AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Faculty of Science and Information Technology

Assignment Cover Sheet

Assign./Case Title:	A Machine Learning Model Based on Linear Regression		
Assign./CaseNo:	1	Date of Submission:	4 December 2021
Course Title:	PROGRAMMING IN PYTHON		
Course Code:	01461	Section:	A
Semester:	Fall 2021-22	Degree Program:	BSc [CSE]
Course Teacher:	**AKINUL ISLAM JONY sir**		

Declaration and Statement of Authorship:

1. I/we hold a copy of this Assignment/Case-Study, which can be produced if the original is lost/damaged.
2. This Assignment/Case-Study is my/our original work and no part of it has been copied from any other student's work or from any other source except where due acknowledgement is made.
3. No part of this Assignment/Case-Study has been written for me/us by any other person except where such collaborationhas been authorized by the concerned teacher and is clearly acknowledged in the assignment.
4. I/we have not previously submitted or currently submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared and archived for the purpose of detecting plagiarism.
6. I/we give permission for a copy of my/our marked work to be retained by the Faculty for review and comparison, including review by external examiners.
7. I/we understand thatPlagiarism is the presentation of the work, idea or creation of another person as though it is your own. It is a formofcheatingandisaveryseriousacademicoffencethatmayleadtoexpulsionfromtheUniversity. Plagiarized material can be drawn from, and presented in, written, graphic and visual form, including electronic data, and oral presentations. Plagiarism occurs when the origin of them arterial used is not appropriately cited.
8. I/we also understand that enabling plagiarism is the act of assisting or allowing another person to plagiarize or to copy my/our work.

* *Student(s) must complete all details except the faculty use part.*
** Please submit all assignments to your course teacher or the office of the concerned teacher.

Group Name/No.:

No	Name	ID	ROLL	Signature
1	Zubair Ahmed	19-39745-1		
2				
3				
4				
5				
6				
7				
8				
9				
10				

Faculty use only

FACULTYCOMMENTS		
	Marks Obtained	
	Total Marks	

In [1]:

```python
import pandas as pd
import numpy as np
import sklearn as sk
from sklearn import datasets
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [2]:

```python
heart = pd.read_csv("C:\\Users\\USER\\Downloads\\heart.csv")
```

In [3]:

```python
heart.keys() # Showing the keys of the 'heart' as it's a dictionary object
```

Out[3]:

```
Index(['Age', 'RestingBP', 'Cholesterol', 'FastingBS', 'MaxHR', 'Oldpeak',
       'HeartDisease'],
      dtype='object')
```

In [4]:

```python
heart.describe() #The describe() shows the summary of the basic statistics of the dataset
```

Out[4]:

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
count	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000	918.000000
mean	53.510893	132.396514	198.799564	0.233115	136.809368	0.887364	0.553377
std	9.432617	18.514154	109.384145	0.423046	25.460334	1.066570	0.497414
min	28.000000	0.000000	0.000000	0.000000	60.000000	-2.600000	0.000000
25%	47.000000	120.000000	173.250000	0.000000	120.000000	0.000000	0.000000
50%	54.000000	130.000000	223.000000	0.000000	138.000000	0.600000	1.000000
75%	60.000000	140.000000	267.000000	0.000000	156.000000	1.500000	1.000000
max	77.000000	200.000000	603.000000	1.000000	202.000000	6.200000	1.000000

In [5]:

```python
heart #Printing the dataset as a DataFrame
```

Out[5]:

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease
0	40	140	289	0	172	0.0	0
1	49	160	180	0	156	1.0	1
2	37	130	283	0	98	0.0	0
3	48	138	214	0	108	1.5	1
4	54	150	195	0	122	0.0	0
...	...	...	...	...	...	...	...
913	45	110	264	0	132	1.2	1
914	68	144	193	1	141	3.4	1
915	57	130	131	0	115	1.2	1
916	57	130	236	0	174	0.0	1
917	38	138	175	0	173	0.0	0

918 rows × 7 columns

In [6]:

```python
hmed = heart.median(axis=1) # The median values are taken as target variables in this dataset

hmed # This is the dependent variable which means 'hmed' is the target variable
```

Out[6]:

```
0      40.0
1      49.0
2      37.0
3      48.0
4      54.0
       ...
913    45.0
914    68.0
915    57.0
916    57.0
917    38.0
Length: 918, dtype: float64
```

In [7]:

```python
heart['Hmed'] = hmed # Including the Target label 'Hmed' into the 'heart' DataFrame
heart
```

Out[7]:

	Age	RestingBP	Cholesterol	FastingBS	MaxHR	Oldpeak	HeartDisease	Hmed
0	40	140	289	0	172	0.0	0	40.0
1	49	160	180	0	156	1.0	1	49.0
2	37	130	283	0	98	0.0	0	37.0
3	48	138	214	0	108	1.5	1	48.0
4	54	150	195	0	122	0.0	0	54.0
...	...	...	...	...	...	...	...	...
913	45	110	264	0	132	1.2	1	45.0
914	68	144	193	1	141	3.4	1	68.0
915	57	130	131	0	115	1.2	1	57.0
916	57	130	236	0	174	0.0	1	57.0
917	38	138	175	0	173	0.0	0	38.0

918 rows × 8 columns

In [8]:

```python
heart.isnull().sum() # isnull() is used to return the number of missing values for each column
#and sum() is used here to add the counted missing values
```

Out[8]:

```
Age             0
RestingBP       0
Cholesterol     0
FastingBS       0
MaxHR           0
Oldpeak         0
HeartDisease    0
Hmed            0
dtype: int64
```

In [9]:

```python
#heart.replace(0,np.nan, inplace=True)
#heart
```

In [10]:

```python
heart.isnull().sum()/len(heart) #the percentage of missing data of each column's
```

Out[10]:

```
Age             0.0
RestingBP       0.0
Cholesterol     0.0
FastingBS       0.0
MaxHR           0.0
Oldpeak         0.0
HeartDisease    0.0
Hmed            0.0
dtype: float64
```

In [11]:

```python
# Dropping the FastingBS,HeartDisease,Oldpeak and Cholesterol columns because they contain too much missing values
heart = heart.drop(['FastingBS'], axis=1)
heart = heart.drop(['HeartDisease'], axis=1)
heart = heart.drop(['Oldpeak'], axis=1)
heart = heart.drop(['Cholesterol'], axis=1)
heart
```

Out[11]:

	Age	RestingBP	MaxHR	Hmed
0	40	140	172	40.0
1	49	160	156	49.0
2	37	130	98	37.0
3	48	138	108	48.0
4	54	150	122	54.0
...	...	...	...	...
913	45	110	132	45.0
914	68	144	141	68.0
915	57	130	115	57.0
916	57	130	174	57.0
917	38	138	173	38.0

918 rows × 4 columns

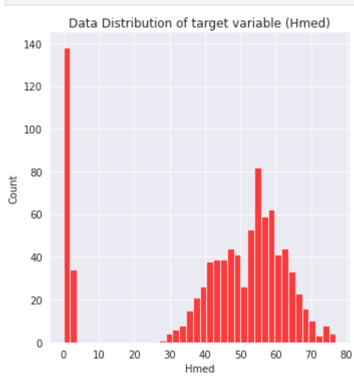
In [12]:

```python
print(heart.describe()) # describe() shows the summary of the basic statistics of the dataset
```

```
              Age   RestingBP       MaxHR        Hmed
count  918.000000  918.000000  918.000000  918.000000
mean    53.510893  132.396514  136.809368   43.213617
std      9.432617   18.514154   25.460334   21.894453
min     28.000000    0.000000   60.000000    0.000000
25%     47.000000  120.000000  120.000000   39.000000
50%     54.000000  130.000000  138.000000   51.000000
75%     60.000000  140.000000  156.000000   58.000000
max     77.000000  200.000000  202.000000   77.000000
```

In [13]:

```python
# The displot function is used to make a visualisation of the data distributions of the target variable.
# The displot function is used from the seaborn
with sns.axes_style('darkgrid'):
 sns.displot(heart['Hmed'], bins=40, color='red')
 plt.title("Data Distribution of target variable (Hmed)");
```



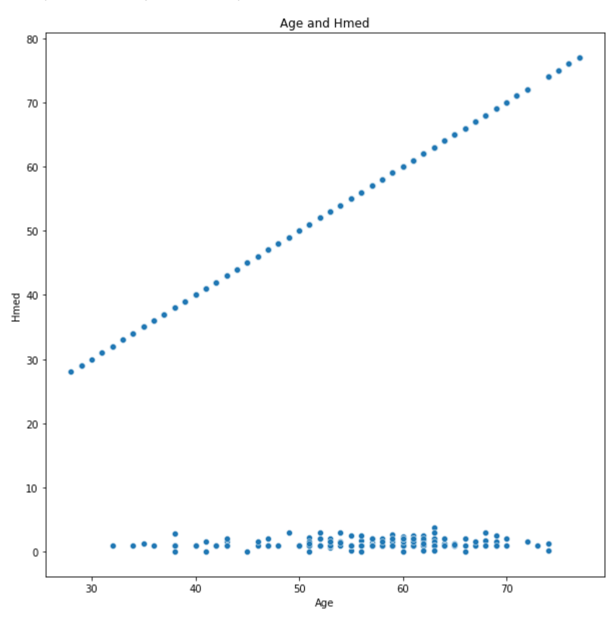
In [14]:

```python
# The function, corr() is used to calculate the correlations between the variables
corr_matrix = heart.corr().round(2)
# Updating the size of the figure
plt.figure(figsize = (15, 10))
# Here the "annot = True" is used to print the values which are inside the square
sns.heatmap(data=corr_matrix, annot=True);
```

In [15]:

```python
plt.figure(figsize=(10,10))
# It shows the Relatiosnships with 'Age' and 'Hmed'
sns.scatterplot(x=heart['Age'], y=heart['Hmed'])
plt.title("Age and Hmed")
```

Out[15]: Text(0.5, 1.0, 'Age and Hmed')

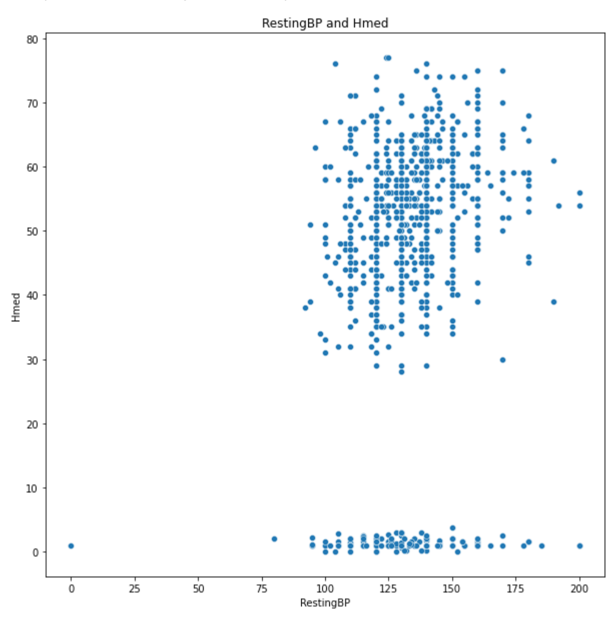


In [16]:

```python
plt.figure(figsize=(10,10))
# It shows the Relatiosnships with 'RestingBP' and 'Hmed'

sns.scatterplot(x=heart['RestingBP'], y=heart['Hmed'])
plt.title("RestingBP and Hmed")
```

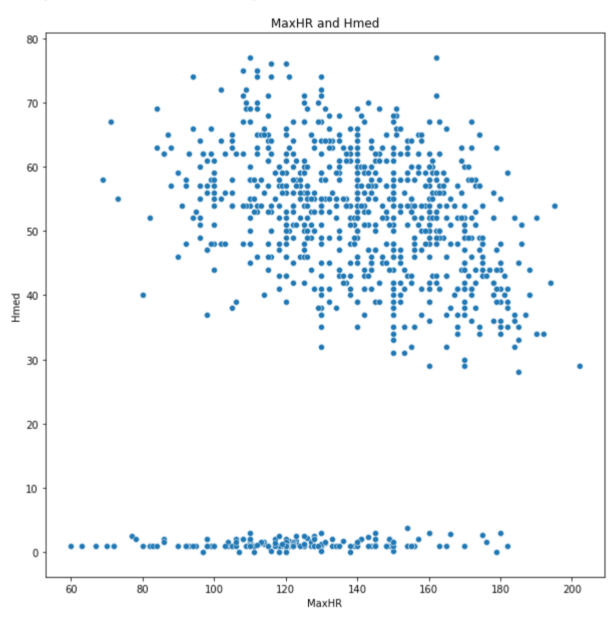
Out[16]: Text(0.5, 1.0, 'RestingBP and Hmed')



In [17]:

```python
plt.figure(figsize=(10,10))
# It shows the Relatiosnships with 'MaxHR' and 'Hmed'
sns.scatterplot(x=heart['MaxHR'], y=heart['Hmed'])
plt.title("MaxHR and Hmed")
```

Out[17]: Text(0.5, 1.0, 'MaxHR and Hmed')



In [18]:

```python
#Age and MaxHR looks like that they have linear relationships with Hmed
# This is the Feature matrix
X = heart[['Age', 'MaxHR']]
X
```

Out[18]:

	Age	MaxHR
0	40	172
1	49	156
2	37	98
3	48	108
4	54	122
...	...	...
913	45	132
914	68	141
915	57	115
916	57	174
917	38	173

918 rows × 2 columns

In [19]:

```python
# This is the Target variable
y = heart[['Hmed']]
y
```

Out[19]:

	Hmed
0	40.0
1	49.0
2	37.0
3	48.0
4	54.0
...	...
913	45.0
914	68.0
915	57.0
916	57.0
917	38.0

918 rows × 1 columns

In [20]:

```python
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3, random_state = 1)
# The test_size is 0.3 which means 918 * 0.3 = 276 rows
# The random_state is 1 which ensures that the split will always remain the same
print("X_train shape: ", X_train.shape)
print("X_test shape: ", X_test.shape)
print("y_train shape: ", y_train.shape)
print("y_test shape: ", y_test.shape)
```

```
X_train shape:  (642, 2)
X_test shape:  (276, 2)
y_train shape:  (642, 1)
y_test shape:  (276, 1)
```

In [21]:

```python
from sklearn.linear_model import LinearRegression
# Instance of Linear Regression
lrm = LinearRegression()
# Fitting the data on the model
lrm.fit(X_train, y_train)
# Prediction
y_predicted = lrm.predict(X_test)
```

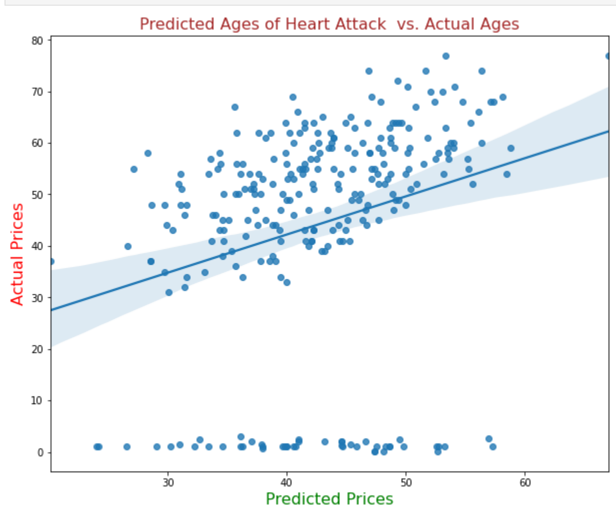
In [22]:

```python
from sklearn.metrics import mean_squared_error
# r-squared values
r2 = lrm.score(X_test, y_test)
# The difference between Predicted values and the Test values.
rmse = (np.sqrt(mean_squared_error(y_test, y_predicted)))
print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
print('r-squared: {}'.format(r2))
print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
print('root mean squared error: {}'.format(rmse))
print('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~')
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
r-squared: 0.055579315383474714
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
root mean squared error: 20.97672871945416
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

In [23]:

```python
# Plotting predictions vs actual
plt.figure(figsize=(10,8))
sns.regplot(x=y_predicted, y=y_test)
plt.xlabel('Predicted Prices', fontsize=16, color='green')
plt.ylabel('Actual Prices', fontsize=16, color='red')
plt.title("Predicted Ages of Heart Attack  vs. Actual Ages",fontsize=16, color='brown');
```



In [ ]: