Heart Disease Prediction Modelling Using Python

Table of Contents

01

Introduction

- Problem Statement
- Primary End User

Introduction

- The healthcare industry collects a **vast amount** of **confidential data**; we can use advanced data mining techniques to provide relevant results and make effective decisions on data.

- In this project, we will **establish** a **relationship** and knowledge between various medical factors related to heart disease and the patterns.

- We will implement various Machine Learning techniques and algorithms to **effectively predict** heart disease risk.

Primary End Users

- **Insurance Companies:** To analyze insurance premiums and make recommendations.
- **Reports & Published Journals:** Detailed Analysis and predictions regarding the heart disease data.
- **Private & Public Health Department:** So that they can closely collaborate with neighborhood health care professionals to stop the spread of the disease and raise required awareness.

02

Exploratory Data Analytics

- Data Description & Transformation
- Feature Analysis
- Data Visualization

Data Description & Transformation

- Heart disease is the **leading cause** of **death**; about half of Americans have at least one of the three major risk factors for heart disease: high blood pressure, high cholesterol, and smoking. Other indicators include diabetes status, obesity (high BMI), physical inactivity, or excessive alcohol consumption.

- **Identifying** and **preventing** the factors that have the most significant **impact** on heart disease is very important in healthcare.

- In turn, the development of computers allowed machine learning methods to identify "**patterns**" from data that could **predict** a patient's condition.

Data Description

The system uses **18 medical parameters** such as Age Category, Sex, Diabetic, Sleep Time, BMI, Smoking, Alcohol, Physical Health, Mental Health, and Race, to name a few.

Categorical / Boolean Variables in the Data:
Heart Disease, Smoking, Alcohol Drinking, Stroke, Difficulty in Walking, Diabetic, Physical Activity, Asthma, Kidney Disease, Skin Cancer.

Numerical Variables in the Data:
Age Category, BMI, Physical Health, Mental Health, Sleep Time.

Strings in the Data:
Race, General Health

Data Transformation

- **Creation of Dummy Variables:**

We have used Pandas using get_dummies.

- **Handling Null Values:**

We have removed all the null values before our analysis and model learning part.

- **Reducing Dimensionality & Variance using feature analysis:**

We have analyzed and removed dimensionality by using correlation.

- **Removed Outliers for better model learning:**

Identified Outliers and released them for better analytics.

Feature Analysis

Analysis of the Correlation:

The dummy variables made have been used to do the feature analysis and their correlation values are calculated using the corr() function.

```
[ ] EDA_HD.corr()
```

	BMI	PhysicalHealth	MentalHealth	SleepTime	HeartDisease_No	HeartDisease_Yes	Smoking_No	Smoking_Yes
BMI	1.000000	0.109788	0.064131	-0.051822	-0.051803	0.051803	-0.023118	0.023118
PhysicalHealth	0.109788	1.000000	0.287987	-0.061387	-0.170721	0.170721	-0.115352	0.115352
MentalHealth	0.064131	0.287987	1.000000	-0.119717	-0.028591	0.028591	-0.085157	0.085157
SleepTime	-0.051822	-0.061387	-0.119717	1.000000	-0.008327	0.008327	0.030336	-0.030336
HeartDisease_No	-0.051803	-0.170721	-0.028591	-0.008327	1.000000	-1.000000	0.107764	-0.107764
HeartDisease_Yes	0.051803	0.170721	0.028591	0.008327	-1.000000	1.000000	-0.107764	0.107764
Smoking_No	-0.023118	-0.115352	-0.085157	0.030336	0.107764	-0.107764	1.000000	-1.000000
Smoking_Yes	0.023118	0.115352	0.085157	-0.030336	-0.107764	0.107764	-1.000000	1.000000
AlcoholDrinking_No	0.038816	0.017254	-0.051282	0.005065	-0.032080	0.032080	0.111768	-0.111768
AlcoholDrinking_Yes	-0.038816	-0.017254	0.051282	-0.005065	0.032080	-0.032080	-0.111768	0.111768
Stroke_No	-0.019733	-0.137014	-0.046467	-0.011900	0.196835	-0.196835	0.061226	-0.061226
Stroke_Yes	0.019733	0.137014	0.046467	0.011900	-0.196835	0.196835	-0.061226	0.061226
DiffWalking_No	-0.181678	-0.428373	-0.152235	0.022216	0.201258	-0.201258	0.120074	-0.120074
DiffWalking_Yes	0.181678	0.428373	0.152235	-0.022216	-0.201258	0.201258	-0.120074	0.120074
Sex_Female	-0.026940	0.040904	0.100058	0.015704	0.070040	-0.070040	0.085052	-0.085052
Sex_Male	0.026940	-0.040904	-0.100058	-0.015704	-0.070040	0.070040	-0.085052	0.085052
AgeCategory_18-24	-0.107060	-0.055866	0.075243	0.016524	0.075385	-0.075385	0.138397	-0.138397
AgeCategory_25-29	-0.023705	-0.046707	0.054452	-0.018231	0.065759	-0.065759	0.052149	-0.052149
AgeCategory_30-34	0.004500	-0.042484	0.043741	-0.039005	0.065611	-0.065611	0.015226	-0.015226
AgeCategory_35-39	0.021160	-0.037248	0.037929	-0.044187	0.066685	-0.066685	-0.004290	0.004290
AgeCategory_40-44	0.036475	-0.026575	0.025892	-0.040646	0.059196	-0.059196	-0.010680	0.010680
AgeCategory_45-49	0.049427	-0.011934	0.016566	-0.036350	0.049733	-0.049733	0.006637	-0.006637
AgeCategory_50-54	0.050800	0.008711	0.015627	-0.035356	0.032648	-0.032648	0.011667	-0.011667
AgeCategory_55-59	0.038984	0.026416	0.006345	-0.029351	0.013276	-0.013276	-0.008701	0.008701
AgeCategory_60-64	0.026797	0.040827	-0.015002	-0.009073	-0.016152	0.016152	-0.031892	0.031892
AgeCategory_65-69	0.019006	0.021009	-0.043933	0.025318	-0.042626	0.042626	-0.030367	0.030367
AgeCategory_70-74	-0.007720	0.022623	-0.055078	0.047893	-0.082578	0.082578	-0.045288	0.045288
AgeCategory_75-79	-0.030726	0.027203	-0.054581	0.058898	-0.098690	0.098690	-0.048040	0.048040
AgeCategory_80 or older	-0.094780	0.039621	-0.071718	0.088321	-0.143041	0.143041	-0.013569	0.013569
Race_American Indian/Alaskan Native	0.026347	0.022955	0.018394	-0.003615	-0.008547	0.008547	-0.035667	0.035667
Race_Asian	-0.078643	-0.035229	-0.023113	-0.019985	0.030262	-0.030262	0.060308	-0.060308

	AlcoholDrinking_No	AlcoholDrinking_Yes	...	GenHealth_Fair	GenHealth_Good	GenHealth_Poor	GenHealth_Very good	Asthma_No	Asthma_Yes	KidneyDisease_No	KidneyDisease_Yes	SkinCancer_No	SkinCancer_Yes
BMI	0.038816	-0.038816	...	0.127364	0.118047	0.062501	-0.065954	-0.092345	0.092345	-0.050768	0.050768	0.033644	-0.033644
PhysicalHealth	0.017254	-0.017254	...	0.303773	-0.037663	0.471919	-0.196462	-0.117907	0.117907	-0.142197	0.142197	-0.041700	0.041700
MentalHealth	-0.051282	0.051282	...	0.151321	0.013353	0.192079	-0.089956	-0.114008	0.114008	-0.037281	0.037281	0.033412	-0.033412
SleepTime	0.005065	-0.005065	...	-0.040923	-0.013725	-0.033074	0.019379	0.048245	-0.048245	-0.006238	0.006238	-0.041266	0.041266
HeartDisease_No	-0.032080	0.032080	...	-0.147954	-0.039033	-0.174662	0.101886	0.041444	-0.041444	0.145197	-0.145197	0.093317	-0.093317
HeartDisease_Yes	0.032080	-0.032080	...	0.147954	0.039033	0.174662	-0.101886	-0.041444	0.041444	-0.145197	0.145197	-0.093317	0.093317
Smoking_No	0.111768	-0.111768	...	-0.095620	-0.059651	-0.086520	0.052305	0.024149	-0.024149	0.034920	-0.034920	0.033977	-0.033977
Smoking_Yes	-0.111768	0.111768	...	0.095620	0.059651	0.086520	-0.052305	-0.024149	0.024149	-0.034920	0.034920	-0.033977	0.033977
AlcoholDrinking_No	1.000000	-1.000000	...	0.018859	0.007808	0.017068	-0.013005	-0.002202	0.002202	-0.028280	0.028280	-0.005702	0.005702
AlcoholDrinking_Yes	-1.000000	1.000000	...	-0.018859	-0.007808	-0.017068	0.013005	0.002202	-0.002202	0.028280	-0.028280	0.005702	-0.005702
Stroke_No	-0.019858	0.019858	...	-0.104983	-0.013159	-0.133641	0.069395	0.038866	-0.038866	0.091167	-0.091167	0.048116	-0.048116
Stroke_Yes	0.019858	-0.019858	...	0.104983	0.013159	0.133641	-0.069395	-0.038866	0.038866	-0.091167	0.091167	-0.048116	0.048116
DiffWalking_No	-0.035328	0.035328	...	-0.282517	-0.031570	-0.308767	0.184986	0.103222	-0.103222	0.153064	-0.153064	0.064840	-0.064840
DiffWalking_Yes	0.035328	-0.035328	...	0.282517	0.031570	0.308767	-0.184986	-0.103222	0.103222	-0.153064	0.153064	-0.064840	0.064840
Sex_Female	0.004200	-0.004200	...	0.022456	-0.003642	0.010667	0.003239	-0.069191	0.069191	-0.009084	0.009084	0.013434	-0.013434
Sex_Male	-0.004200	0.004200	...	-0.022456	0.003642	-0.010667	-0.003239	0.069191	-0.069191	0.009084	-0.009084	-0.013434	0.013434
AgeCategory_18-24	-0.004334	0.004334	...	-0.047034	-0.029306	-0.040685	0.018047	-0.033416	0.033416	0.043093	-0.043093	0.082247	-0.082247
AgeCategory_25-29	-0.023069	0.023069	...	-0.037142	-0.017925	-0.032709	0.013041	-0.024371	0.024371	0.037750	-0.037750	0.071893	-0.071893
AgeCategory_30-34	-0.015902	0.015902	...	-0.034864	-0.013802	-0.031300	0.012599	-0.012928	0.012928	0.037219	-0.037219	0.072759	-0.072759
AgeCategory_35-39	-0.021545	0.021545	...	-0.029789	-0.011409	-0.029053	0.006085	-0.004643	0.004643	0.033914	-0.033914	0.072501	-0.072501
AgeCategory_40-44	-0.018818	0.018818	...	-0.021467	-0.008870	-0.021787	0.005039	-0.009222	0.009222	0.027323	-0.027323	0.067055	-0.067055
AgeCategory_45-49	-0.009857	0.009857	...	-0.011371	-0.008464	-0.011851	0.001805	-0.007782	0.007782	0.023167	-0.023167	0.053725	-0.053725
AgeCategory_50-54	-0.010588	0.010588	...	-0.005852	-0.005515	0.004513	-0.001180	-0.002352	0.002352	0.014427	-0.014427	0.042876	-0.042876
AgeCategory_55-59	-0.008189	0.008189	...	0.010708	-0.008782	0.017940	-0.002574	0.001461	-0.001461	0.005603	-0.005603	0.021718	-0.021718
AgeCategory_60-64	-0.001621	0.001621	...	0.024054	0.002738	0.027481	-0.010070	0.000537	-0.000537	-0.007098	0.007098	-0.006900	0.006900
AgeCategory_65-69	0.009026	-0.009026	...	0.021388	0.013075	0.012698	0.000424	0.013619	-0.013619	-0.023175	0.023175	-0.049571	0.049571
AgeCategory_70-74	0.021730	-0.021730	...	0.026606	0.015349	0.021192	-0.001791	0.015297	-0.015297	-0.046293	0.046293	-0.096897	0.096897
AgeCategory_75-79	0.027861	-0.027861	...	0.028147	0.024777	0.021436	-0.007261	0.017448	-0.017448	-0.053572	0.053572	-0.121747	0.121747
AgeCategory_80 or older	0.045226	-0.045226	...	0.050048	0.038115	0.037282	-0.027169	0.034488	-0.034488	-0.067691	0.067691	-0.162096	0.162096
Race_American Indian/Alaskan Native	0.004243	-0.004243	...	0.022334	0.015839	0.022017	-0.025611	-0.013757	0.013757	-0.007401	0.007401	0.026784	-0.026784
Race_Asian	0.022275	-0.022275	...	-0.024360	0.003752	-0.018024	-0.003102	0.017007	-0.017007	0.016957	-0.016957	0.047749	-0.047749

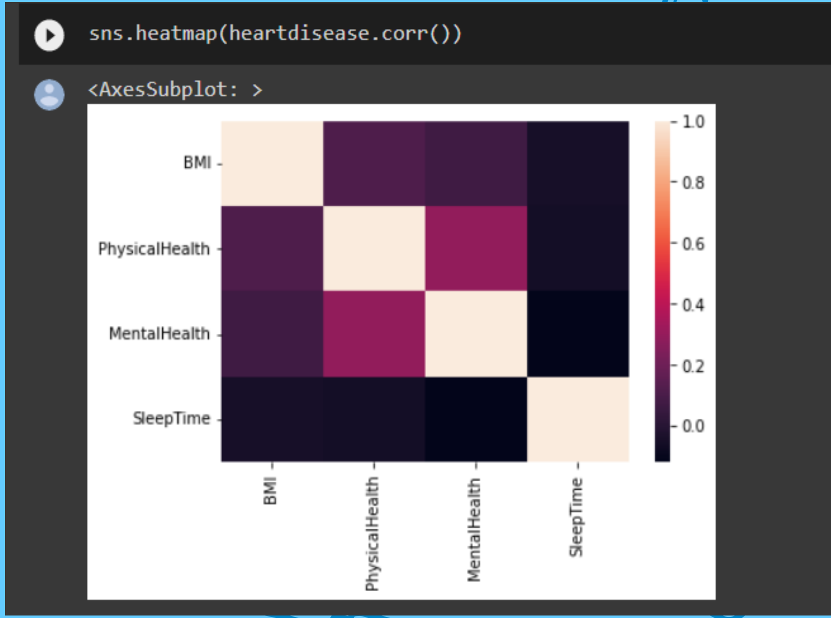
```
[ ] sns.heatmap(heartdisease.corr())
```

Feature Analysis

In feature analysis, we will study the relationship between variables. The idea is to find interesting relationships that show the influence of one variable on the other, preferably on the target variable (Heart disease).

The most significant variables which have a strong relationship with Heart Disease are:

- **Stroke – Yes**
- **Difficulty in Walking – Yes**
- **Diabetic – Yes**
- **Physical Health**
- **General Health – Poor**
- **Kidney Disease – Yes**

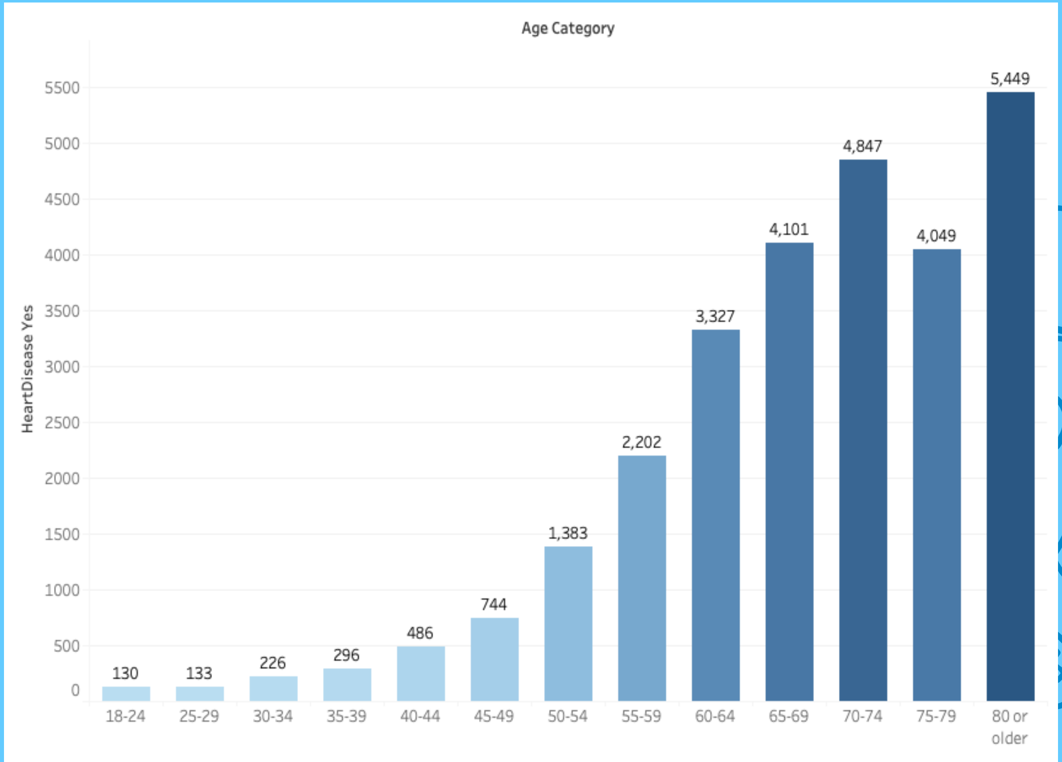


These variables will be used for Model Learning analysis and data visualization purposes.

Data Visualization

Age category v Heart Disease (Yes)

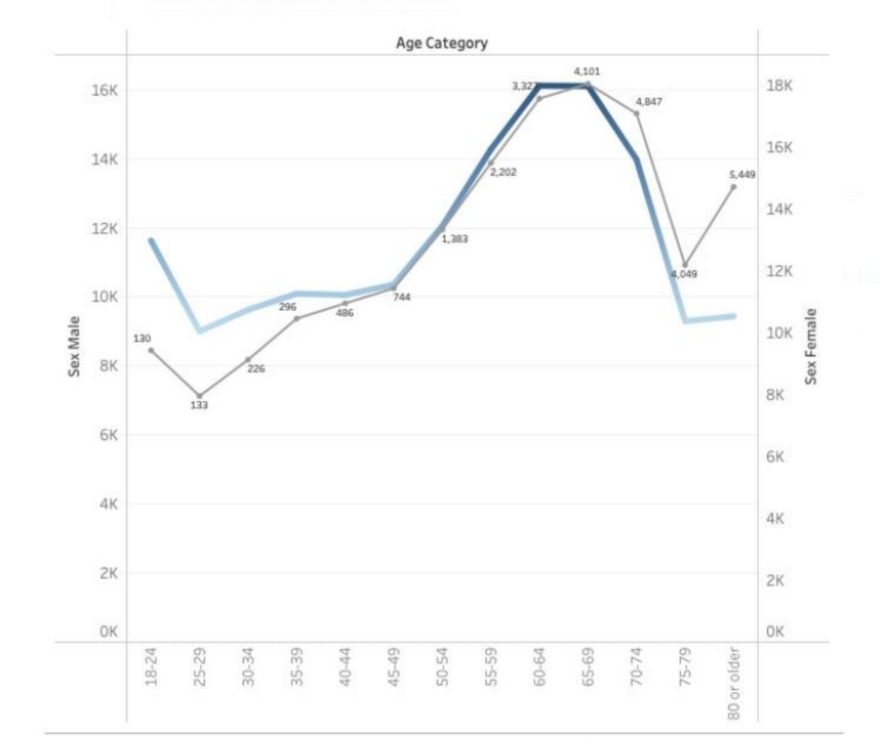
- Overall, there is an upward trend of heart disease as the age increases
- As age increases, heart disease in people increases accordingly
- However, there is a drop in heart disease at ages 75-79



Data Visualization

Age Category v Male v Female

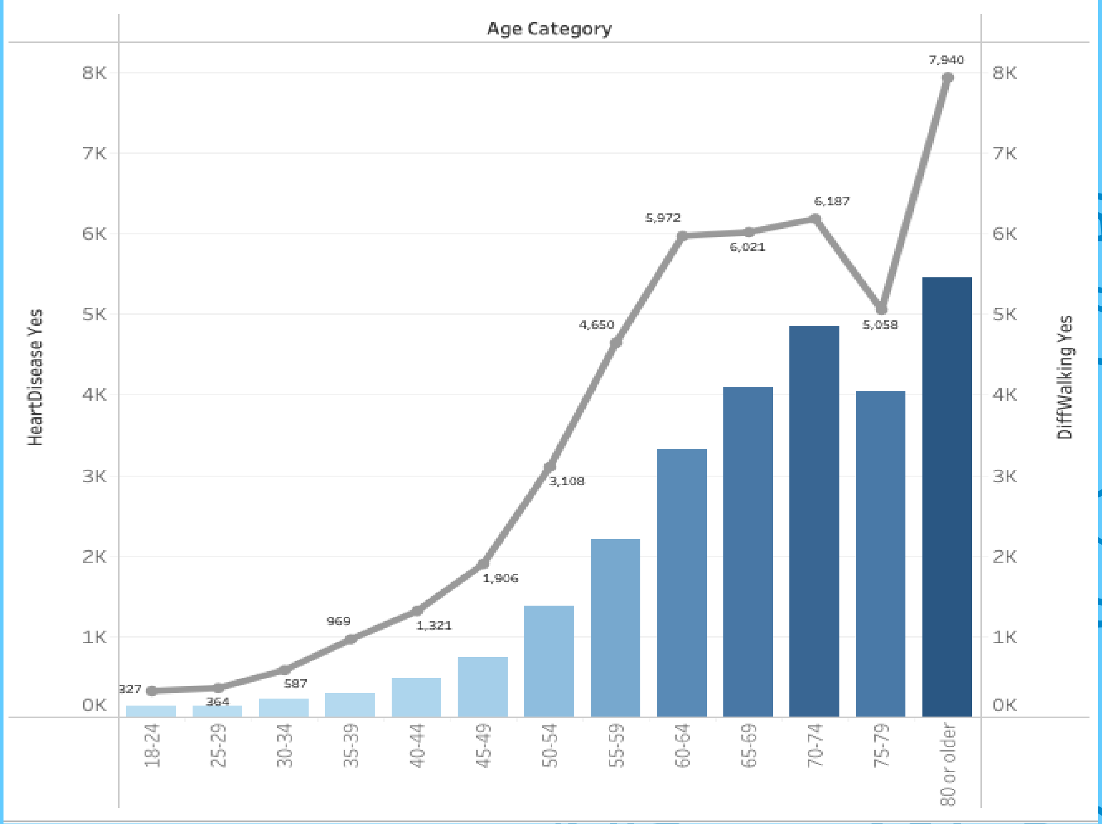
- As people get older, heart disease affects both females and males in the same way
- Overall, males have a higher risk of heart disease as compared to females
- However, there is a sudden drop in heart disease in both females and males at ages 25-29 and 75-79



Data Visualization

Age category v Heart Disease v Difficulty in walking

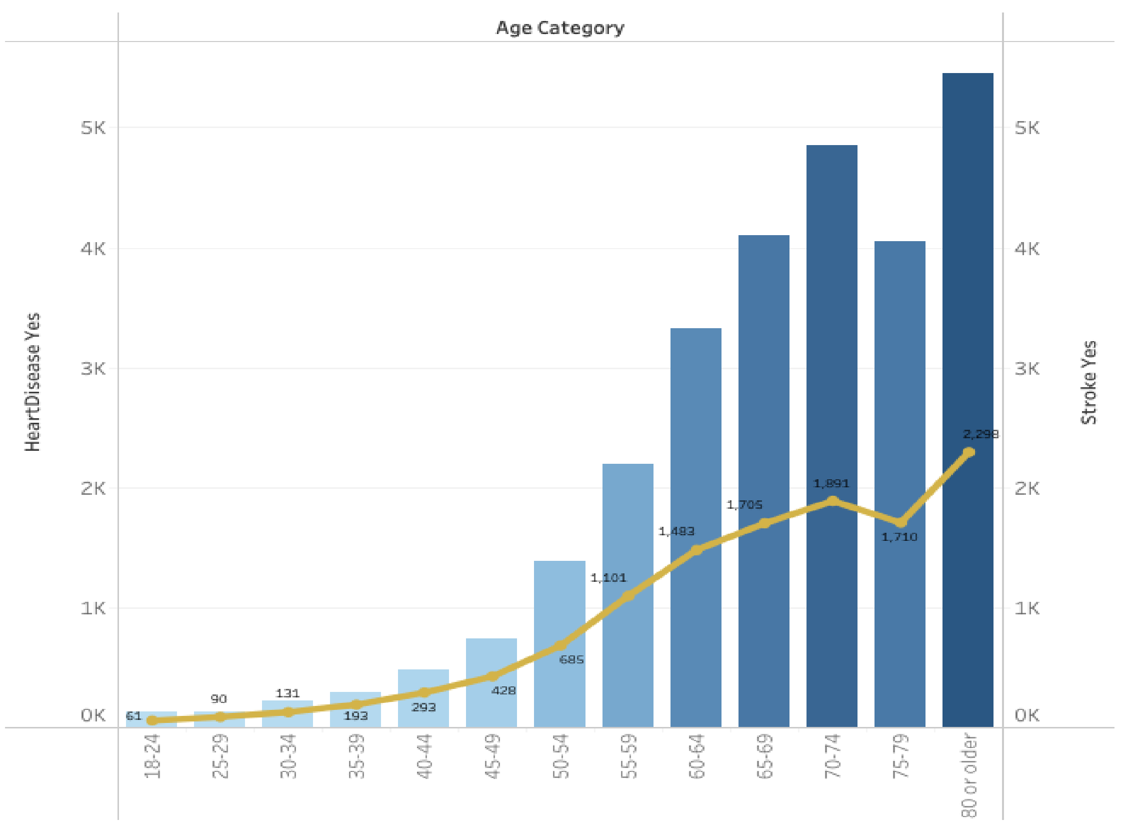
- Overall, there is an increasing trend among all the variables
- As the difficulty in walking increases age-wise, the chances of heart disease increases
- However, there is a drop in the number of people with both heart disease and difficulty in walking at ages 75-79.
- Hence, we can say that heart disease and difficulty in walking are going in sync



Data Visualization

Age category v Heart Disease v Stroke

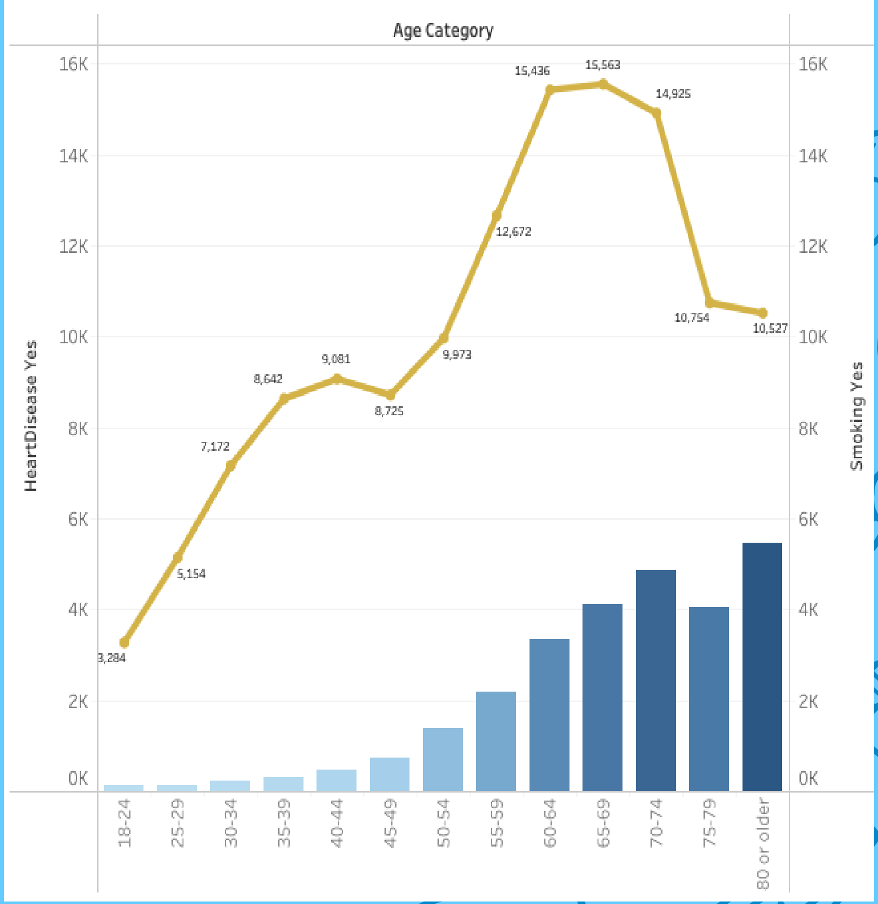
- Overall, there is an increasing trend among all the variables
- As the strokes increases age-wise, the chances of heart disease increases
- However, there is a drop in the number of people with both heart disease and strokes at ages 75-79.
- Hence, we can say that heart disease and strokes are going in sync



Data Visualization

Age Category v Heart Disease v Smoking

- Overall, there is an increasing trend among all the variables
- People who smoke are more in number than people who have heart disease. On average, ¼ people who smoke have heart disease
- Furthermore, there is a sudden drop in heart disease and smoking in people at ages 75-79



03

Modelling

- Logistic Regression
- KNN
- Decision Tree
- Random Forest

Sample Data

HeartDisease	BMI	Smoking	AlcoholDrinking	Stroke	PhysicalHealth	MentalHealth	DiffWalking	Sex	AgeCategory	Race
No	16.6	Yes	No	No	3.0	30.0	No	Female	55-59	White
No	20.34	No	No	Yes	0.0	0.0	No	Female	80 or older	White
No	26.58	Yes	No	No	20.0	30.0	No	Male	65-69	White
No	24.21	No	No	No	0.0	0.0	No	Female	75-79	White
No	23.71	No	No	No	28.0	0.0	Yes	Female	40-44	White

Diabetic	PhysicalActivity	GenHealth	SleepTime	Asthma	KidneyDisease	SkinCancer
Yes	Yes	Very good	5.0	Yes	No	Yes
No	Yes	Very good	7.0	No	No	No
Yes	Yes	Fair	8.0	Yes	No	No
No	No	Good	6.0	No	No	Yes
No	Yes	Very good	8.0	No	No	No

Data Insights

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 319795 entries, 0 to 319794
Data columns (total 18 columns):
 #   Column            Non-Null Count   Dtype  
---  ------            --------------   -----  
 0   HeartDisease      319795 non-null  object 
 1   BMI               319795 non-null  float64
 2   Smoking           319795 non-null  object 
 3   AlcoholDrinking   319795 non-null  object 
 4   Stroke            319795 non-null  object 
 5   PhysicalHealth    319795 non-null  float64
 6   MentalHealth      319795 non-null  float64
 7   DiffWalking       319795 non-null  object 
 8   Sex               319795 non-null  object 
 9   AgeCategory       319795 non-null  object 
 10  Race              319795 non-null  object 
 11  Diabetic          319795 non-null  object 
 12  PhysicalActivity  319795 non-null  object 
 13  GenHealth         319795 non-null  object 
 14  SleepTime         319795 non-null  float64
 15  Asthma            319795 non-null  object 
 16  KidneyDisease     319795 non-null  object 
 17  SkinCancer        319795 non-null  object 
```

```
data.shape
```

```
(319795, 18)
```

Checking null values in data

```
data.isnull().sum()
```

```
HeartDisease        0
BMI                 0
Smoking             0
AlcoholDrinking     0
Stroke              0
PhysicalHealth      0
MentalHealth        0
DiffWalking         0
Sex                 0
AgeCategory         0
Race                0
Diabetic            0
PhysicalActivity    0
GenHealth           0
SleepTime           0
Asthma              0
KidneyDisease       0
SkinCancer          0
```

Data Balancing

For balancing data, We have used SMOTE method to do oversampling the data

```
from imblearn.over_sampling import SMOTENC
smote = SMOTENC([1,2,3,6,7,8,9,10,11,12,14,15,16],random_state = 101)
X_oversample, y_oversample = smote.fit_resample(X, y)
```

```
print('X sample :',X_oversample.shape)
print('Y_sample :',y_oversample.shape)
```

```
X sample : (584844, 17)
Y_sample : (584844,)
```

```
y_oversample.value_counts(normalize=True)*100
```

```
No     50.0
Yes    50.0
Name: HeartDisease, dtype: float64
```

Building Pipeline

```
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import make_pipeline, Pipeline
from sklearn.impute import SimpleImputer

cat_columns = [i for i in X_oversample.columns if data[i].dtype == 'object']
num_columns = [i for i in data.columns if data[i].dtype != 'object' and i not in ['BMI']]
```

For numeric data

```
num_pipe = make_pipeline(SimpleImputer(missing_values=np.nan,strategy = 'mean'),
    StandardScaler(), SimpleImputer(missing_values=np.nan,strategy = 'mean')
)
```

For Categorical data

```
cat_pipe = make_pipeline(SimpleImputer(missing_values=np.nan,strategy = 'most_frequent'),
    OneHotEncoder(handle_unknown='ignore',drop='if_binary'),
    SimpleImputer(missing_values=np.nan,strategy = 'most_frequent')
)
```

Full pipeline

```
full_pipe = ColumnTransformer([('num', num_pipe, num_columns),
    ('cat', cat_pipe, cat_columns)
])
```

Splitting Data

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X_oversample,y_dummy,test_size = 0.4,random_state=2)
X_validation,X_test,y_validation,y_test = train_test_split(X_test,y_test,test_size = 0.5,random_state=2)
```

```
y_train.value_counts(True)*100
```

```
0    50.110571
1    49.889429
Name: HeartDisease, dtype: float64
```

We have split the data into 3 parts:

1 Training dataset,
2 Validation dataset
3 Test dataset

Logistic Regression

- Logistic regression is a **supervised classification algorithm**.

- Logistic Regression accomplishes binary classification tasks by predicting an outcome, event, or observation probability.

- It uses logistic functions to predict the probability of a binary outcome.

- This classification model delivers a binary outcome limited to two possible effects: yes/no, 0/1, or true/false.

Data Preparation

```
LR_data = X_oversample.copy()
LR_data['HeartDisease'] = y_dummy.copy()
```

Removing Outliers

```
def outliers(features,df):
  Q1 = df[[features]].quantile(q = 0.25)[0]
  Q3 = df[[features]].quantile(q = 0.75)[0]
  iqr = Q3 - Q1
  min_iqr = Q1 - 1.5*iqr
  max_iqr = Q3 + 1.5*iqr
  return min_iqr,max_iqr,df[[features]].min()[0],df[[features]].max()[0]

def convert_nan(x,min_iqr = min_iqr,max_iqr = max_iqr):
    if (x > max_iqr):
      x=np.nan
    else:
      x = x
    return x

for i in ['BMI','SleepTime']:
  c=0
  min_iqr,max_iqr,min_value,max_value = outliers(i,LR_data)
  if min_value < min_iqr:
    c+=1
    print('Column ->',i)
    print('--->low bound Outliers at',min_iqr)
  if max_value > max_iqr:
    if c==0:
      print('Column ->',i)
    print('--------->uppar bound Outliers at',max_iqr)
  print()
  print('Conver outliers to nan\n')

  LR_data[i] = LR_data[i].map(lambda x: np.nan if x<min_iqr or x >max_iqr else x )
```

Model Evaluation

The accuracy of the Logistic Regression model is 76.8%

```
LR = make_pipeline(full_pipe, LogisticRegression(max_iter=10000))
LR.fit(X_train,y_train)
y_lr_predict = LR.predict(X_validation)
print('LogisticRegression accuracy:',(accuracy_score(y_validation,y_lr_predict))*100)
```

```
LogisticRegression accuracy: 76.80240063606597
```

Confusion Matrix

	Predicted Yes	Predicted No
Actual Yes	43295	11716
Actual No	14855	47103

K- Nearest Neighbours

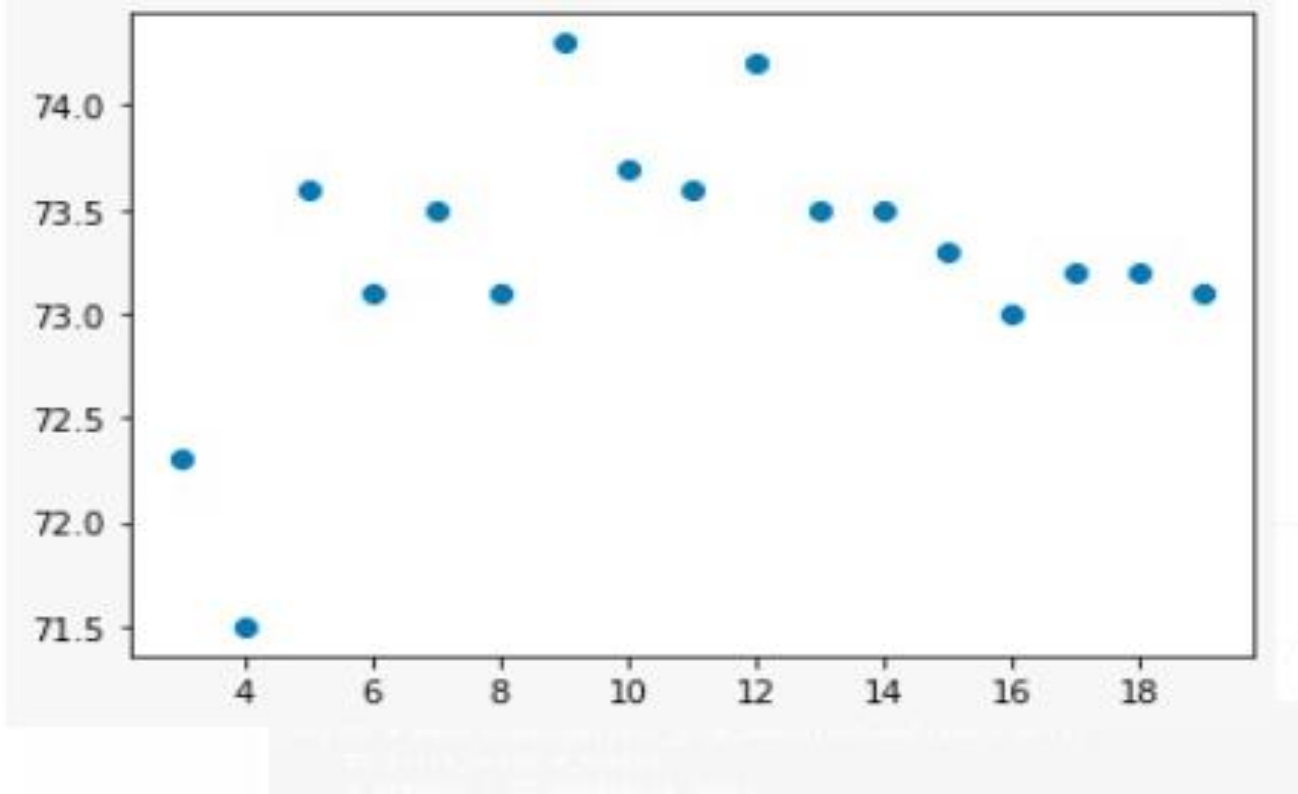
- KNN algorithm is a **supervised machine learning algorithm**.

- It's a classification algorithm that predicts a class of a target variable based on a defined number of nearest neighbors.

- Choosing the number of nearest neighbors, i.e., the value of k, significantly determines the model's efficacy.

Choosing hyper-parameter

We have tried multiple K values to check which K value is the best for the model.
We have selected K value as 9 based on this graph.



Model Evaluation

The accuracy of the KNN model is 74.4%

```
knn = make_pipeline(full_pipe,KNeighborsClassifier(n_neighbors=9))
knn.fit(X_train,y_train)
y_knn_predict = knn.predict(X_validation)
print('knn accuracy:',(accuracy_score(y_validation,y_knn_predict))*100)
```

```
knn accuracy: 74.3
```

```
predict = knn.predict(X_test)
print('knn accuracy:',(accuracy_score(y_test,predict))*100)
```

```
knn accuracy: 74.4
```

Confusion Matrix

```
cm(confusion_matrix(y_test,predict))
```

	Predicted Yes	Predicted No
Actual Yes	354	92
Actual No	164	390

Decision Tree

- A decision tree is one of the supervised machine learning algorithms. This algorithm can be used for regression and classification problems

- A decision tree follows a set of if-else conditions to visualize the data and classify it according to the requirements.

Model Evaluation

The accuracy of the Decision Tree is 75.6%

```
DT = make_pipeline(full_pipe,DecisionTreeClassifier())
DT.fit(X_train,y_train)
y_predict = DT.predict(X_test)

print('Decision Tree accuracy:',(accuracy_score(y_test,y_predict))*100)

cm(confusion_matrix(y_test,y_predict))
```

```
Decision Tree accuracy: 75.6
```

Confusion Matrix

	Predicted Yes	Predicted No
Actual Yes	406	132
Actual No	112	350

Random Forest

- Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems.

- The “forest” it builds is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result.

- Random forest adds additional randomness to the model while growing the trees. Whenever the tree splits, it only has access to a random sample of predictors.

Model Evaluation

The accuracy of the Random Forest model is 77.1%

```
RF = make_pipeline(full_pipe,RandomForestClassifier())
RF.fit(X_train,y_train)
y_predict = RF.predict(X_test)

print('Random Forest Tree accuracy:',(accuracy_score(y_test,y_predict))*1

cm(confusion_matrix(y_test,y_predict))
```

```
Random Forest Tree accuracy: 77.10000000000001
```

Confusion Matrix

	Predicted Yes	Predicted No
Actual Yes	381	92
Actual No	137	390

Null Value Test

```
import random
null_test_data = X_test.copy()
```

```
column_list = null_test_data.columns
for i in range(1000):
  r_index = random.randint(0,null_test_data.shape[0]-1)
  column = column_list[random.randint(0,len(column_list)-1)]
  null_test_data.loc[null_test_data.index == r_index,column] = np.nan
```

```
null_test_data.isna().sum()
```

```
BMI                  17
Smoking              20
AlcoholDrinking      13
Stroke               32
PhysicalHealth       31
MentalHealth         19
DiffWalking          19
Sex                  33
AgeCategory          25
Race                 22
Diabetic             21
PhysicalActivity     33
GenHealth            23
SleepTime            24
Asthma               34
KidneyDisease        30
SkinCancer           33
dtype: int64
```

```
RF.predict(null_test_data)
```

```
array([0, 1, 1, ..., 0, 0, 0])
```

04

Conclusions

- The most significant variable is Stroke – Yes, for the complete classification process.
- We can observe that Males have higher chances of heart disease than females.
- People older than 65 years have a very high chance of heart disease.
- The age group between 25 to 40 who have a smoking habit have a higher chance of having heart disease.
- Random Forrest is the best-performing model, with an accuracy of 77%.

MODE	ACCURACY
Logistic	76.8
KNN	74.4
Decision Tree	75.6
Random Forest	77

Thanks