Performing all model to the given dataset to choose the best fit. ¶

Linear Regression

Data Collection

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:
```
df=pd.read_csv(r"C:\Users\Sushma sree\Downloads\insurance.csv")
df
```

Out[2]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

Data Cleaning

In [3]:
```
1 df.head()
```

Out[3]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

In [4]:
```
1 df.tail()
```

Out[4]:

	age	sex	bmi	children	smoker	region	charges
1333	50	male	30.97	3	no	northwest	10600.5483
1334	18	female	31.92	0	no	northeast	2205.9808
1335	18	female	36.85	0	no	southeast	1629.8335
1336	21	female	25.80	0	no	southwest	2007.9450
1337	61	female	29.07	0	yes	northwest	29141.3603

In [5]:
```
1 df.describe
```

Out[5]:

```
<bound method NDFrame.describe of       age     sex     bmi  children smoker
region      charges
0      19  female  27.900         0    yes  southwest  16884.92400
1      18    male  33.770         1     no  southeast   1725.55230
2      28    male  33.000         3     no  southeast   4449.46200
3      33    male  22.705         0     no  northwest  21984.47061
4      32    male  28.880         0     no  northwest   3866.85520
...   ...     ...     ...       ...    ...        ...          ...
1333   50    male  30.970         3     no  northwest  10600.54830
1334   18  female  31.920         0     no  northeast   2205.98080
1335   18  female  36.850         0     no  southeast   1629.83350
1336   21  female  25.800         0     no  southwest   2007.94500
1337   61  female  29.070         0    yes  northwest  29141.36030

[1338 rows x 7 columns]>
```

In [6]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object 
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object 
 5   region    1338 non-null   object 
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

In [7]:
```
1 df.shape
```

Out[7]:
```
(1338, 7)
```

Data Preprocessing

In [8]:
```
1 df.isna().any()
```

Out[8]:
```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [9]:
```
1 df.isnull().sum()
```

Out[9]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

Hence,the given dataset has no null values and missing.

In [10]:
```
1 df['sex'].value_counts()
```

Out[10]:
```
sex
male      676
female    662
Name: count, dtype: int64
```

In [11]:
```
1 df['age'].value_counts()
```

Out[11]:
```
age
18    69
19    68
50    29
51    29
47    29
46    29
45    29
20    29
48    29
52    29
22    28
49    28
54    28
53    28
21    28
26    28
24    28
25    28
28    28
27    28
23    28
43    27
29    27
30    27
41    27
42    27
44    27
31    27
40    27
32    26
33    26
56    26
34    26
55    26
57    26
37    25
59    25
58    25
36    25
38    25
35    25
39    25
61    23
60    23
63    23
62    23
64    22
Name: count, dtype: int64
```

In [12]:
```
1 df['smoker'].value_counts()
```

Out[12]:
```
smoker
no     1064
yes     274
Name: count, dtype: int64
```

In [13]:
```
1 convert={"sex":{"female":1,"male":0}}
2 df=df.replace(convert)
3 df
```

Out[13]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	yes	southwest	16884.92400
1	18	0	33.770	1	no	southeast	1725.55230
2	28	0	33.000	3	no	southeast	4449.46200
3	33	0	22.705	0	no	northwest	21984.47061
4	32	0	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	0	30.970	3	no	northwest	10600.54830
1334	18	1	31.920	0	no	northeast	2205.98080
1335	18	1	36.850	0	no	southeast	1629.83350
1336	21	1	25.800	0	no	southwest	2007.94500
1337	61	1	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [14]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

Out[14]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	southwest	16884.92400
1	18	0	33.770	1	0	southeast	1725.55230
2	28	0	33.000	3	0	southeast	4449.46200
3	33	0	22.705	0	0	northwest	21984.47061
4	32	0	28.880	0	0	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	0	30.970	3	0	northwest	10600.54830
1334	18	1	31.920	0	0	northeast	2205.98080
1335	18	1	36.850	0	0	southeast	1629.83350
1336	21	1	25.800	0	0	southwest	2007.94500
1337	61	1	29.070	0	1	northwest	29141.36030

1338 rows × 7 columns

In [15]:
```
convert={"region":{"southwest":1,"southeast":2,"northeast":3,"northwest":4
df=df.replace(convert)
df
```

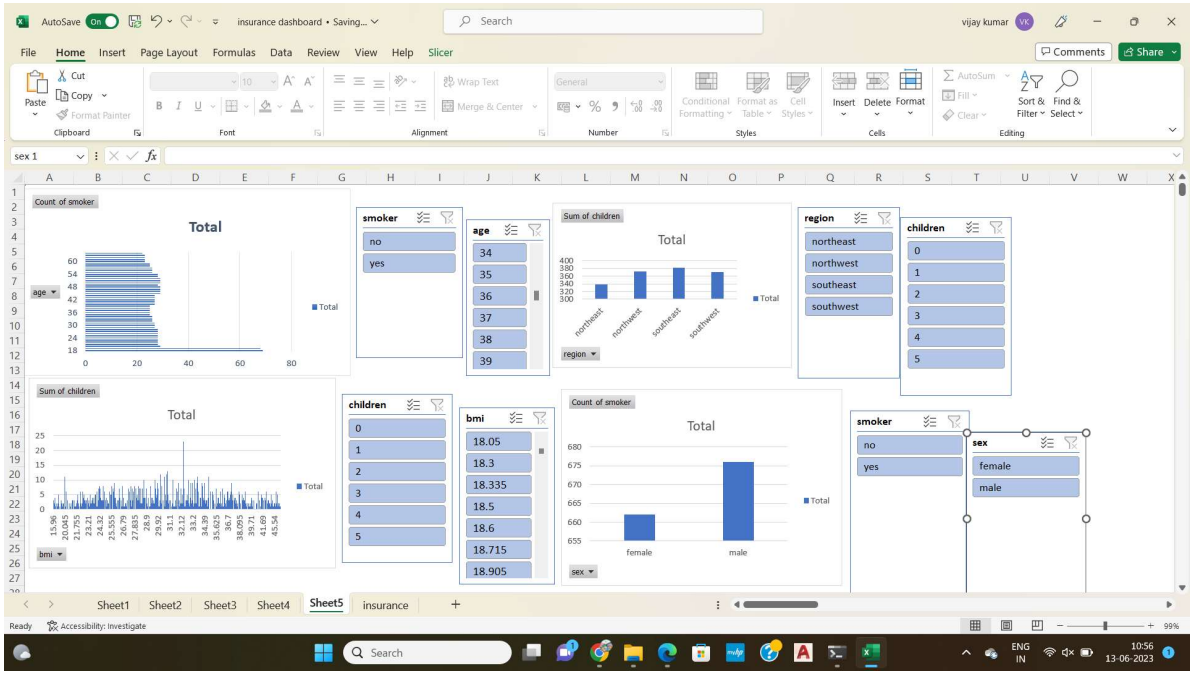
Out[15]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	1	16884.92400
1	18	0	33.770	1	0	2	1725.55230
2	28	0	33.000	3	0	2	4449.46200
3	33	0	22.705	0	0	4	21984.47061
4	32	0	28.880	0	0	4	3866.85520
...	...	...	...	...	...	...	...
1333	50	0	30.970	3	0	4	10600.54830
1334	18	1	31.920	0	0	3	2205.98080
1335	18	1	36.850	0	0	2	1629.83350
1336	21	1	25.800	0	0	1	2007.94500
1337	61	1	29.070	0	1	4	29141.36030

1338 rows × 7 columns

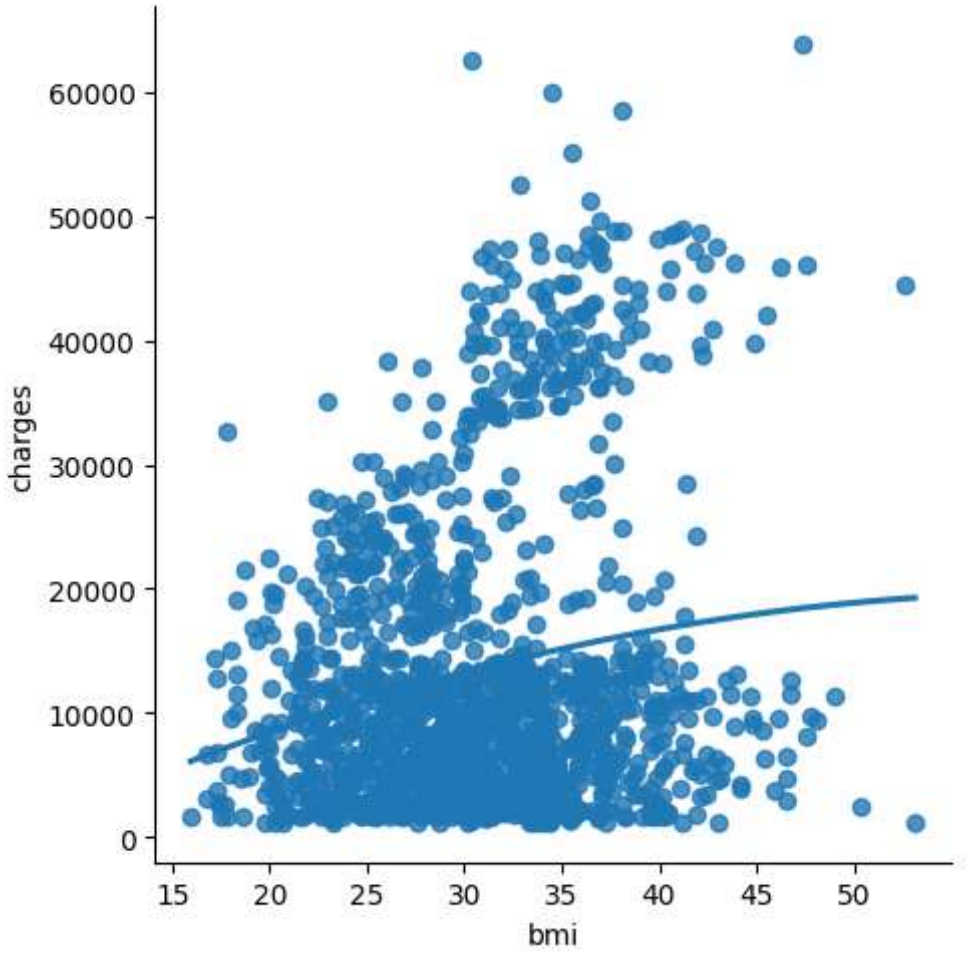
Data Visualisation

Dataset Dashboard in Excel sheet



In [16]:

```
sns.lmplot(x='bmi',y='charges',order=2,data=df,ci=None)
plt.show()
```



In [17]:
```
1 x=df[['bmi']]
2 y=df['charges']
```

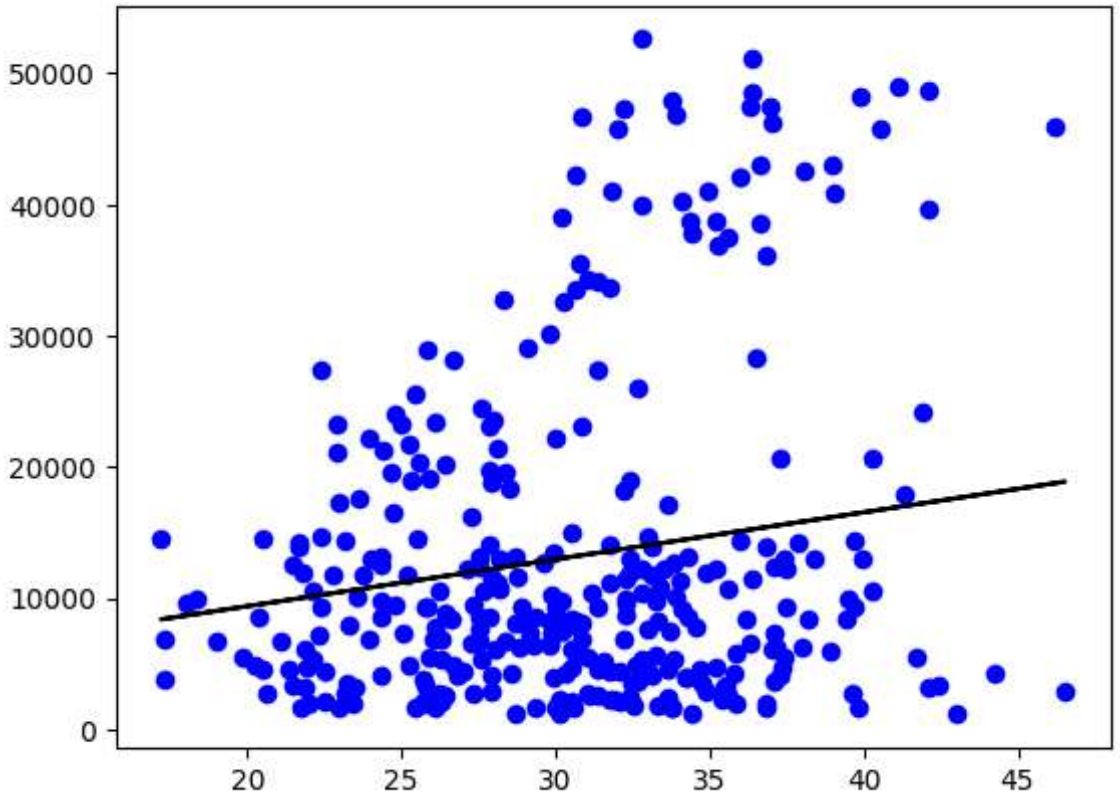
In [18]:
```
1 x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_
2 lr=LinearRegression()
```

In [19]:
```
1 lr.fit(x_train,y_train)
2 print(lr.score(x_test,y_test))
```

```
0.050136213258239914
```

In [20]:

```
1 y_pred=lr.predict(x_test)
2 plt.scatter(x_test,y_test,color='b')
3 plt.plot(x_test,y_pred,color='k')
4 plt.show()
```



Working with the sub data of the given dataset

In [21]:

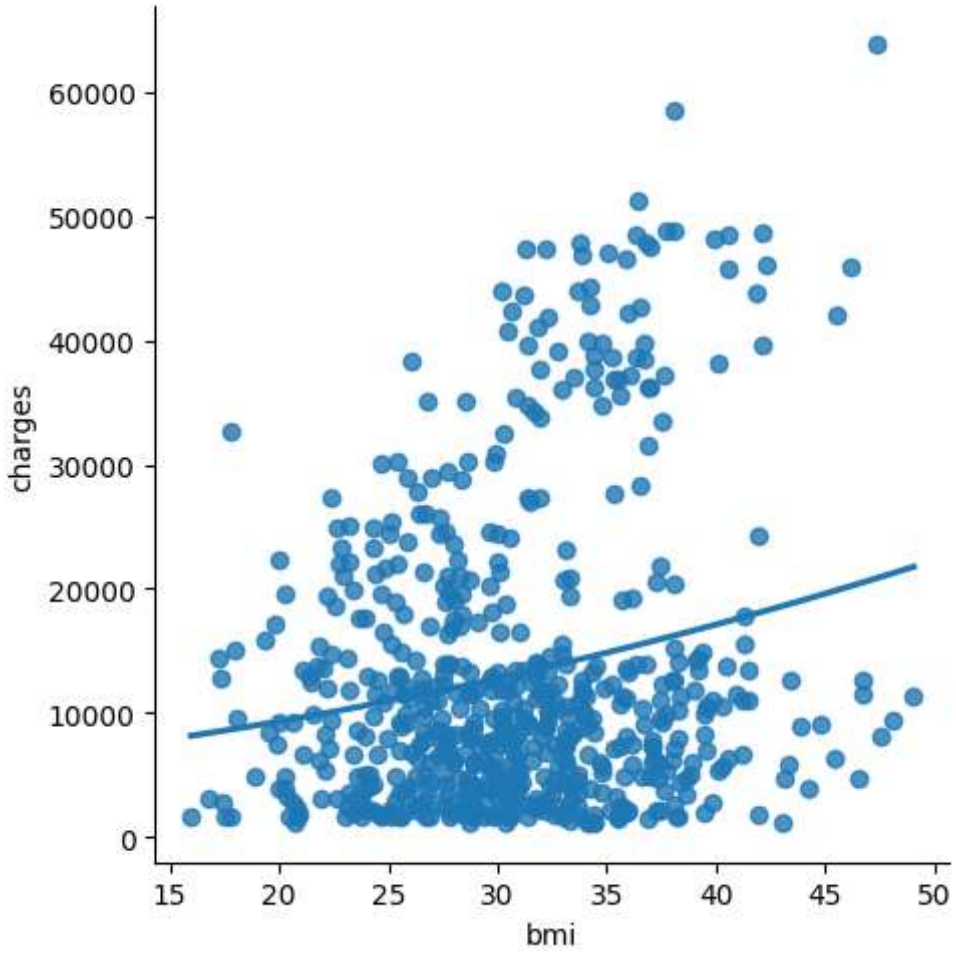
```
df600=df[:600]
df600
```

Out[21]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	1	16884.92400
1	18	0	33.770	1	0	2	1725.55230
2	28	0	33.000	3	0	2	4449.46200
3	33	0	22.705	0	0	4	21984.47061
4	32	0	28.880	0	0	4	3866.85520
...	...	...	...	...	...	...	...
595	46	1	33.725	1	0	3	8823.98575
596	42	1	29.480	2	0	2	7640.30920
597	34	1	33.250	1	0	3	5594.84550
598	43	0	32.600	2	0	1	7441.50100
599	52	1	37.525	2	0	4	33471.97189

600 rows × 7 columns

In [22]:
```
sns.lmplot(x='bmi',y='charges',order=2,ci=None,data=df600)
plt.show()
```

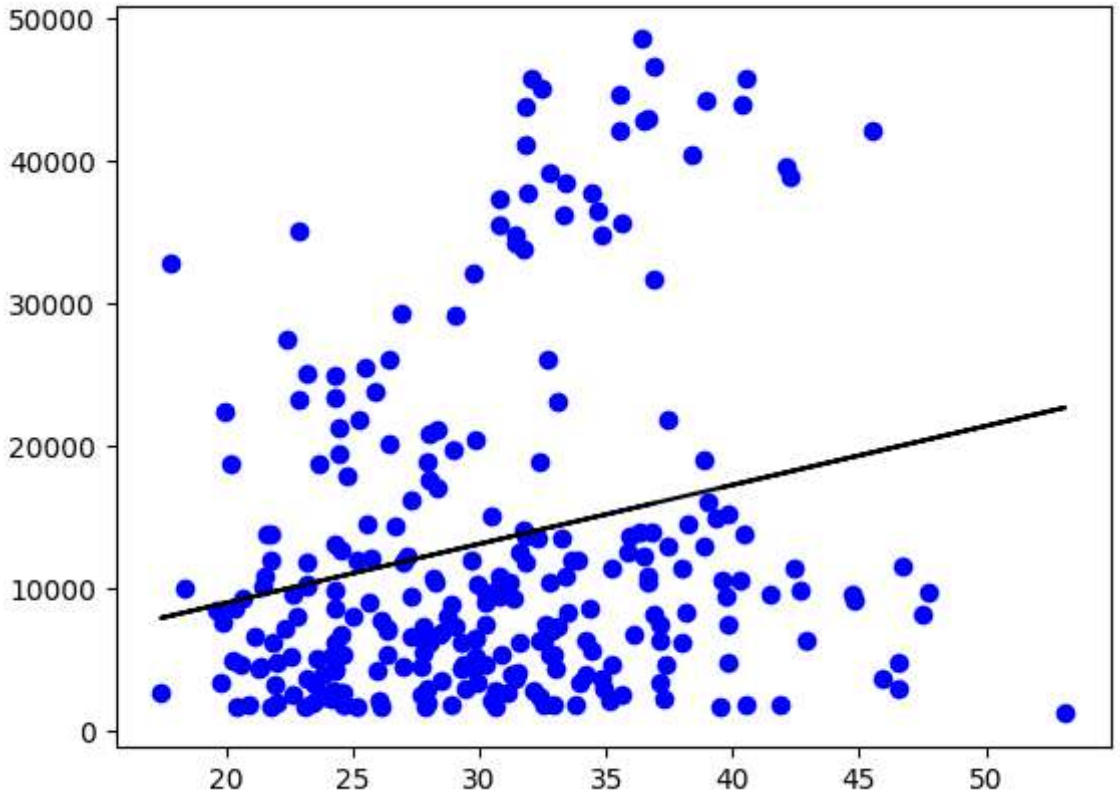


In [23]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
lr=LinearRegression()
```

In [24]:
```
lr.fit(x_train,y_train)
print(lr.score(x_test,y_test))
```

```
0.030715187812894063
```

In [25]:
```
y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



Logistic Regression

In [26]:
```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [27]:
```
df=pd.read_csv(r"C:\Users\Sushma sree\Downloads\insurance.csv")
df
```

Out[27]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [28]:
```
print('This Dataset has %d rows and %d columns'%(df.shape))
```

```
This Dataset has 1338 rows and 7 columns
```

In [29]:
```
df.head()
```

Out[29]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

In [30]:
```
df.tail()
```

Out[30]:

	age	sex	bmi	children	smoker	region	charges
1333	50	male	30.97	3	no	northwest	10600.5483
1334	18	female	31.92	0	no	northeast	2205.9808
1335	18	female	36.85	0	no	southeast	1629.8335
1336	21	female	25.80	0	no	southwest	2007.9450
1337	61	female	29.07	0	yes	northwest	29141.3603

In [31]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object 
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object 
 5   region    1338 non-null   object 
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

In [32]:
```
1 df.isnull().sum()
```

Out[32]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [33]:
```
1 convert={"smoker":{"yes":1,"no":0}}
2 df=df.replace(convert)
3 df
```

Out[33]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	1	southwest	16884.92400
1	18	male	33.770	1	0	southeast	1725.55230
2	28	male	33.000	3	0	southeast	4449.46200
3	33	male	22.705	0	0	northwest	21984.47061
4	32	male	28.880	0	0	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	0	northwest	10600.54830
1334	18	female	31.920	0	0	northeast	2205.98080
1335	18	female	36.850	0	0	southeast	1629.83350
1336	21	female	25.800	0	0	southwest	2007.94500
1337	61	female	29.070	0	1	northwest	29141.36030

1338 rows × 7 columns

In [34]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[34]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	southwest	16884.92400
1	18	0	33.770	1	0	southeast	1725.55230
2	28	0	33.000	3	0	southeast	4449.46200
3	33	0	22.705	0	0	northwest	21984.47061
4	32	0	28.880	0	0	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	0	30.970	3	0	northwest	10600.54830
1334	18	1	31.920	0	0	northeast	2205.98080
1335	18	1	36.850	0	0	southeast	1629.83350
1336	21	1	25.800	0	0	southwest	2007.94500
1337	61	1	29.070	0	1	northwest	29141.36030

1338 rows × 7 columns

In [35]:
```
convert={"region":{"southeast":1,"southwest":2,"northeast":3,"northwest":4
df=df.replace(convert)
df
```

Out[35]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	2	16884.92400
1	18	0	33.770	1	0	1	1725.55230
2	28	0	33.000	3	0	1	4449.46200
3	33	0	22.705	0	0	4	21984.47061
4	32	0	28.880	0	0	4	3866.85520
...	...	...	...	...	...	...	...
1333	50	0	30.970	3	0	4	10600.54830
1334	18	1	31.920	0	0	3	2205.98080
1335	18	1	36.850	0	0	1	1629.83350
1336	21	1	25.800	0	0	2	2007.94500
1337	61	1	29.070	0	1	4	29141.36030

1338 rows × 7 columns

In [36]:
```
features_matrix=df.iloc[:,0:4]
```

In [37]:
```
target_vector=df.iloc[:,-3]
```

In [38]:
```
print('The Feature Matrix has %d Rows and %d columns(s)'%(features_matrix
print('The Target Matrix has %d Rows and %d columns(s)'%(np.array(target_
```

```
The Feature Matrix has 1338 Rows and 4 columns(s)
The Target Matrix has 1338 Rows and 1 columns(s)
```

In [41]:
```
features_matrix_standardized=StandardScaler().fit_transform(features_matr
```

In [42]:
```
lgr=LogisticRegression(max_iter=10000)
```

In [43]:
```
Logistic_Regression_Model=lgr.fit(features_matrix_standardized,target_vec
```

In [44]:
```
observation=[[1,0,0.99539,-0.0588]]
```

In [45]:
```
predictions=Logistic_Regression_Model.predict(observation)
print('The model predicted the observation to belong to class %s'%(predic
```

```
The model predicted the observation to belong to class [0]
```

In [46]:
```
print('The algoritham was trained to predict one of the two classes:%s'%(
```

```
The algoritham was trained to predict one of the two classes:[0 1]
```

In [47]:
```
print("The model says the probability of the observation we passed belong
print("The Model says the probability of the observation we passed belong
```

```
The model says the probability of the observation we passed belonging to clas
s[0] Is 0.8057075871331396
The Model says the probability of the observation we passed belonging to clas
s['1'] Is 0.8057075871331396
```

In [48]:
```
x=np.array(df['age']).reshape(-1,1)
y=np.array(df['smoker']).reshape(-1,1)
```

In [49]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.05)
lr=LogisticRegression()
```

In [50]:
```
lr.fit(x_train,y_train)
print(lr.score(x_test,y_test))
```

```
0.8208955223880597

C:\Users\Sushma sree\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\utils\validation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Decision Tree

In [51]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [52]:
```
df=pd.read_csv(r"C:\Users\Sushma sree\Downloads\insurance.csv")
df
```

Out[52]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [53]:
```
1 df.head()
```

Out[53]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520

In [54]:
```
1 df.tail()
```

Out[54]:

	age	sex	bmi	children	smoker	region	charges
1333	50	male	30.97	3	no	northwest	10600.5483
1334	18	female	31.92	0	no	northeast	2205.9808
1335	18	female	36.85	0	no	southeast	1629.8335
1336	21	female	25.80	0	no	southwest	2007.9450
1337	61	female	29.07	0	yes	northwest	29141.3603

In [55]:
```
1 convert={'smoker':{'yes':1,'no':0}}
2 df=df.replace(convert)
3 df
```

Out[55]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	1	southwest	16884.92400
1	18	male	33.770	1	0	southeast	1725.55230
2	28	male	33.000	3	0	southeast	4449.46200
3	33	male	22.705	0	0	northwest	21984.47061
4	32	male	28.880	0	0	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	0	northwest	10600.54830
1334	18	female	31.920	0	0	northeast	2205.98080
1335	18	female	36.850	0	0	southeast	1629.83350
1336	21	female	25.800	0	0	southwest	2007.94500
1337	61	female	29.070	0	1	northwest	29141.36030

1338 rows × 7 columns

In [56]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[56]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	southwest	16884.92400
1	18	0	33.770	1	0	southeast	1725.55230
2	28	0	33.000	3	0	southeast	4449.46200
3	33	0	22.705	0	0	northwest	21984.47061
4	32	0	28.880	0	0	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	0	30.970	3	0	northwest	10600.54830
1334	18	1	31.920	0	0	northeast	2205.98080
1335	18	1	36.850	0	0	southeast	1629.83350
1336	21	1	25.800	0	0	southwest	2007.94500
1337	61	1	29.070	0	1	northwest	29141.36030

1338 rows × 7 columns

In [57]:
```
x=["bmi","children"]
y=["Yes","No"]
all_inputs=df[x]
all_classes=df["sex"]
```

In [58]:
```
(x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,t
```

In [59]:
```
clf=DecisionTreeClassifier(random_state=0)
```

In [60]:
```
clf.fit(x_train,y_train)
```

Out[60]:
```
DecisionTreeClassifier(random_state=0)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [61]:
```
score=clf.score(x_test,y_test)
print(score)
```

```
0.4146341463414634
```

Random Forest

In [62]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt ,seaborn as sns
```

In [63]:
```
df=pd.read_csv(r"C:\Users\Sushma sree\Downloads\insurance.csv")
df
```

Out[63]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [64]:
```
df['region'].value_counts()
```

Out[64]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [65]:
```
df['bmi'].value_counts()
```

Out[65]:
```
bmi
32.300    13
28.310     9
30.495     8
30.875     8
31.350     8
          ..
46.200     1
23.800     1
44.770     1
32.120     1
30.970     1
Name: count, Length: 548, dtype: int64
```

In [66]:
```
m={"sex":{"female":1,"male":0}}
df=df.replace(m)
print(df)
```

```
      age  sex     bmi  children smoker     region      charges
0      19    1  27.900         0    yes  southwest  16884.92400
1      18    0  33.770         1     no  southeast   1725.55230
2      28    0  33.000         3     no  southeast   4449.46200
3      33    0  22.705         0     no  northwest  21984.47061
4      32    0  28.880         0     no  northwest   3866.85520
...   ...  ...     ...       ...    ...        ...          ...
1333   50    0  30.970         3     no  northwest  10600.54830
1334   18    1  31.920         0     no  northeast   2205.98080
1335   18    1  36.850         0     no  southeast   1629.83350
1336   21    1  25.800         0     no  southwest   2007.94500
1337   61    1  29.070         0    yes  northwest  29141.36030

[1338 rows x 7 columns]
```

In [67]:
```
n={"smoker":{"yes":1,"no":0}}
df=df.replace(n)
print(df)
```

```
      age  sex     bmi  children  smoker     region      charges
0      19    1  27.900         0       1  southwest  16884.92400
1      18    0  33.770         1       0  southeast   1725.55230
2      28    0  33.000         3       0  southeast   4449.46200
3      33    0  22.705         0       0  northwest  21984.47061
4      32    0  28.880         0       0  northwest   3866.85520
...   ...  ...     ...       ...     ...        ...          ...
1333   50    0  30.970         3       0  northwest  10600.54830
1334   18    1  31.920         0       0  northeast   2205.98080
1335   18    1  36.850         0       0  southeast   1629.83350
1336   21    1  25.800         0       0  southwest   2007.94500
1337   61    1  29.070         0       1  northwest  29141.36030

[1338 rows x 7 columns]
```

In [68]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[68]:
```
RandomForestClassifier()
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [69]:
```
rf=RandomForestClassifier()
params={'max_depth':[2,3,5,20],
'min_samples_leaf':[5,10,20,50,100,200],
'n_estimators':[10,25,30,50,100,200]}
```

In [70]:
```
x=df[['bmi']]
y=df['smoker']
```

In [71]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rf,param_grid=params,cv=2,scoring="acc
grid_search.fit(x_train,y_train)
```

Out[71]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [2, 3, 5, 20],
                         'min_samples_leaf': [5, 10, 20, 50, 100, 200],
                         'n_estimators': [10, 25, 30, 50, 100, 200]},
             scoring='accuracy')
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [72]:
```
grid_search.best_score_
```

Out[72]:
```
0.5165746447526108
```

In [73]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=3, min_samples_leaf=200, n_estimators=200)
```

In [74]:
```
rf_best.feature_importances_
```

Out[74]:
```
array([0.67662889, 0.32337111])
```

In [75]:
```
rf=RandomForestClassifier(random_state=0)
```

In [76]:
```
1 rf.fit(x_train,y_train)
```

Out[76]:
```
RandomForestClassifier(random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [77]:
```
1 score=rf.score(x_test,y_test)
2 print(score)
```

```
0.4878048780487805
```

Conclusion:

After performing all models the to the given dataset. Among all models "Logistic Regression" has 80% accuracy

In [ ]:
```
1
```