2023

Heart Disease Prediction- Individual Report

MACHINE LEARNING I - GROUP 8

YU, GUOSHAN

1. Introduction

Heart disease is a leading cause of death in the US, affecting people of most races. Detecting and preventing the key risk factors for heart disease is crucial in healthcare. This project aims to develop a machine-learning model to predict a patient's likelihood of having heart disease based on key indicators such as high blood pressure, high cholesterol, smoking, diabetic status, obesity, physical activity, and alcohol consumption.

We will use the Personal Key Indicators of Heart Disease dataset from Kaggle: 'https://www.kaggle.com/datasets/kamilpytlak/personal-key-indicators-of-heart-disease' which contains 319,795 samples with 17 attributes (13 categorical features and 4 numerical features), with the target variable "HeartDisease" indicating whether the respondent had heart disease or not. It is well noticed that this dataset is imbalanced with 292,422 data points with target-no and 27,373 data points with target-yes.

2. Description of your individual work

2.1 Data Preprocessing

2.1.1 Remove outliers

Identifying and handling outliers is an essential step in data preprocessing that involves assessing the quality and accuracy of the dataset. In the data preprocessing step, we followed a standard approach to identify and remove the outliers using the Interquartile Range (IQR) method Outliers = ***1.5*** ***** IQR. The graph below indicated the outliers in our dataset.

```
feature: BMI
Q1: 24.03
Q3: 31.42
IQR: 7.390000000000001
number of outliners in upper bound:10351
number of outliners in lower bound:45
---------------------------------
feature: PhysicalHealth
Q1: 0.0
Q3: 2.0
IQR: 2.0
number of outliners in upper bound:47146
number of outliners in lower bound:0
---------------------------------
feature: MentalHealth
Q1: 0.0
Q3: 3.0
IQR: 3.0
number of outliners in upper bound:51576
number of outliners in lower bound:0
---------------------------------
feature: SleepTime
Q1: 6.0
Q3: 8.0
IQR: 2.0
number of outliners in upper bound:3204
number of outliners in lower bound:1339
---------------------------------
```

2.1.2 Encoding the data

The categorical variables were encoded using one-hot encoding to convert them into a format that can be used in machine learning algorithms. As the categorical features we have in the

dataset are not ranking in difference which share the same importance For our target, I simply replace yes with 1 and no with 0.

```
# encode the target with yes:1 and no:0
y_encoded = y_clean.replace({'HeartDisease':{'Yes':1,'No':0}})
## one hot encoding for categorical features
x_encoded = pd.get_dummies(x_clean)
```

2.1.3 Standardized the Data

After encoding the categorical variables using one-hot encoding, the continuous features in the dataset were normalized using the `MinMaxScaler` from the `sklearn.preprocessing` module. By normalizing the continuous features, it ensured that they were on a comparable scale, enabling the machine learning algorithm to more effectively learn from the data.

```
# %%
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
x_scaled = scaler.fit_transform(x_encoded)
x_scaled = pd.DataFrame(x_scaled,columns=x_encoded.columns)
```

2.1.4 Split the Data

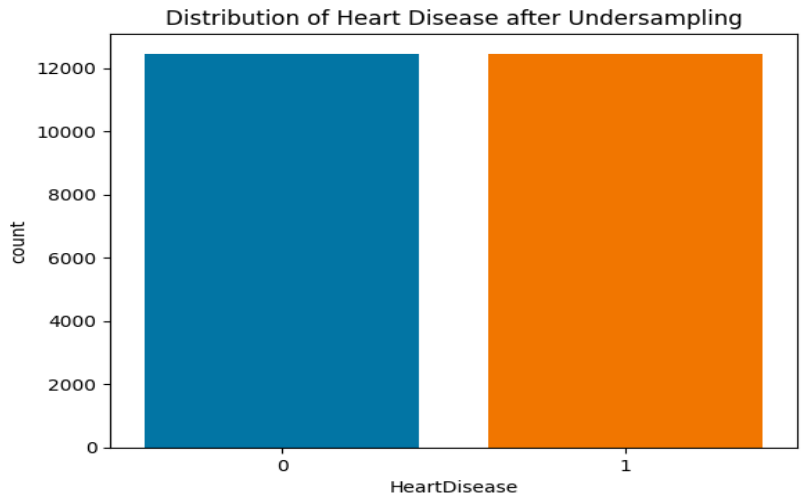
After encoding and scaling the dataset, the data was split into training and testing sets.

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test =
train_test_split(x_scaled,y_encoded,test_size=0.2,random_state=42)
```

2.2 Modelling

2.2.1 Balance the data – Undersampler()

As the dataset we have is imbalance, I apply undersampler to the majority class and match the size of the minority class.



2.2.2 RFECV

After applying the undersampling technique, a feature selection technique called Recursive Feature Elimination with Cross-Validation (RFECV) is used to select the most important features for the classifier. RFECV is a wrapper method that works by recursively removing features, fitting the model, and evaluating performance until the optimal number of features is reached. The number of features that result in the best performance is determined by cross-validation.

3. Describe the work

I responsible for the data preprocessing and modelling session which implement the pipline to guide the logistic regression, decision Tree and random forest. Before training the model, I apply undersampling and feature selection method (RFECV) for each classifier.

3.1 Data Preprocessing

3.1.1 Remover outliers

After removing the outliers, the data size has been dropped down to ***225,471***. The following is the code for removing the outliers.

```
def remove_outliers(feature):
    Q1 = df_clean[feature].quantile(0.25)
    Q3 = df_clean[feature].quantile(0.75)
    IQR = Q3 - Q1
    df_clean.drop(df_clean[df_clean[feature] > Q3 + 1.5 * IQR].index, inplace = True)
    df_clean.drop(df_clean[df_clean[feature] < Q1 - 1.5 * IQR].index, inplace = True)
for i in numerical_features:
    remove_outliers(i)
```

3.1.2 Encoding the data

Following is the printout for numbers of encoded categorical features.

Smoking_No	Smoking_Yes	AlcoholDrinking_No	AlcoholDrinking_Yes	Stroke_No	Stroke_Yes
1	0	1	0	0	1
1	0	1	0	1	0
0	1	1	0	1	0
1	0	1	0	1	0
1	0	1	0	1	0
...	...	...	...	...	...
0	1	1	0	1	0
1	0	1	0	1	0
1	0	1	0	1	0
0	1	1	0	1	0
1	0	1	0	1	0

3.1.3 Standardized the Data

Following are the number of the numerical features after being normalized.

	BMI	PhysicalHealth	MentalHealth	SleepTime
0	0.249577	0.0	0.0	0.500
1	0.380630	0.0	0.0	0.375
2	0.632238	1.0	0.0	0.750
3	0.456485	0.0	0.0	0.250
4	0.938706	0.0	0.0	0.875
...	...	...	...	...
225466	0.685066	0.0	0.0	0.625
225467	0.352523	0.0	0.0	0.375
225468	0.313241	0.0	0.0	0.625
225469	0.571283	0.0	0.0	0.250

3.2 Modelling

3.2.1 Train pipeline

The following is the training pipeline which first balance the train data and preceed to the RFECV method to select the best subset of features which have the optimal score of 'f1'. After the cross validation, the selected features will be selected and process to the classifier.

```
    train_pipeline = Pipeline([
        ('sampler', RandomUnderSampler()),
        ('feature_selector', RFECV(estimator=clf, step=1,
cv=StratifiedKFold(n_splits=10), scoring='f1')),
        ('classifier', clf)
    ])
    train_pipeline.fit(x_train, y_train)
```

3.2.2 Evaluation of the RFECV

The following is the codes for reviewing RFECV results, we are able to know that which features has been selected and how the cross-validation score performs in each fold from statistics values and graphs.

```
   print(f"Classifier: {clf_name}")
    print(f"Selected features:
{list(x_train.columns[train_pipeline.named_steps['feature_selector'].get_support()])}"
)
    print(f"Number of features selected:
{train_pipeline.named_steps['feature_selector'].n_features_}")
    print(f"Cross-validation score:
{train_pipeline.named_steps['feature_selector'].grid_scores_[train_pipeline.named_step
s['feature_selector'].n_features_ - 1]}")
    plt.figure()
    plt.xlabel("Number of features selected")
    plt.ylabel("Cross validation score - F1 (macro)")
    plt.plot(range(1, len(train_pipeline.named_steps['feature_selector'].grid_scores_)
+ 1), train_pipeline.named_steps['feature_selector'].grid_scores_)
```

3.2.4 Evaluation on the Test dataset

The following is the pipeline which instructed the test dataset which use the stored information from the train pipeline but exclude balancing the data such that we are testing on the raw unseen dataset.

```
    test_pipeline = Pipeline([
        ('feature_selector', train_pipeline.named_steps['feature_selector']),
        ('classifier', train_pipeline.named_steps['classifier']),

    ])
    y_pred = test_pipeline.predict(x_test)
```

The evaluation code is implemented as following which we will look at the classification report that representing the essential information like f1 score and precision sucn that we can have a better understanding about the classification performance. Also, confusion matrix and both precision recall curve and roc curve will be under reviewed as well.

```
print(classification_report(y_test, y_pred))
    print('---------------------------------')
    # print the confusion matrix in to a tabel form
    print(pd.DataFrame(confusion_matrix(y_test, y_pred), columns=['Predicted No
Disease', 'Predicted Disease'],
                       index=['Actual No Disease', 'Actual Disease']))
    print('---------------------------------')
    y_pred_proba = test_pipeline.predict_proba(x_test)[:, 1]
    print(f"ROC-AUC score: {roc_auc_score(y_test, y_pred_proba)}")
    print('---------------------------------')
    # plot the precision_recall_curve
   # draw the no skill line of the precision_recall_curve
    precision, recall, thresholds = precision_recall_curve(y_test, y_pred_proba)
    plt.figure()
    plt.plot(recall, precision)
    plt.xlabel('Recall')
    plt.ylabel('Precision')
    baseline = len(y_test[(y_test['HeartDisease']==1)])/len(y_test)
    plt.plot([0, 1],[baseline,baseline], linestyle='--')
    plt.title(f'Precision-Recall curve of {clf_name}')
    plt.show()
    print('---------------------------------')
    fpr_train, tpr_train, thresholds_train = roc_curve(y_train, y_pred_proba_train[:,
1])
    fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)
    plt.figure()
    plt.plot(fpr_train, tpr_train,
             label='Train ROC curve (area = %0.2f)' % roc_auc_score(y_train,
y_pred_proba_train[:, 1]))
    plt.plot(fpr, tpr, label='Test ROC curve (area = %0.2f)' % roc_auc_score(y_test,
y_pred_proba))
    plt.plot([0, 1], [0, 1], color='navy', linestyle='--')
    plt.xlabel('False Positive Rate')
```

```
    plt.ylabel('True Positive Rate')
    plt.title(f'ROC curve of {clf_name}')
    plt.legend()
    plt.show()
    tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()
    balanced_accuracy = (tp / (tp + fn) + tn / (tn + fp)) / 2
    print(f"Balanced accuracy: {balanced_accuracy}")
    print('---------------------------------')
```

4. Results

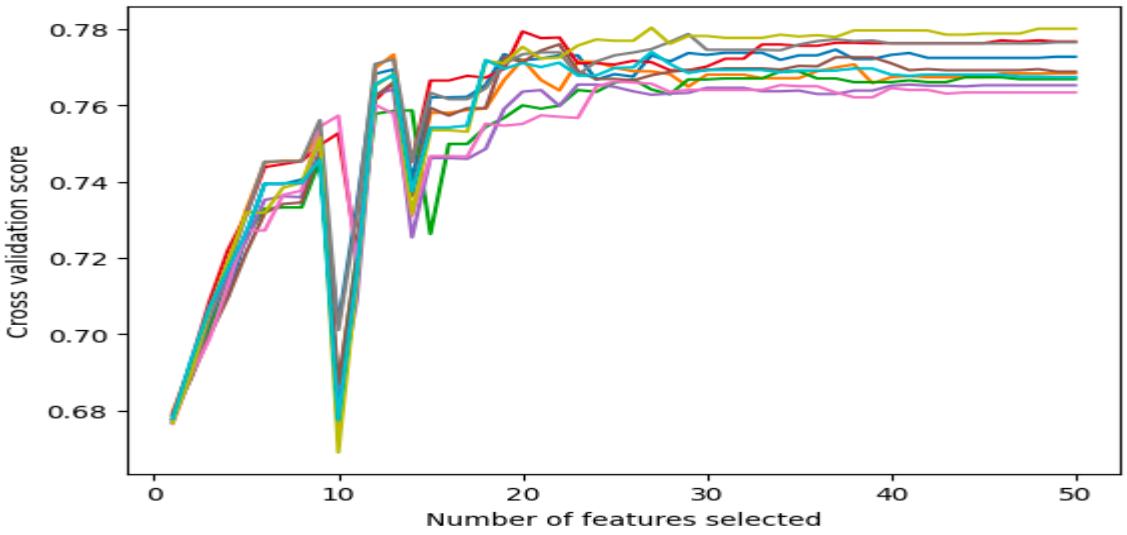
4.1 logistic Regression

RFECV

The logistic regression model with 37 features selected by RFECV achieved a cross-validation score of around 0.77, which is a significant improvement compared to the Decision Tree. Overall, the logistic regression model with the selected features by RFECV is a comparable better model with a higher f1 score.

```
Number of features selected: 37

Selected features: ['BMI', 'PhysicalHealth', 'MentalHealth', 'Smoking_No',
'Smoking_Yes', 'AlcoholDrinking_No', 'Stroke_No', 'Stroke_Yes', 'DiffWalking_No',
'DiffWalking_Yes', 'Sex_Female', 'Sex_Male', 'AgeCategory_18-24', 'AgeCategory_25-29',
'AgeCategory_30-34', 'AgeCategory_35-39', 'AgeCategory_40-44', 'AgeCategory_45-49',
'AgeCategory_55-59', 'AgeCategory_60-64', 'AgeCategory_65-69', 'AgeCategory_70-74',
'AgeCategory_75-79', 'AgeCategory_80 or older', 'Race_American Indian/Alaskan Native',
'Race_Asian', 'Race_Black', 'Diabetic_Yes', 'GenHealth_Excellent', 'GenHealth_Fair',
'GenHealth_Poor', 'GenHealth_Very good', 'Asthma_No', 'Asthma_Yes',
'KidneyDisease_No', 'KidneyDisease_Yes', 'SkinCancer_Yes']

Cross-validation score:
[0.77464789 0.76996424 0.76706984 0.77642436 0.76299213 0.77262181
0.7633946  0.77734375 0.77795152 0.76917027]
```

Given the cross validation graph of Logistic Regression, the ‘f1’ score experienced a dramatic drop in the cross-validation score at some subset when the features achieved 10, this can occur due to the particular combination of samples resulting in a less representative training set for the model, leading to a drop in performance. However, as more features are added, the overall performance of the model improves again, leading to the upward trend in the graph.



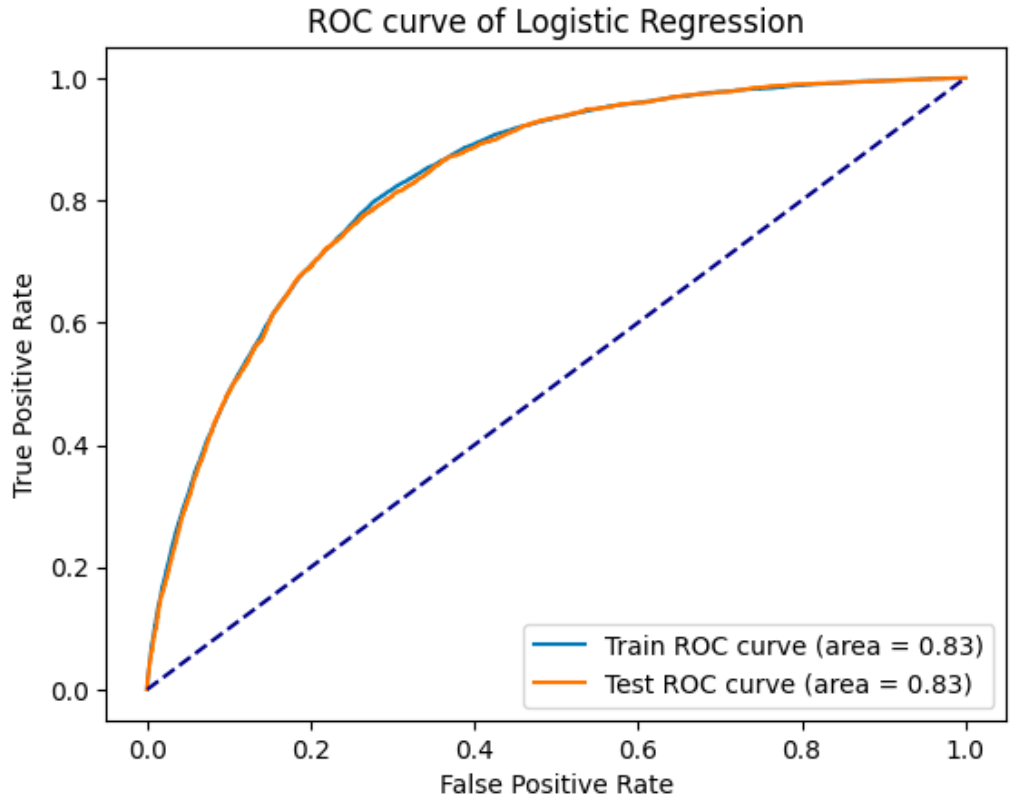
Performance

The result of logistic regression is shown below:
The classification report for the train dataset showed an overall accuracy of 0.73. The precision for predicting the positive class was low (0.18), while the recall was relatively high (0.78). This indicates that the model correctly identified most of the positive cases but also misclassified a large number of negative cases as positive. The F1-score for the positive class was 0.29, which is relatively low, indicating that the model has difficulty in correctly classifying the positive class. The balanced accuracy for the test dataset was 0.752, which is similar to the train dataset. This indicates that the model's performance on the imbalanced test dataset was also slightly better at correctly classifying the negative cases than the positive cases.

```
Train:
---------------------------------
                precision    recall  f1-score   support

           0       0.98      0.74      0.84    167914
           1       0.18      0.79      0.30     12462

    accuracy                           0.74    180376
   macro avg       0.58      0.76      0.57    180376
weighted avg       0.92      0.74      0.80    180376

---------------------------------
ROC AUC score: 0.8364346464569035
---------------------------------
Test:
              precision    recall  f1-score   support

           0       0.98      0.74      0.84     41926
           1       0.18      0.78      0.30      3169

    accuracy                           0.74     45095
   macro avg       0.58      0.76      0.57     45095
weighted avg       0.92      0.74      0.80     45095
```

```
---------------------------------
                  Predicted No Disease  Predicted Disease
Actual No Disease                30946              10980
Actual Disease                     703               2466
---------------------------------
ROC-AUC score: 0.8346955635533716
---------------------------------
Balanced accuracy: 0.7581367309217384
---------------------------------
```

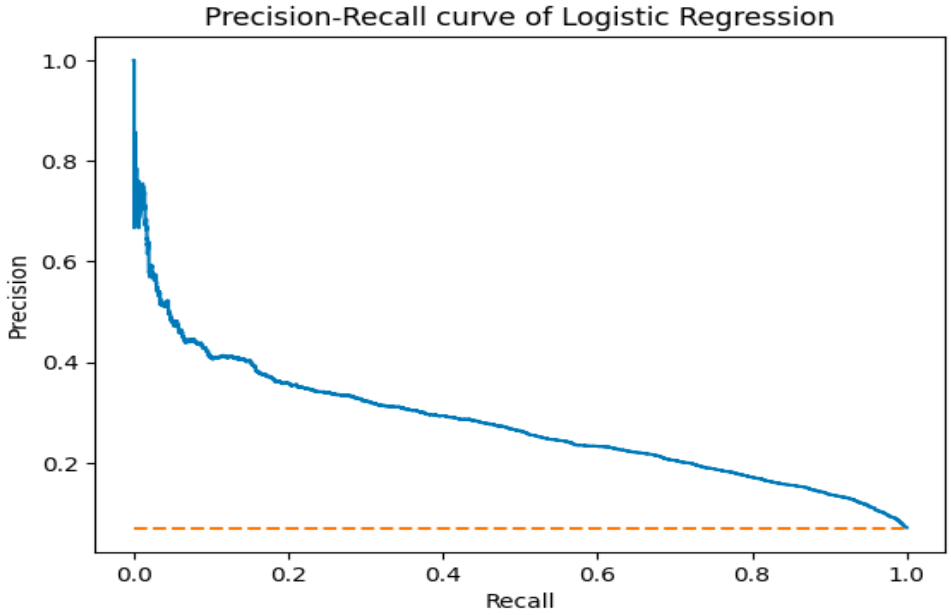
The ROC AUC score for the train dataset was 0.83, indicating good overall performance of the model. However, it is not the best indicator for imbalance data, we could see that the model has similar performance on both train and test dataset.



The precision-recall curve showed a trade-off between precision and recall, with the highest F1-score achieved at a threshold value of around 0.2. The graph of the precision-recall curve just above the no skill line which indicated the same result from the classification report.

On the other hand, the test dataset showed similar performance to the train dataset. The classification report for the test dataset showed an overall accuracy of 0.74, with a precision of 0.18 and recall of 0.78 for the positive class. The F1-score for the positive class was 0.29, similar to that of the train dataset.

The precision-recall curve for the test dataset showed a similar trade-off between precision and recall as the train dataset. The model is just barely above the no skill line for most of the thresholds.



In summary, the logistic regression model showed similar performance on both the train and test datasets, with relatively low precision and F1-score for the positive class. The ROC AUC score and precision-recall curve indicated good overall performance of the model. However, the misclassification of positive cases suggests that the model may not be suitable for practical use in identifying the positive cases in an imbalanced dataset.

4.2 Decision Tree

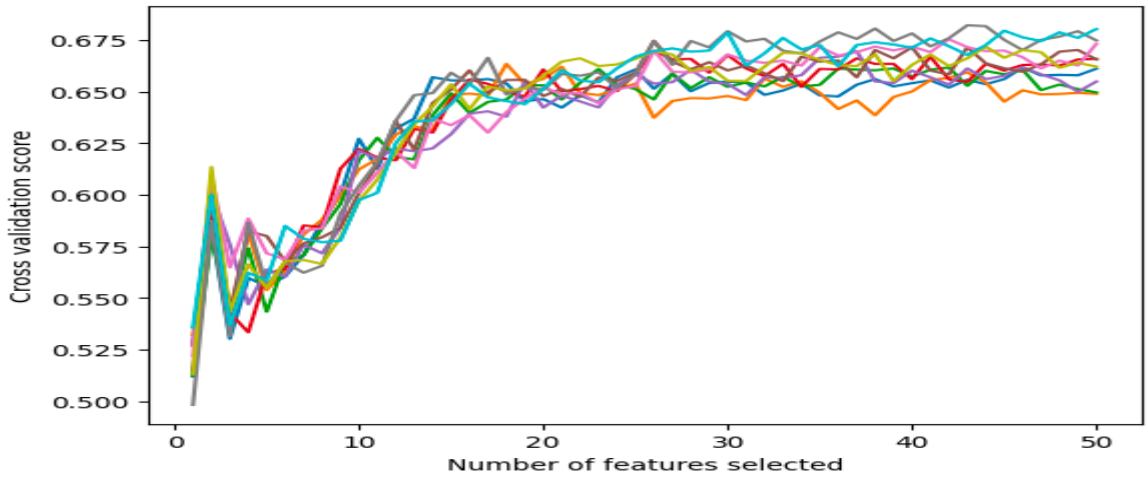
RFECV

he feature selection step of the pipeline selected 44 features from the dataset. The cross-validation score varied between 0.65 and 0.68 for different numbers of selected features, with the highest score achieved using all 44 features. This indicates that the selected features are relevant for predicting the target variable, and that the decision tree classifier performs moderately well on this dataset.

```
Number of features selected: 44

Selected features: ['BMI', 'PhysicalHealth', 'MentalHealth', 'SleepTime',
'Smoking_No', 'Smoking_Yes', 'AlcoholDrinking_No', 'AlcoholDrinking_Yes', 'Stroke_No',
'Stroke_Yes', 'DiffWalking_No', 'DiffWalking_Yes', 'Sex_Female', 'Sex_Male',
'AgeCategory_35-39', 'AgeCategory_45-49', 'AgeCategory_50-54', 'AgeCategory_55-59',
'AgeCategory_60-64', 'AgeCategory_65-69', 'AgeCategory_70-74', 'AgeCategory_75-79',
'AgeCategory_80 or older', 'Race_American Indian/Alaskan Native', 'Race_Asian',
'Race_Black', 'Race_Hispanic', 'Race_Other'Race_White', 'Diabetic_No', 'Diabetic_No,
borderline diabetes', 'Diabetic_Yes', 'PhysicalActivity_No', 'PhysicalActivity_Yes',
'GenHealth_Excellent', 'GenHealth_Fair', 'GenHealth_Good', 'GenHealth_Very good',
'Asthma_No', 'Asthma_Yes', 'KidneyDisease_No', 'KidneyDisease_Yes', 'SkinCancer_No',
'SkinCancer_Yes']
```

```
Cross-validation score:
[0.6536 0.65447154 0.66004141 0.66343434 0.65600656 0.66370251 0.66961771 0.68171021
0.67177419 0.67252396]
```

From the cross-validation graph, we could see how cross validation performed in each fold. It reached a peak when the number of features achieved was around 3 and experienced a drop. After that, the 'f1' score gradually climbed up as more features were selected.



Performance

The Results from Decision Tree is shown below:

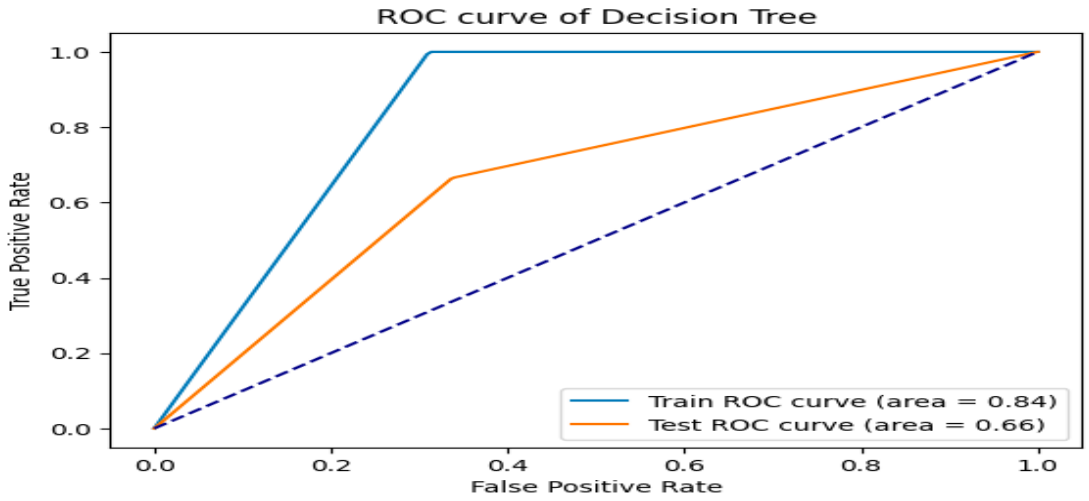
The classification report for the train dataset showed an overall accuracy of 0.71. The precision for predicting the positive class was low (0.19), while the recall was relatively high (0.99). This indicates that the model correctly identified almost all of the positive cases but also misclassified a large number of negative cases as positive. The F1-score for the positive class was 0.32, which is relatively low, indicating that the model has difficulty in correctly classifying the positive class. The confusion matrix for the train dataset showed that the model correctly classified most of the negative cases but misclassified a large number of positive cases as negative.

On the other hand, the test dataset showed lower performance than the train dataset. The classification report for the test dataset showed an overall accuracy of 0.66, with a precision of 0.13 and recall of 0.66 for the positive class. The F1-score for the positive class was 0.22, lower than that of the train dataset. The balanced accuracy for the test dataset was 0.664, indicating that the model is slightly better than random in correctly classifying the positive cases.

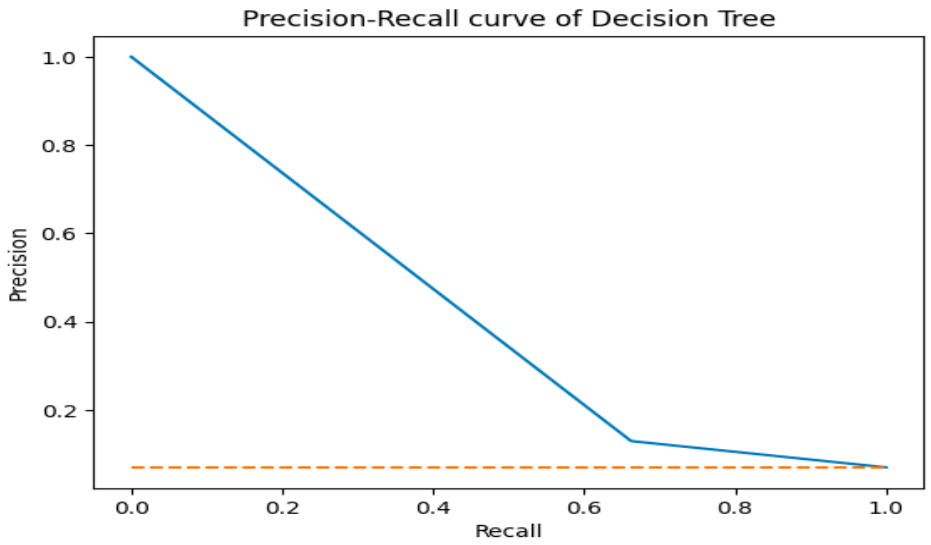
```
Train:
--------------------------------
              precision    recall  f1-score   support

           0       1.00      0.70      0.82    167914
           1       0.20      0.99      0.33     12462

    accuracy                           0.72    180376
   macro avg       0.60      0.84      0.57    180376
weighted avg       0.94      0.72      0.79    180376
```

```
---------------------------------
       ROC AUC score: 0.8468571002771255
       ---------------------------------

       Test:
                     precision    recall  f1-score   support

           0       0.96      0.67      0.79     41926
           1       0.13      0.65      0.22      3169

    accuracy                           0.67     45095
   macro avg       0.55      0.66      0.51     45095
weighted avg       0.90      0.67      0.75     45095

---------------------------------
                   Predicted No Disease  Predicted Disease
Actual No Disease                 28274              13652
Actual Disease                     1104               2065
---------------------------------
ROC-AUC score: 0.6634858744569821
---------------------------------
Balanced accuracy: 0.6630018927546795
---------------------------------
```

From the graph, we can indicate that the model on train dataset works much better than the test set, unlike the distribution of the logistic regression which works similarly.



From the Precision-Recall curve, the model does not give an optimistic result from the model, the precision-recall curve here is a straight line that went down, it indicates that the model is not able to distinguish between the positive and negative classes very well. In other words, the model is not very precise in identifying the positive class, and as the recall increases, the precision decreases. This occurs because the positive class is very rare in the dataset, or if the features used by the model are not informative enough to distinguish between the positive and negative classes. It is also a sign that the model is overfitting to the training data and not generalizing well to new data.



In summary, the decision tree model showed higher recall and lower precision for the positive class. In comparison with the logistic Regression, Decision Tree works worse with lower F1 score, and the precision to indicate the positive class is also less than the logistic Regression.

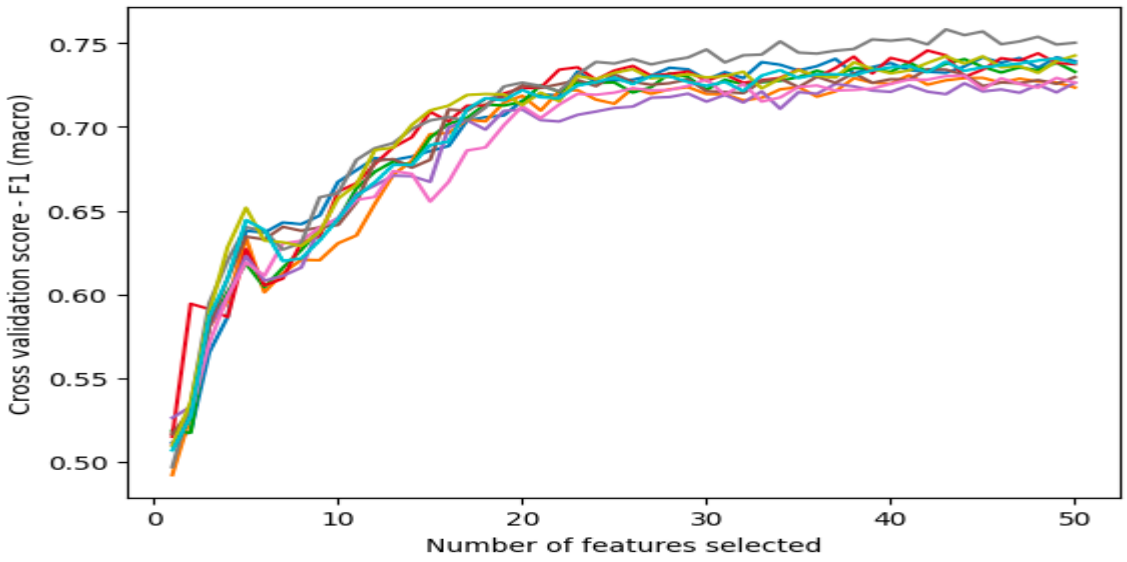
4.3 Random Forest

RFECV

The number of features selected by the random forest algorithm was 43, which is slightly more than the logistic regression model. The cross-validation scores obtained from the random forest model ranged between 0.719 and 0.758, indicating that the model is performing moderately well in predicting the presence of heart disease.

```
Number of features selected: 43

Selected features: ['BMI', 'PhysicalHealth', 'MentalHealth', 'SleepTime',
'Smoking_No', 'Smoking_Yes', 'AlcoholDrinking_Yes', 'Stroke_No', 'Stroke_Yes',
'DiffWalking_No', 'DiffWalking_Yes', 'Sex_Female', 'Sex_Male', 'AgeCategory_18-24',
'AgeCategory_25-29', 'AgeCategory_30-34', 'AgeCategory_35-39', 'AgeCategory_40-44',
'AgeCategory_45-49', 'AgeCategory_50-54', 'AgeCategory_55-59', 'AgeCategory_60-64',
'AgeCategory_65-69', 'AgeCategory_70-74', 'AgeCategory_75-79', 'AgeCategory_80 or
older', 'Race_Black', 'Race_Hispanic', 'Race_White', 'Diabetic_No', 'Diabetic_Yes',
'PhysicalActivity_No', 'PhysicalActivity_Yes', 'GenHealth_Excellent',
'GenHealth_Fair', 'GenHealth_Good', 'GenHealth_Very good', 'Asthma_No', 'Asthma_Yes',
'KidneyDisease_No', 'KidneyDisease_Yes', 'SkinCancer_No', 'SkinCancer_Yes']

Cross-validation score:
[0.73234942 0.72785062 0.73768522 0.74276655 0.71956437 0.73420739 0.73035926
0.75826772 0.74224344 0.73894325]
```

Upon the cross validation given below, random forest did not experience a dramatic drop and the performance is better when more features are selected.



Performance

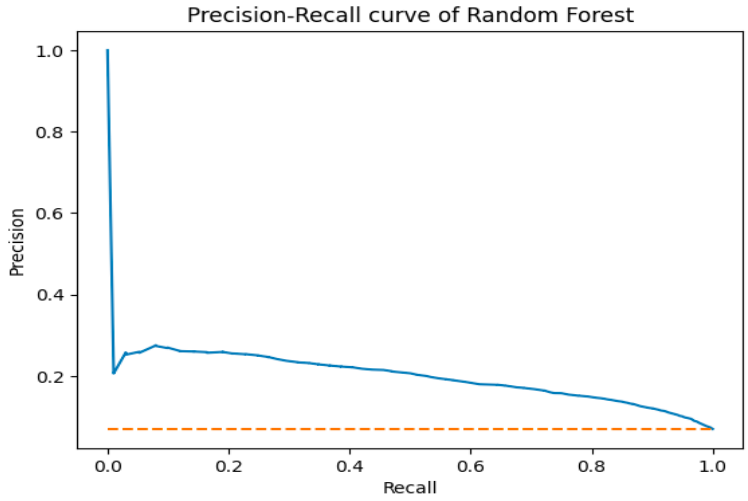
The model achieved a training accuracy of 0.74, with a precision of 0.95 and a recall of 0.74 for the majority class (no disease) and a precision of 0.21 and a recall of 1.0 for the minority class (disease). The f1-score for the minority class was 0.35, indicating that the model struggled to accurately predict the presence of disease.

On the test set, which was imbalanced, the model achieved an accuracy of 0.71, with a precision of 0.92 and a recall of 0.71 for the majority class and a precision of 0.16 and a recall of 0.74 for the minority class. The f1-score for the minority class was 0.26, which is lower than the f1-score for the training set, indicating that the model performed worse on the test set. The ROC-AUC score for the test set was 0.79, indicating that the model was able to distinguish between the two classes with reasonable accuracy.

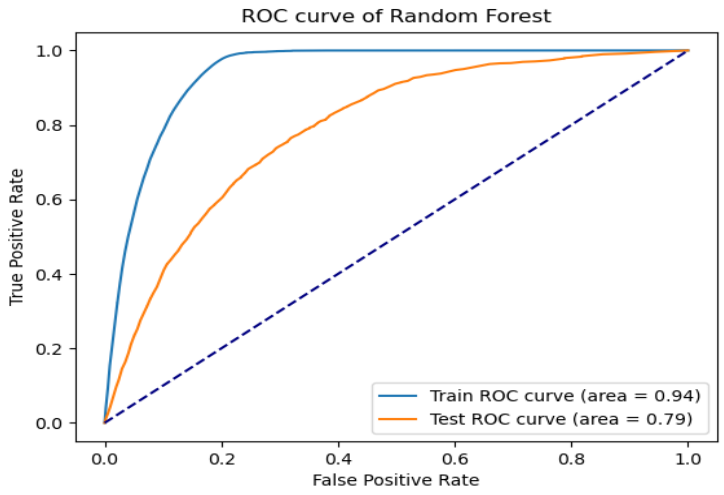
However, it is important to note that the balanced accuracy, which takes into account the imbalanced nature of the test set, was 0.72, which is lower than the overall accuracy. This suggests that the model may be biased towards the majority class, and may not be as effective at detecting the minority class in real-world scenarios.

```
Train:
              precision    recall  f1-score   support

           0       1.00      0.73      0.84    167914
           1       0.21      1.00      0.35     12462

    accuracy                           0.74    180376
   macro avg       0.61      0.86      0.60    180376
weighted avg       0.95      0.74      0.81    180376

---------------------------------
ROC AUC score: 0.9394879628419884
---------------------------------
```

```
Test:
              precision    recall  f1-score   support

           0       0.97      0.70      0.82     41926
           1       0.16      0.74      0.26      3169

    accuracy                           0.71     45095
   macro avg       0.57      0.72      0.54     45095
weighted avg       0.92      0.71      0.78     45095

---------------------------------
                    Predicted No Disease  Predicted Disease
Actual No Disease                  29480              12446
Actual Disease                       829               2340
---------------------------------
ROC-AUC score: 0.7897017520854901
---------------------------------
Balanced accuracy: 0.7207734579071057
```



The precision-Recall Curve indicated the same situation as logistic regression in which the precision decrease as the Recall decline. Overall, the model performed barely precision just barely above the baseline for most threshold.



4 Summary and conclusions

In conclusion, the heart disease prediction task using machine learning algorithms was explored in this study. Four popular classifiers were implemented, including logistic regression, random forest, decision tree, and multi-layer perceptron, and their performances were evaluated using F1 score as the metric.

From the models I have evaluated, logistic regression was found to perform the best among the four classifiers with an F1-score of 0.57, followed by random forest with an F1-score of 0.54 and Decision Tree is the worse out of the three models.

The performance of our model did not give optimal result. However, I have researched and learned different methods like dimension reduction (FAMD) or tuning parameters that may improve out model performance. And I tried to implement in our model to see if they give a more optimal result, It could be seen that FAMD improve the training models a lot, but it is overfitting in this way since the performance on our test data only had minor changes.

5 Calculate the percentage of the code

Total lines: 173 lines

Blanks and comments: 46 lines

Packages imported:17 lines

Code from the Internet: 30 lines

Own codes: 80

Modified Codes:18

Percentage: 10.9%