kernels that performs the best amongst all other kernel. Also, the computational speed was another factor to choose the kernel. In Diabetes dataset, linear kernel showed the best results.

Best performing algorithm: In Diabetes dataset, the best performing algorithm is Decision Tree after performing Grid search with hyper tuning of parameters and k-fold cross validation with an accuracy of 80%**.**