IBM SUPERVISED CLASSIFICATION ML FINAL ASSIGNMENT

- Main objective of the analysis that specifies whether your model will be focused on prediction or interpretation.

  To predict if a given patient is likely/prone to heart disease based on the given features with a high degree of accuracy using different types of classification models.

- Brief description of the data set you chose and a summary of its attributes.

  The dataset concerns the various medical attributes of patients with respect to their cardiovascular health. Easy detection of cardiovascular disease can greatly increase the probability of survival; hence a ML model could be of great use.

  The attributes include

  - age: age of the patient (years)
  - anaemia: decrease of red blood cells or hemoglobin (boolean)
  - high blood pressure: if the patient has hypertension (boolean)
  - creatinine phosphokinase (CPK): level of the CPK enzyme in the blood (mcg/L)
  - diabetes: if the patient has diabetes (boolean)
  - ejection fraction: percentage of blood leaving the heart at each contraction (percentage)
  - platelets: platelets in the blood (kiloplatelets/mL)
  - sex: woman or man (binary)
  - serum creatinine: level of serum creatinine in the blood (mg/dL)
  - serum sodium: level of serum sodium in the blood (mEq/L)
  - smoking: if the patient smokes or not (boolean)
  - time: follow-up period (days)
  - [target] death event: if the patient deceased during the follow-up period (boolean)

  Target variable is HeartDisease, determining whether anybody is likely to get hearth disease based on the input parameters like gender, age and various test results or not.

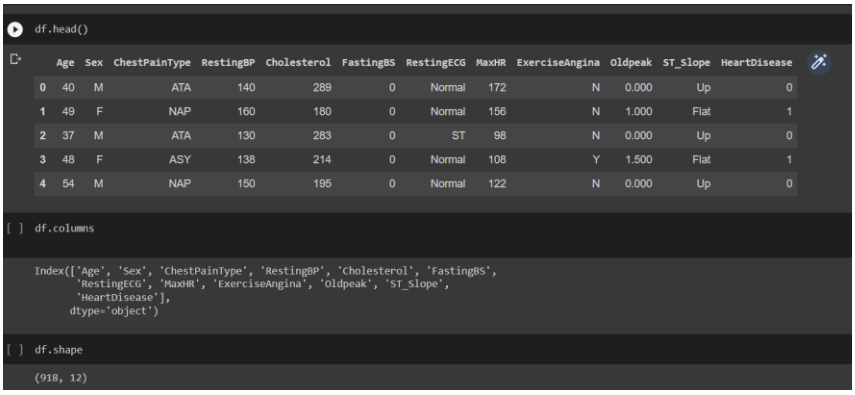
  Source – https://archive.ics.uci.edu/ml/datasets/Heart+failure+clinical+records

  Context - Cardiovascular diseases (CVDs) are the number 1 cause of death globally, taking an estimated 17.9 million lives each year, which accounts for 31% of all deaths worlwide.
  Heart failure is a common event caused by CVDs and this dataset contains 12 features that can be used to predict mortality by heart failure.

- Brief summary of data exploration and actions taken for data cleaning and feature engineering.

  Data exploration includes –

  - ➔ Shape of dataset – 918 observations, 12 attributes
  - ➔ Checking null values – 0
  - ➔ Describing data
  - ➔ Dummy variable operation
  - ➔ Feature scaling



```
df.head()
```

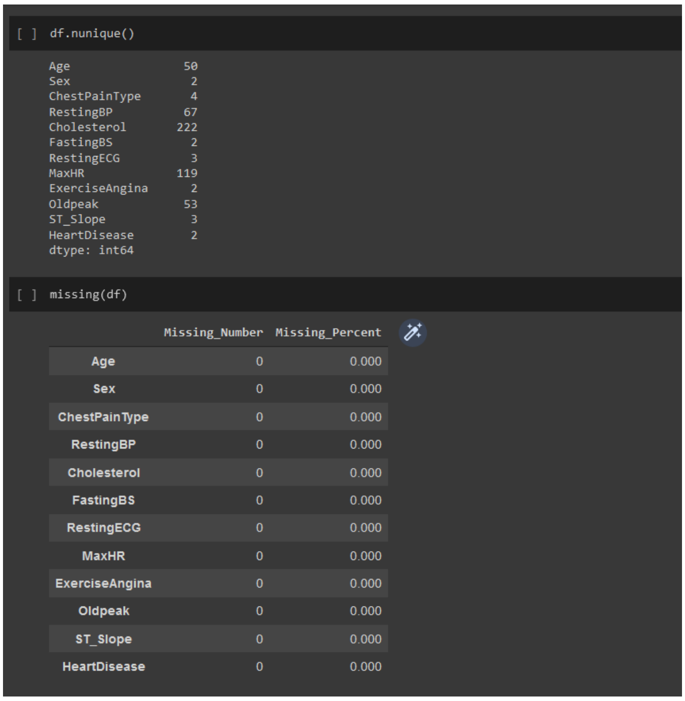
	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.000	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.000	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.000	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.500	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.000	Up	0

```
[ ] df.columns
```

```
Index(['Age', 'Sex', 'ChestPainType', 'RestingBP', 'Cholesterol', 'FastingBS',
       'RestingECG', 'MaxHR', 'ExerciseAngina', 'Oldpeak', 'ST_Slope',
       'HeartDisease'],
      dtype='object')
```

```
[ ] df.shape
```

```
(918, 12)
```

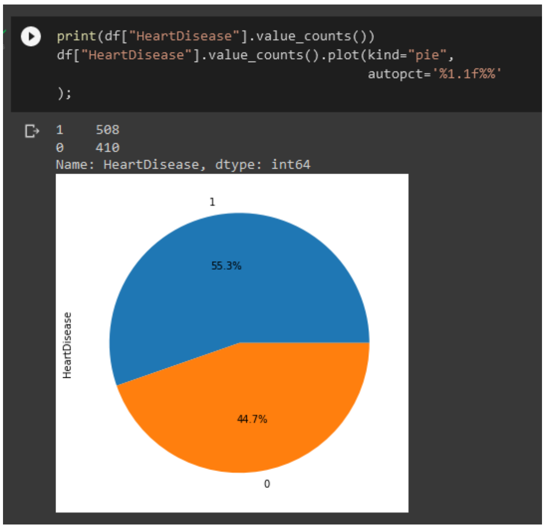


```
[ ] df.nunique()
```

```
Age                50
Sex                 2
ChestPainType       4
RestingBP          67
Cholesterol       222
FastingBS           2
RestingECG          3
MaxHR             119
ExerciseAngina      2
Oldpeak            53
ST_Slope            3
HeartDisease        2
dtype: int64
```

```
[ ] missing(df)
```

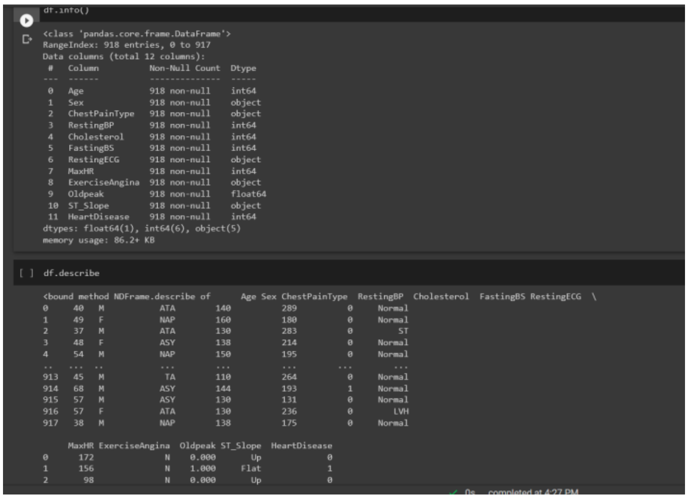
	Missing_Number	Missing_Percent
Age	0	0.000
Sex	0	0.000
ChestPainType	0	0.000
RestingBP	0	0.000
Cholesterol	0	0.000
FastingBS	0	0.000
RestingECG	0	0.000
MaxHR	0	0.000
ExerciseAngina	0	0.000
Oldpeak	0	0.000
ST_Slope	0	0.000
HeartDisease	0	0.000



```
print(df["HeartDisease"].value_counts())
df["HeartDisease"].value_counts().plot(kind="pie",
                                       autopct='%1.1f%%'

);
```

```
1    508
0    410
Name: HeartDisease, dtype: int64
```



```
df.info()
```

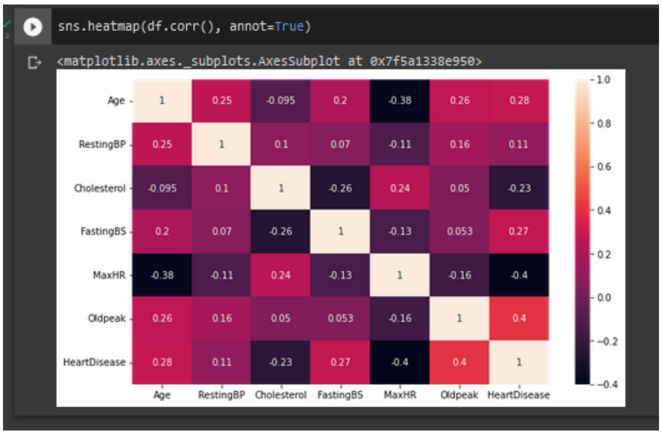
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 918 entries, 0 to 917
Data columns (total 12 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Age             918 non-null    int64  
 1   Sex             918 non-null    object 
 2   ChestPainType   918 non-null    object 
 3   RestingBP       918 non-null    int64  
 4   Cholesterol     918 non-null    int64  
 5   FastingBS       918 non-null    int64  
 6   RestingECG      918 non-null    object 
 7   MaxHR           918 non-null    int64  
 8   ExerciseAngina  918 non-null    object 
 9   Oldpeak         918 non-null    float64
 10  ST_Slope        918 non-null    object 
 11  HeartDisease    918 non-null    int64  
dtypes: float64(1), int64(6), object(5)
memory usage: 86.2+ KB
```

```
[ ] df.describe
```

```
<bound method NDFrame.describe of      Age Sex ChestPainType  RestingBP  Cholesterol  FastingBS RestingECG  \
0     40   M           ATA        140          289          0     Normal   
1     49   F           NAP        160          180          0     Normal   
2     37   M           ATA        130          283          0         ST   
3     48   F           ASY        138          214          0     Normal   
4     54   M           NAP        150          195          0     Normal   
..   ...  ..           ...        ...          ...        ...        ...   
913   45   M            TA        110          264          0     Normal   
914   68   M           ASY        144          193          1     Normal   
915   57   M           ASY        130          131          0     Normal   
916   57   F           ATA        130          236          0        LVH   
917   38   M           NAP        138          175          0     Normal   

     MaxHR ExerciseAngina  Oldpeak ST_Slope  HeartDisease  
0      172              N    0.000       Up             0  
1      156              N    1.000     Flat             1  
2       98              N    0.000       Up             0  
```

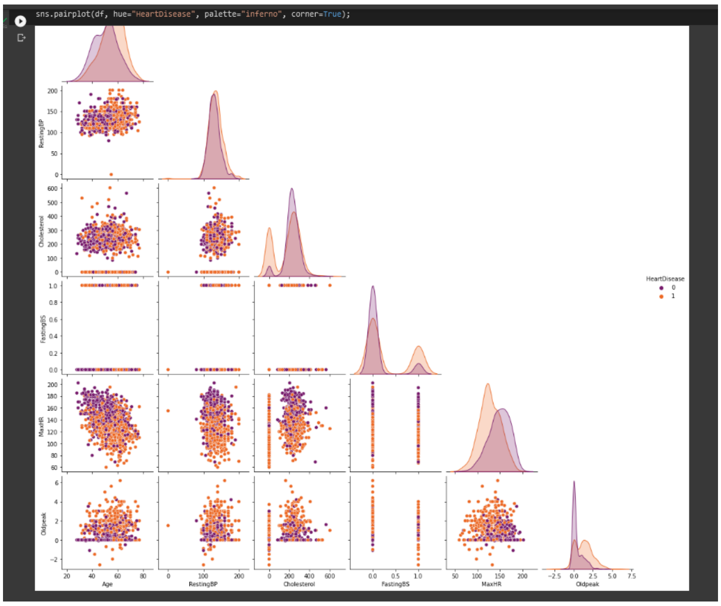
✓ 0s completed at 4:27 PM



```
sns.heatmap(df.corr(), annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5a1338e950>
```

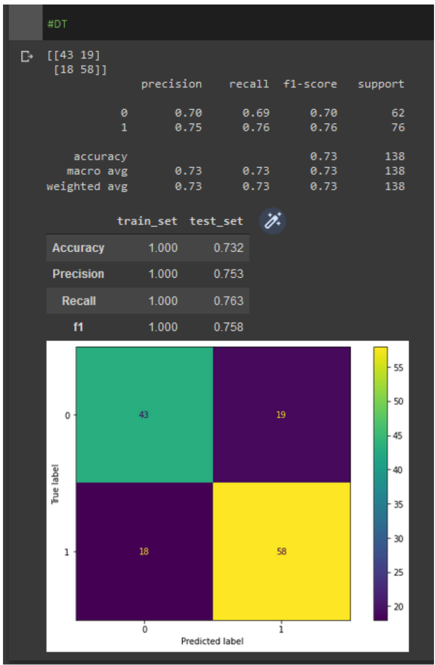
	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
Age	1	0.25	-0.095	0.2	-0.38	0.26	0.28
RestingBP	0.25	1	0.1	0.07	-0.11	0.16	0.11
Cholesterol	-0.095	0.1	1	-0.26	0.24	0.05	-0.23
FastingBS	0.2	0.07	-0.26	1	-0.13	0.053	0.27
MaxHR	-0.38	-0.11	0.24	-0.13	1	-0.16	-0.4
Oldpeak	0.26	0.16	0.05	0.053	-0.16	1	0.4
HeartDisease	0.28	0.11	-0.23	0.27	-0.4	0.4	1



```
sns.pairplot(df, hue="HeartDisease", palette="inferno", corner=True);
```

- Summary of training at least three different classifier models, preferably of different nature in explainability and predictability. For example, you can start with a simple logistic regression as a baseline, adding other models or ensemble models. Preferably, all your models use the same training and test splits, or the same cross-validation method.

  - ➔ Among employed models – linear, random forest, SVM, adaboost, knn, decision tree, random forest give us the best result for this data while decision tree performs the worst.
  - ➔ All models are built using train test split



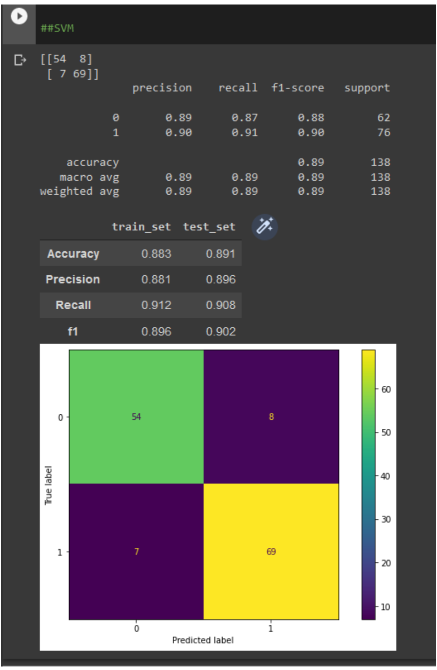
```
#DT
```

```
[[43 19]
 [18 58]]
              precision    recall  f1-score   support

           0       0.70      0.69      0.70        62
           1       0.75      0.76      0.76        76

    accuracy                           0.73       138
   macro avg       0.73      0.73      0.73       138
weighted avg       0.73      0.73      0.73       138
```

	train_set	test_set
Accuracy	1.000	0.732
Precision	1.000	0.753
Recall	1.000	0.763
f1	1.000	0.758



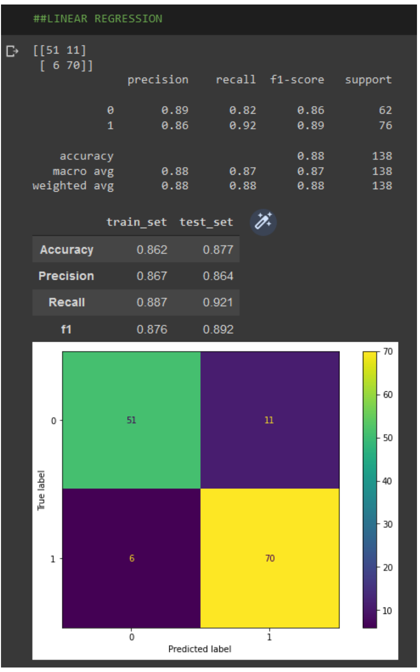
```
##SVM
```

```
[[54  8]
 [ 7 69]]
              precision    recall  f1-score   support

           0       0.89      0.87      0.88        62
           1       0.90      0.91      0.90        76

    accuracy                           0.89       138
   macro avg       0.89      0.89      0.89       138
weighted avg       0.89      0.89      0.89       138
```

	train_set	test_set
Accuracy	0.883	0.891
Precision	0.881	0.896
Recall	0.912	0.908
f1	0.896	0.902



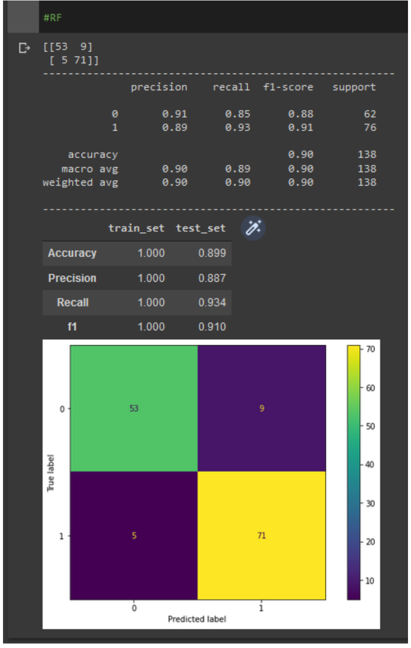
```
##LINEAR REGRESSION
```

```
[[51 11]
 [ 6 70]]
              precision    recall  f1-score   support

           0       0.89      0.82      0.86        62
           1       0.86      0.92      0.89        76

    accuracy                           0.88       138
   macro avg       0.88      0.87      0.87       138
weighted avg       0.88      0.88      0.88       138
```

	train_set	test_set
Accuracy	0.862	0.877
Precision	0.867	0.864
Recall	0.887	0.921
f1	0.876	0.892



```
#RF
```

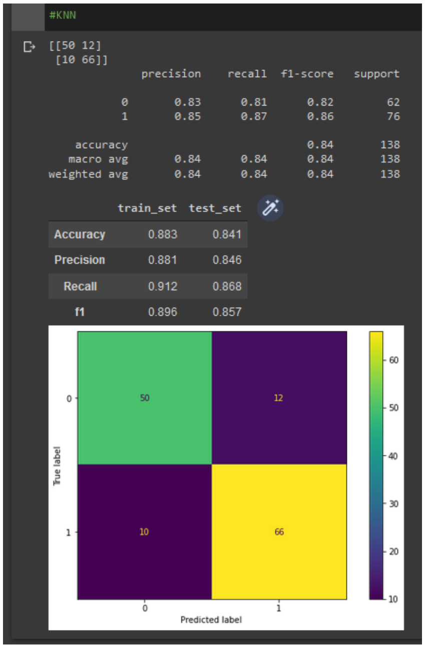
```
[[53  9]
 [ 5 71]]
------------------------------------------------------
              precision    recall  f1-score   support

           0       0.91      0.85      0.88        62
           1       0.89      0.93      0.91        76

    accuracy                           0.90       138
   macro avg       0.90      0.89      0.90       138
weighted avg       0.90      0.90      0.90       138

------------------------------------------------------
```

	train_set	test_set
Accuracy	1.000	0.899
Precision	1.000	0.887
Recall	1.000	0.934
f1	1.000	0.910



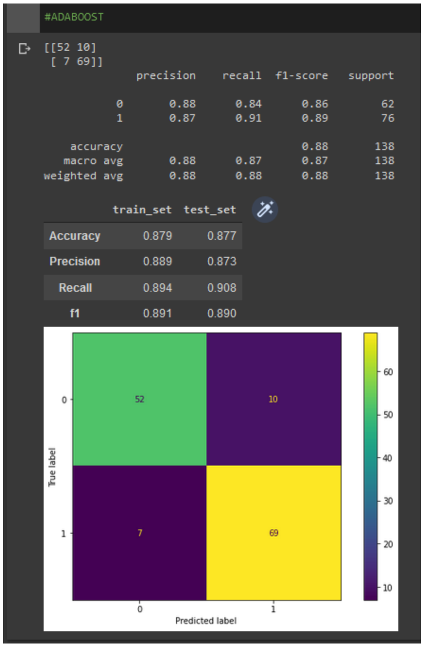
```
#KNN
```

```
[[50 12]
 [10 66]]
              precision    recall  f1-score   support

           0       0.83      0.81      0.82        62
           1       0.85      0.87      0.86        76

    accuracy                           0.84       138
   macro avg       0.84      0.84      0.84       138
weighted avg       0.84      0.84      0.84       138
```

	train_set	test_set
Accuracy	0.883	0.841
Precision	0.881	0.846
Recall	0.912	0.868
f1	0.896	0.857



```
#ADABOOST
```

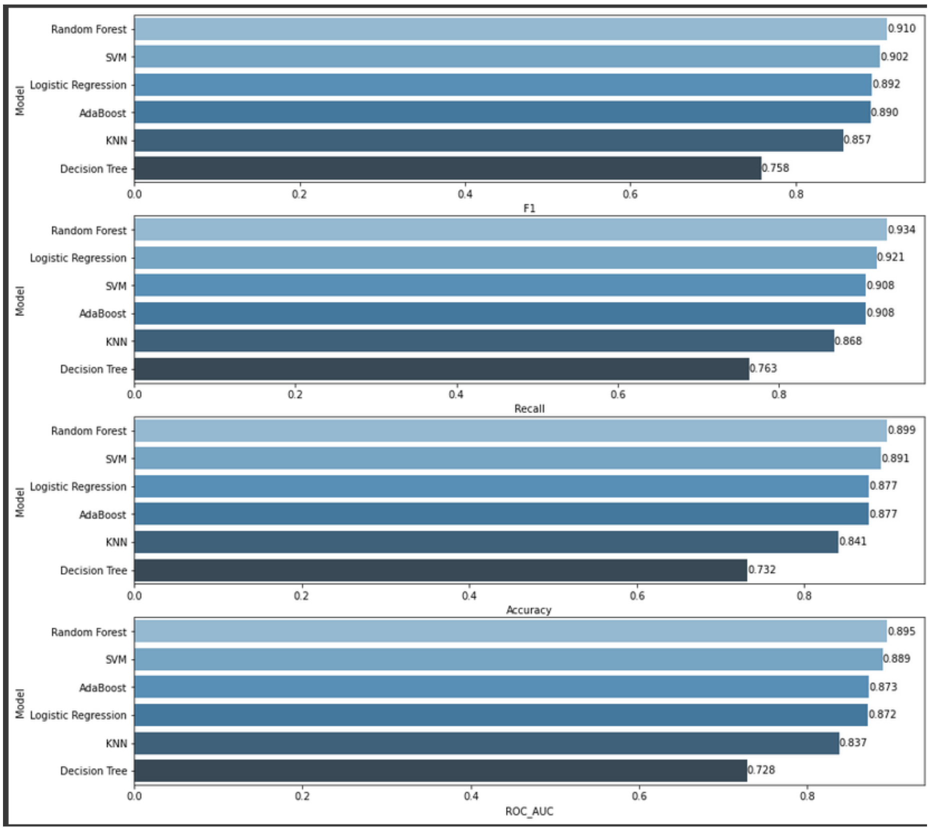
```
[[52 10]
 [ 7 69]]
              precision    recall  f1-score   support

           0       0.88      0.84      0.86        62
           1       0.87      0.91      0.89        76

    accuracy                           0.88       138
   macro avg       0.88      0.87      0.87       138
weighted avg       0.88      0.88      0.88       138
```

	train_set	test_set
Accuracy	0.879	0.877
Precision	0.889	0.873
Recall	0.894	0.908
f1	0.891	0.890

- A paragraph explaining which of your regressions you recommend as a final model that best fits your needs in terms of accuracy and explain ability.
  - ➔ Using Random Forest followed by SVM gives us the best results and predicts the values with high accuracy. Thus, this is the best model of the purpose of classifying the target variable in this dataset.
  - ➔ The performance metrics are given below



- Summary Key Findings and Insights, which walks your reader through the main drivers of your model and insights from your data derived from your linear regression model.

  - ➔ The ST_slope, attributes has a high weightage in the model followed by Chestpaintype ExcerciseAngina, Oldpeak according to feature importance analysis
  - ➔ The final model is overall able to generalise and predict the values thus avoiding overfitting and underfitting

- Suggestions for next steps in analyzing this data, which may include suggesting revisiting this model adding specific data features to achieve a better explanation or a better prediction.

  - ➔ The model is highly simplistic thus, implementing gradient boost, XGBoost and k flod cross validation might improve results.
  - ➔ The current dataset has only a handful of attributes and does not include behavioural metrics like smoking, alcohol, diet etc thus only give a limited scope for interpretation
  - ➔ Using a more updated dataset with more numerous and recent values will yield better interpretation of how factors like smoking, bmi, age affects cardiovascular health.