FINAL CENTRAL LP

March 17, 2024

[2]:
```
#importing libraries
import numpy as np
import pandas as pd
```

[3]:
```
#importing data
df=pd.read_csv("C:\\Users\\hp\\Downloads\\Health_Care_dataset.csv")
df
```

[3]:

	Age	Heart_disease
0	3.745401	186.756403
1	9.507143	653.040955
2	7.319939	438.165461
3	5.986585	319.056918
4	1.560186	94.475345
..	…	…
95	4.937956	251.907334
96	5.227328	274.649221
97	4.275410	217.232574
98	0.254191	65.519389
99	1.078914	86.139910

[100 rows x 2 columns]

[4]:
```
#calling arrays
```

[5]:
```
x= np.array(df["Age"]).reshape(-1,1)
```

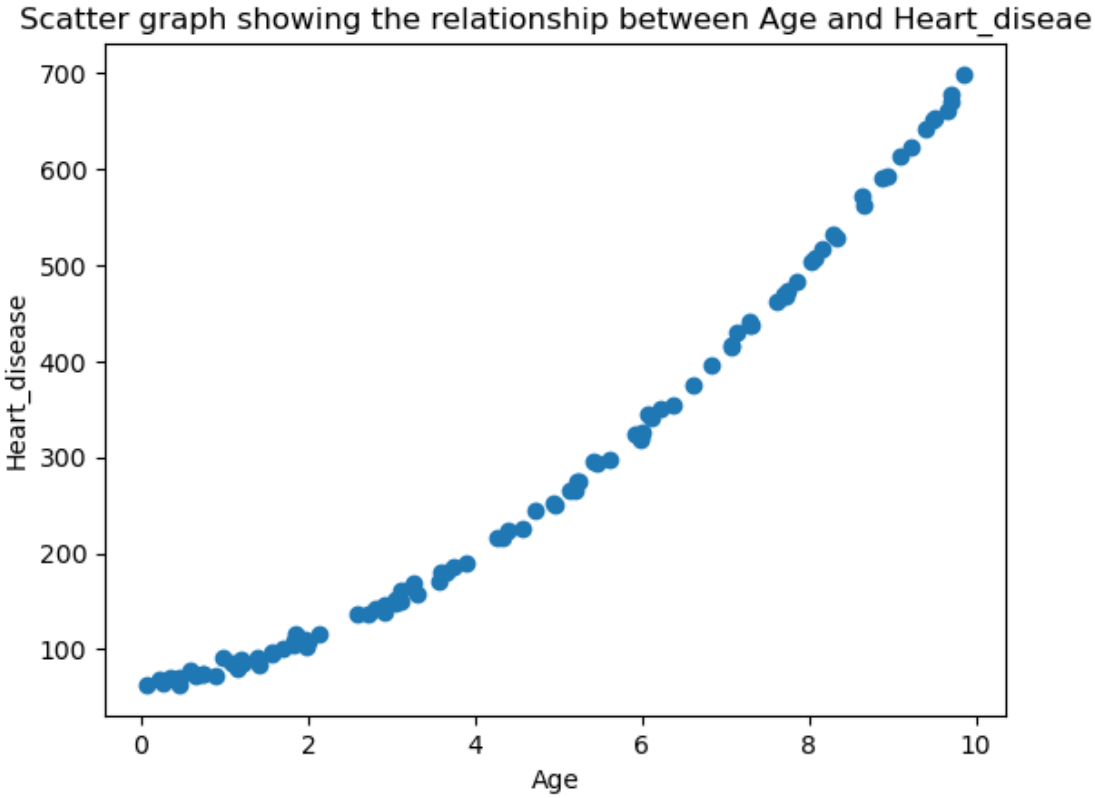
[6]:
```
y=np.array(df["Heart_disease"])
```

[7]:
```
#checking for missing data
df.isna().sum()
```

[7]:
```
Age              0
Heart_disease    0
dtype: int64
```

[8]:
```
#visualisation of the graph
```

[9]:
```
import matplotlib.pyplot as plt
plt.scatter(x,y)
plt.xlabel("Age")
plt.ylabel("Heart_disease")
plt.title("Scatter graph showing the relationship between Age and Heart_diseae")
#plt.grid(True)
plt.show()
```



[10]:
```
#spliting data
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test =train_test_split(x,y,test_size = 0.2)
```

[11]:
```
#standardizing data
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
x_train_scaled = scaler.fit_transform(x_train)
x_test_scaled = scaler.transform(x_test)
```

[13]:
```
#building model
from sklearn.linear_model import LinearRegression
model=LinearRegression()
```

```
model
```

[13]:
```
LinearRegression()
```

[14]:
```
#Model fitting
model.fit(x_train_scaled,y_train)
```

[14]:
```
LinearRegression()
```

[15]:
```
#Making prediction
y_pred=model.predict(x_test_scaled)
y_pred
```

[15]:
```
array([342.93280613, 195.83881853, 434.44797703, 533.01289795,
       406.05101465, 367.76849487,  14.45988554, 467.30998378,
       215.42854157, 187.34830324, 226.51718292, 186.89398322,
       567.73619884, 331.34896842, 597.47981024, 374.46270551,
       608.26641502, 117.50201731, 107.35942813, 199.30983018])
```

[16]:
```
#getting coefficient
model.coef_
```

[16]:
```
array([186.74997827])
```

[17]:
```
#Getting intercept
model.intercept_
```

[17]:
```
276.658381441299
```

[18]:
```
#Model accuracy on train values
model.score(x_train_scaled,y_train)
```

[18]:
```
0.9623361610251643
```

[19]:
```
#Model accuracy on test values
model.score(x_test_scaled,y_test)
```

[19]:
```
0.9511348054806654
```

[20]:
```
from sklearn.metrics import mean_absolute_error,r2_score,mean_squared_error
mae=mean_absolute_error(y_test,y_pred)
r2 = r2_score(y_test,y_pred)
mse = mean_squared_error(y_test,y_pred)

print(f"mae:{mae}")
print(f"r2:{r2}")
print(f"mse:{mse}")
```

```
mae:37.164980048735444
r2:0.9511348054806654
mse:1818.918837925957
```

MODEL OPTIMIZATION

[21]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LinearRegression
```

[37]:
```
model=LinearRegression()
model
```

[37]:
```
LinearRegression()
```

[73]:
```
#Define the parameter grid search
param_grid = {
'fit_intercept':[True,False],
'copy_X':[True,False],
'n_jobs':[True,False],

            }
param_grid
```

[73]:
```
{'fit_intercept': [True, False],
 'copy_X': [True, False],
 'n_jobs': [True, False]}
```

[74]:
```
# Perform GridSearchCV
grid_search = GridSearchCV(model, param_grid, cv=5)
grid_search
```

[74]:
```
GridSearchCV(cv=5, estimator=LinearRegression(),
             param_grid={'copy_X': [True, False],
                         'fit_intercept': [True, False],
                         'n_jobs': [True, False]})
```

[75]:
```
grid_search.fit(x_train_scaled, y_train)
grid_search
```

[75]:
```
GridSearchCV(cv=5, estimator=LinearRegression(),
             param_grid={'copy_X': [True, False],
                         'fit_intercept': [True, False],
                         'n_jobs': [True, False]})
```

[78]:
```
#Get the best parameters found from Grid search
best_params = grid_search.best_params_
best_params
```

[78]:
```
{'copy_X': True, 'fit_intercept': True, 'n_jobs': True}
```

[82]:
```
# Train LinearRegression model with the best parameters
best_model = LinearRegression(**best_params)
best_model.fit(x_train_scaled, y_train)
```

[82]:
```
LinearRegression(n_jobs=True)
```

[85]:
```
# Make predictions
y_pred_ridge = best_model.predict(x_test_scaled)
y_pred
```

[85]:
```
array([342.93280613, 195.83881853, 434.44797703, 533.01289795,
       406.05101465, 367.76849487,  14.45988554, 467.30998378,
       215.42854157, 187.34830324, 226.51718292, 186.89398322,
       567.73619884, 331.34896842, 597.47981024, 374.46270551,
       608.26641502, 117.50201731, 107.35942813, 199.30983018])
```

[90]:
```
# Model evaluation
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
print(f"mae:{mae}")
print(f"r2:{r2}")
print(f"mse:{mse}")
```

```
mae:37.164980048735444
r2:0.9511348054806654
mse:1818.918837925957
```