# Heart Disease

# **Prediction**

# Introduction

The goal of our heart disease prediction notebook is to determine if a patient should be diagnosed with heart disease or not, which is a binary outcome.

For this we use various EDA, data wranging , data visulaization and finally data modelling to predit the ourput.

# features

**Features**

We use the following 13 features (X) to determine our predictor (Y):

1. Age.
2. Sex: 1 = Male, 0 = Female.
3. (cp) chest pain type (4 values – Ordinal), 1st value: typical angina, 2nd value: atypical angina, 3rd value: non-anginal pain, 4th value: asymptomatic.
4. (trestbps) resting blood pressure.
5. (chol) serum cholesterol.
6. (Fbs) – fasting blood sugar > 120 mg/dl.
7. (restecg) – resting electrocardiography results.
8. (thalach) – maximum heart rate achieved.
9. (exang) – exercise-induced angina.
10. (oldpeak) – ST depression caused by exercise relative to rest.
11. (slope) – the slope of the peak exercise ST segment.
12. (ca) – the number of major vessels colored by fluoroscopy.
13. (thal) – maximum heart rate achieved (Ordinal), 3 = normal, 6 = fixed defect, 7 = reversible defect.

# Steps Involved:

1- Data Wrangling

2-Conducting EDA

3- Filtering Data

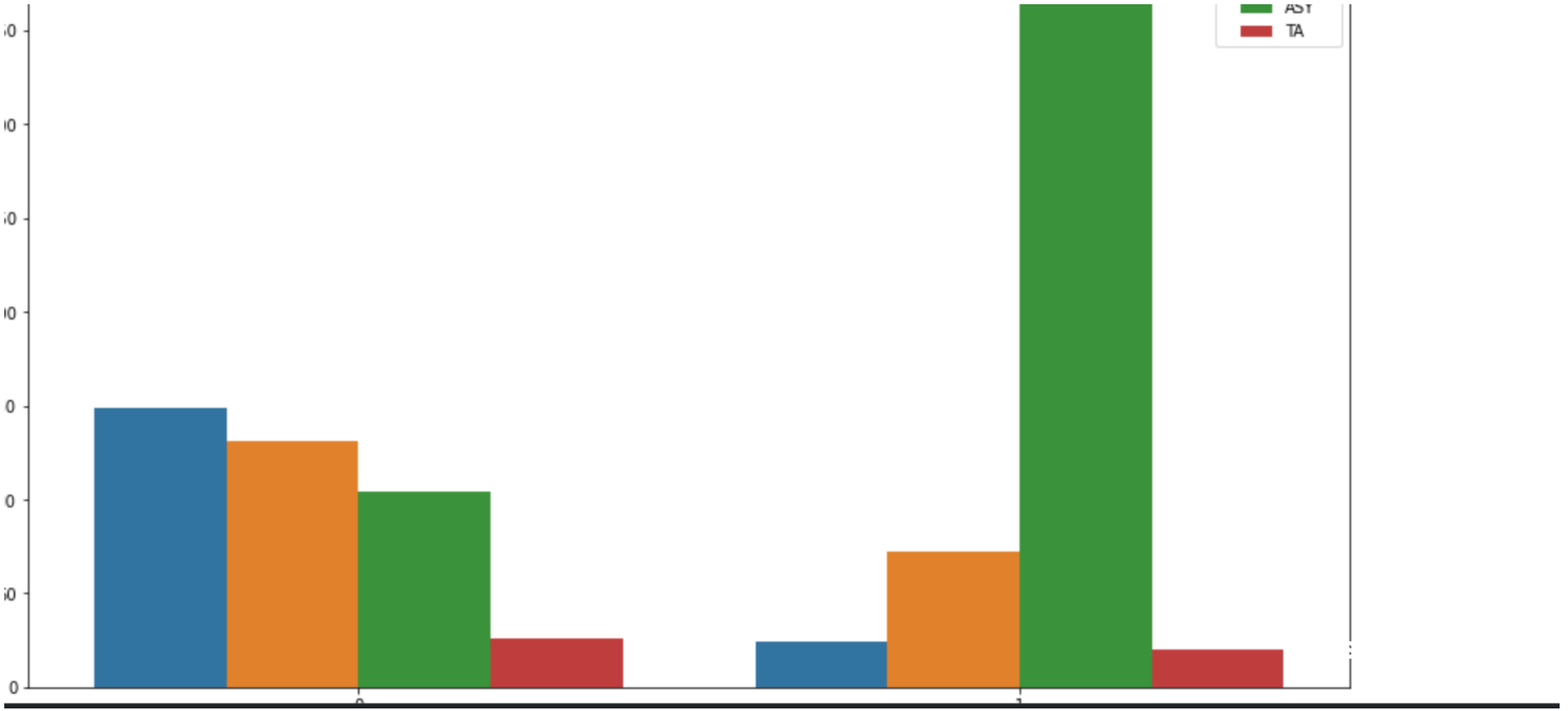
4-  Machine learning model

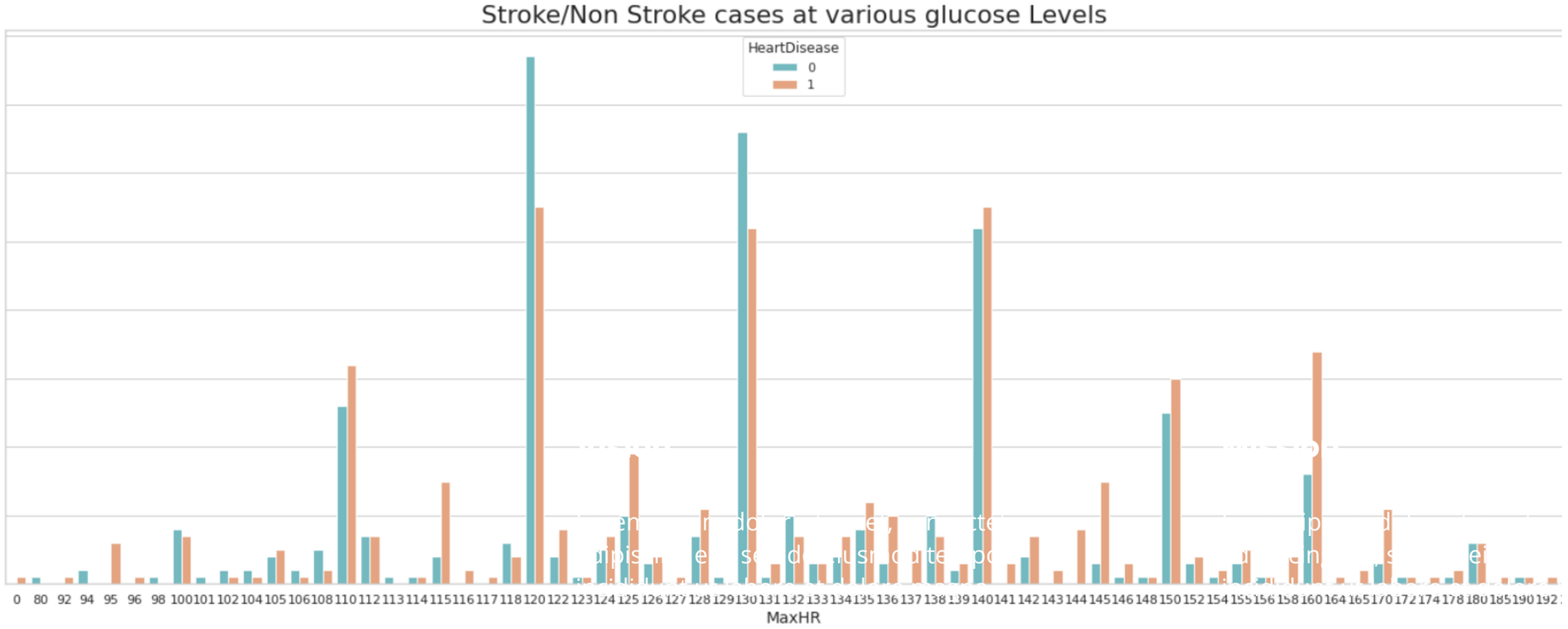
5- Using ANN

```
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      0.0       Up             0
1      156              N      1.0     Flat             1
2       98              N      0.0       Up             0
3      108              Y      1.5     Flat             1
4      122              N      0.0       Up             0
..     ...            ...      ...      ...           ...
913    132              N      1.2     Flat             1
914    141              N      3.4     Flat             1
915    115              Y      1.2     Flat             1
916    174              N      0.0     Flat             1
917    173              N      0.0       Up             0
```

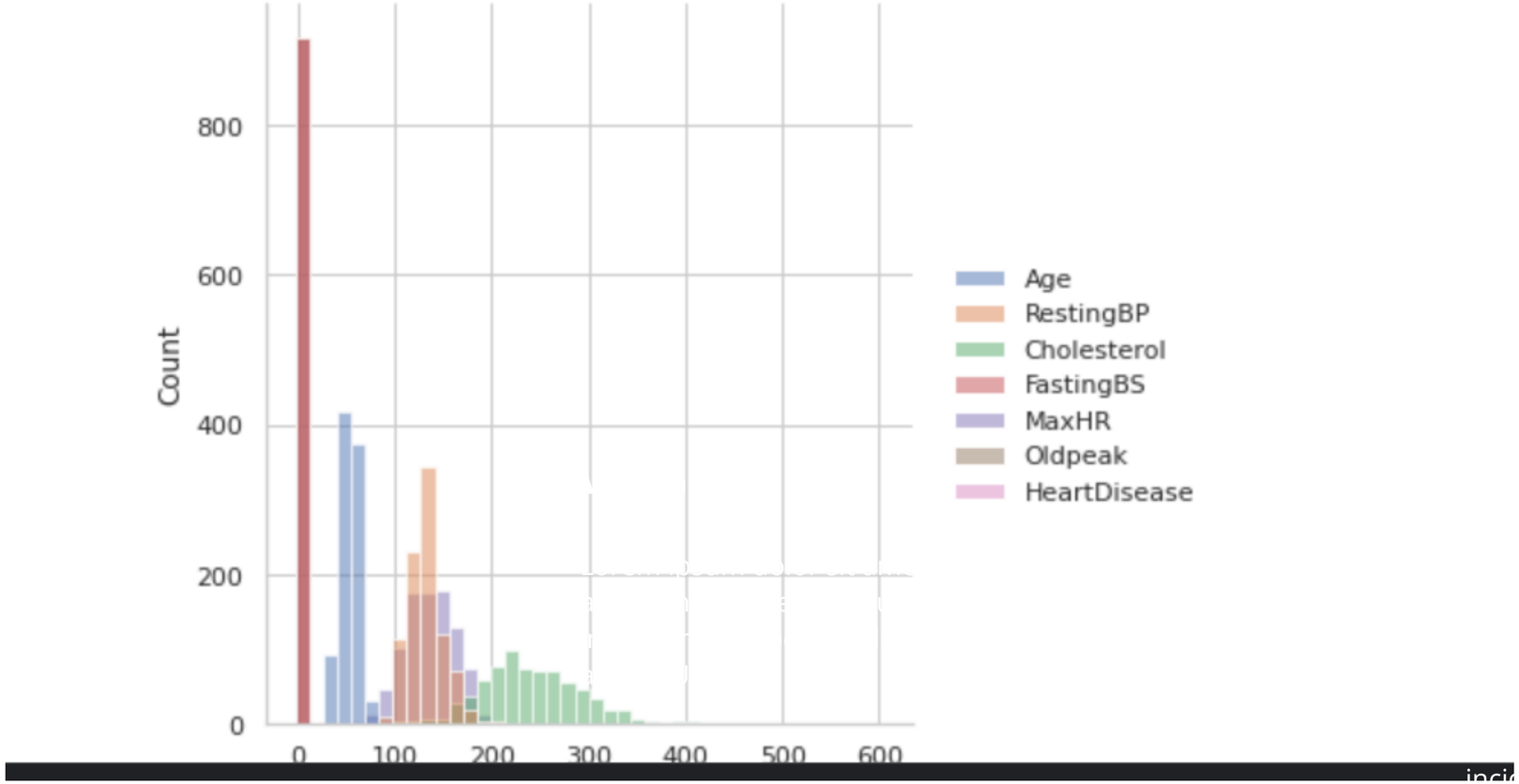

AST
TA
0
1


Stroke/Non Stroke cases at various glucose Levels
HeartDisease
0
1
0 80 92 94 95 96 98 100 101 102 104 105 106 108 110 112 113 114 115 116 117 118 120 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 148 150 152 154 155 156 158 160 164 165 170 172 174 178 180 185 190 192
MaxHR

```
plt.figure(figsize=(15,10))
sns.displot(df)
```

[20]: <seaborn.axisgrid.FacetGrid at 0x7f3f6b77f690>

<Figure size 1080x720 with 0 Axes>



# Label Encoding

```
from sklearn.preprocessing import LabelEncoder,MinMaxScaler
gender_le=LabelEncoder()
df['Sex']=gender_le.fit_transform(df['Sex'])
```

```
df.Sex.value_counts()
```

```
1    725
0    193
Name: Sex, dtype: int64
```

```
ChestPainType_le = LabelEncoder()
df["ChestPainType"] = ChestPainType_le.fit_transform(df["ChestPainType"])
```

# Feature Selection

```
from sklearn.feature_selection import SelectPercentile
from sklearn.feature_selection import chi2,f_classif
```

```
X=df.drop('HeartDisease',axis=1).values
y=df['HeartDisease'].values
```

```
X
```

Vision Mission

```
array([[40. ,  1. ,  1. , ...,  0. ,  0. ,  2. ],
       [49. ,  0. ,  2. , ...,  0. ,  1. ,  1. ],
       [37. ,  1. ,  1. , ...,  0. ,  0. ,  2. ],
```

# Standard scaler

+ Code + Markdown

```
from sklearn.preprocessing import StandardScaler
sca=StandardScaler(copy=True,with_mean=True, with_std=True)
X=sca.fit_transform(X_Sel)
```

```
X
```

```
array([[-1.4331398 ,  0.51595242,  0.22903206, ..., -0.8235563 ,
        -0.83243239,  1.05211381],
       [-0.47848359, -1.93816322,  1.27505906, ..., -0.8235563 ,
         0.10566353, -0.59607813],
       [-1.75135854,  0.51595242,  0.22903206, ..., -0.8235563 ,
        -0.83243239,  1.05211381],
       ...,
       [ 0.37009972,  0.51595242, -0.81699495, ...,  1.21424608,
         0.29328271, -0.59607813],
```

# Data Splitting

```
y.shape
```

```
(918,)
```

```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=42,shuffle=True)
```

Modelling Logistic Regression

# *Data Modeling*

- SVM
- DECISION TREE
- LOGISTIC REGRESSION
- RANDOM FOREST
- ANN

# *Data Modeling*

- SVM- **Accuracy(0.85)**
- DECISION TREE **Accuracy(0.82)**
- LOGISTIC REGRESSION **Accuracy(0.86)**
- RANDOM FOREST **Accuracy(0.89)**
- ANN **Accuracy(0.86)**

# Thank You

Github Repository