Mini Project 1

Problem Statement: Which model is suitable for Insurance dataset

Importing Packages

In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression,LogisticRegression
from sklearn.metrics import r2_score
```

Read data

In [2]:

```
df=pd.read_csv(r"C:\Users\P. VIJAY KUMAR\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 Pre-processing

In [3]:

```
df.isnull().any()
```

Out[3]:

```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [4]:

```
df.columns
```

Out[4]:

```
Index(['age', 'sex', 'bmi', 'children', 'smoker', 'region', 'charges'], dtype
='object')
```

In [5]:

```
df["sex"].value_counts()
```

Out[5]:

```
sex
male      676
female    662
Name: count, dtype: int64
```

In [6]:

```
convert={"sex":{"male":1,"female":2}}
df=df.replace(convert)
df
```

Out[6]:

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

1338 rows × 7 columns

In [7]:

```
df["smoker"].value_counts()
```

Out[7]:

```
smoker
no     1064
yes     274
Name: count, dtype: int64
```

In [8]:

```
convert={"smoker":{"no":1,"yes":2}}
df=df.replace(convert)
df
```

Out[8]:

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

1338 rows × 7 columns

In [9]:

```
df["region"].value_counts()
```

Out[9]:

```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [10]:

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

Out[10]:

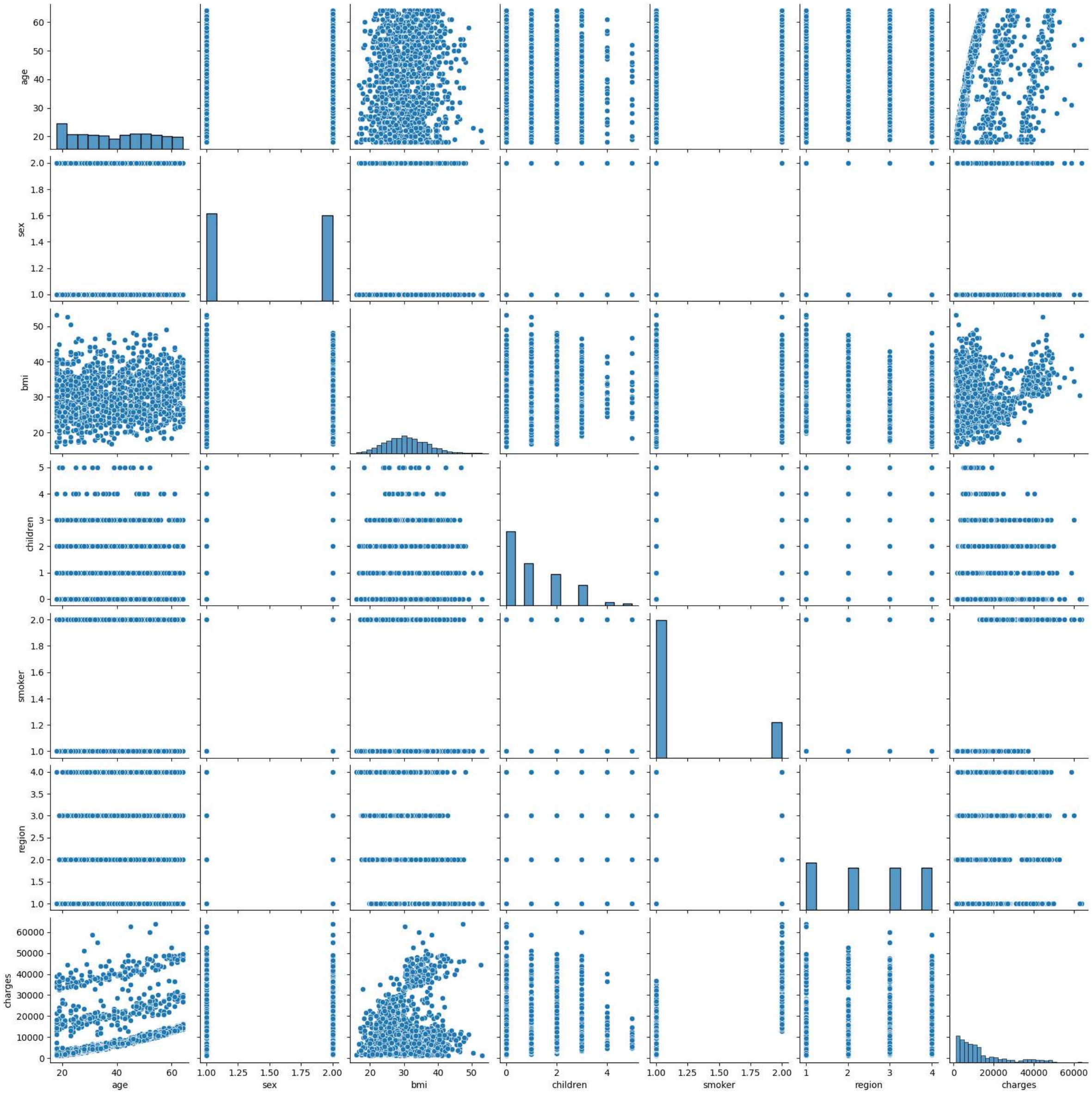
	age	sex	bmi	children	smoker	region	charges
0	19	2	27.900	0	2	2	16884.92400
1	18	1	33.770	1	1	1	1725.55230
2	28	1	33.000	3	1	1	4449.46200
3	33	1	22.705	0	1	3	21984.47061
4	32	1	28.880	0	1	3	3866.85520
...	...	...	...	...	...	...	...
1333	50	1	30.970	3	1	3	10600.54830
1334	18	2	31.920	0	1	4	2205.98080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

1338 rows × 7 columns

Data Visualization

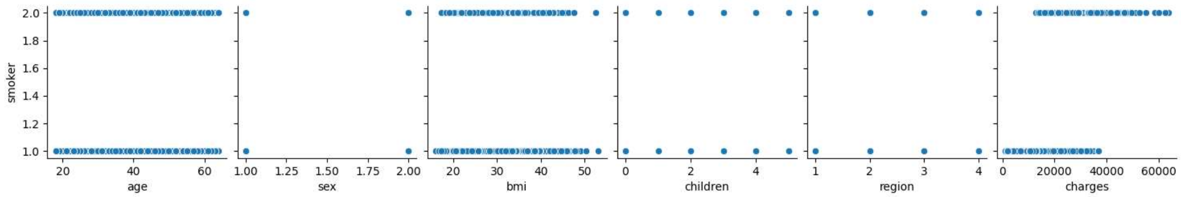
In [11]:
```
sns.pairplot(df)
```

Out[11]:
```
<seaborn.axisgrid.PairGrid at 0x25ddca779a0>
```



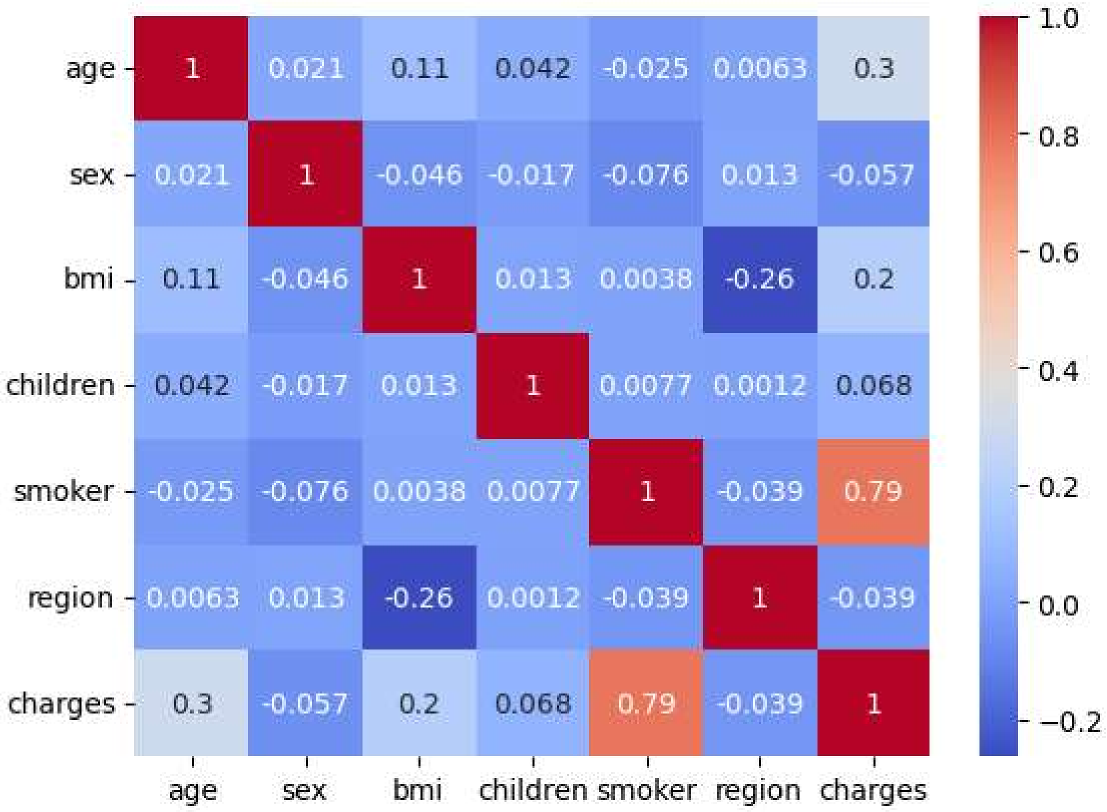
In [12]:
```
columns_to_plot=df[['age', 'sex', 'bmi', 'children', 'smoker', 'region',
sns.pairplot(columns_to_plot,x_vars=['age', 'sex', 'bmi', 'children', 're
              aspect=1,kind='scatter')
```

Out[12]:
```
<seaborn.axisgrid.PairGrid at 0x25d837402e0>
```



In [13]:
```
#To plot heatmap to find out correlations
subset_data =df[['age', 'sex', 'bmi', 'children', 'smoker', 'region', 'ch
correlation_matrix = subset_data.corr()  # Use .corr() for correlation
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
```

Out[13]:
```
<Axes: >
```



Feature Scaling: Splitting data into training and testing

In [14]:
```
X=np.array(df["smoker"]).reshape(-1,1)
y=np.array(df["charges"]).reshape(-1,1)
```

In [15]:
```
x_train,x_test,y_train,y_test=train_test_split(X,y,train_size=0.7,random_
```

Applying Linear Regression

In [16]:
```
lr=LinearRegression()
```

In [17]:

```
lr.fit(x_train,y_train)
```

Out[17]:

```
LinearRegression()
```

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 [18]:

```
lr.score(x_test,y_test)
```

Out[18]:

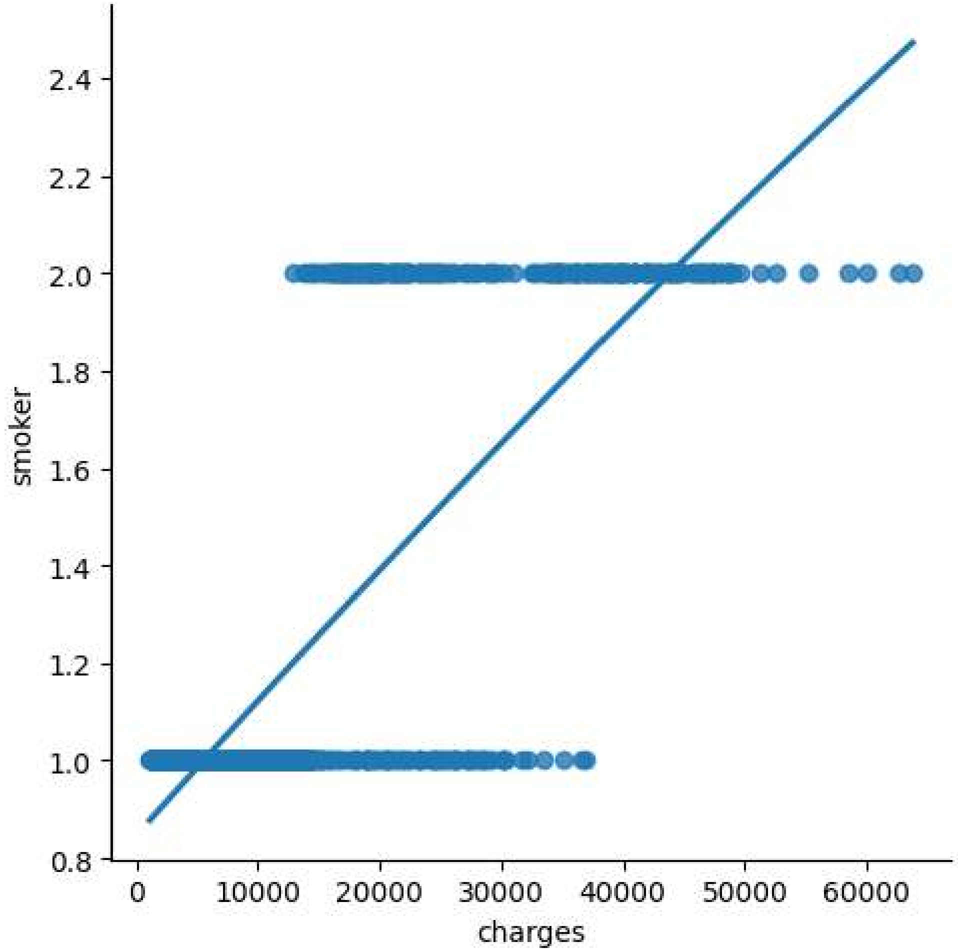
```
0.6197902354385714
```

In [19]:

```
sns.lmplot(x="charges",y="smoker",data=df,order=2,ci=None)
```

Out[19]:

```
<seaborn.axisgrid.FacetGrid at 0x25d834081f0>
```



Applying Logistic Regression

In [20]:

```
x=np.array(df["charges"]).reshape(-1,1)
y=np.array(df["smoker"]).reshape(-1,1)
df.dropna(inplace=True)
```

In [21]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_s
```

In [22]:

```
lg=LogisticRegression(max_iter=10000)
```

In [23]:

```
lg.fit(x_train,y_train)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Roaming\Python\Python310\site-packages\sklear
n\utils\validation.py:1143: DataConversionWarning: A column-vector y was pass
ed 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)
```

Out[23]:

```
LogisticRegression(max_iter=10000)
```

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 [24]:

```
lg.score(x_test,y_test)
```

Out[24]:

```
0.8930348258706468
```

In [25]:

```
prediction=lg.predict(x_test)
r2=r2_score(y_test,prediction)
r2
```

Out[25]:

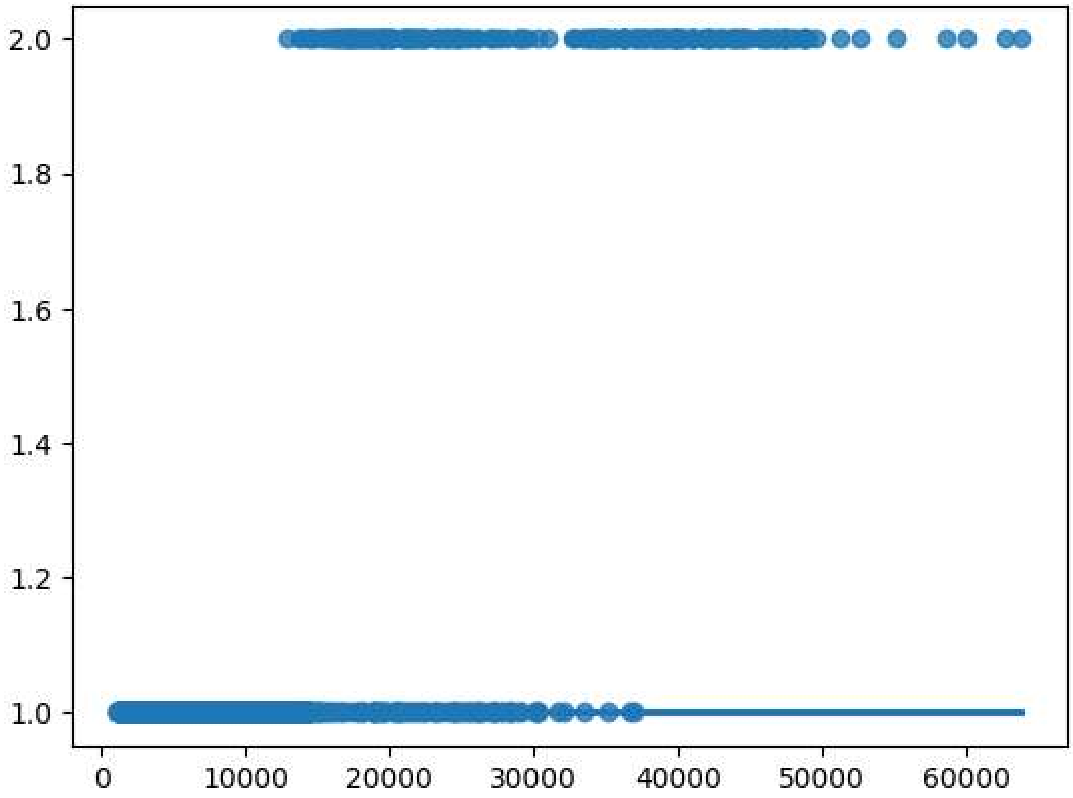
```
0.335179416176301
```

In [28]:

```
sns.regplot(x=x,y=y,data=df,logistic=True,ci=None)
```

Out[28]:

```
<Axes: >
```



Applying Decision Tree

In [29]:

```
from sklearn.tree import DecisionTreeClassifier
dtl=DecisionTreeClassifier(random_state=0)
```

In [30]:

```
dtl.fit(x_train,y_train)
```

Out[30]:

```
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 [32]:

```
dtl.score(x_test,y_test)
```

Out[32]:

```
0.8880597014925373
```

Applying Random Forest

In [33]:

```python
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
```

In [34]:

```python
rfc.fit(x_train,y_train)
```

```
C:\Users\P. VIJAY KUMAR\AppData\Local\Temp\ipykernel_17540\4070307935.py:1: D
ataConversionWarning: A column-vector y was passed when a 1d array was expect
ed. Please change the shape of y to (n_samples,), for example using ravel().
  rfc.fit(x_train,y_train)
```

Out[34]:

```
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 [35]:

```python
rfc.score(x_test,y_test)
```

Out[35]:

```
0.8880597014925373
```

CONCLUSION: Here i developed LinearRegression model, LogisticRegression model, Decision Tree model and RandomForest model for provided dataset.Among them the LogisticRegression has got more accuracy on given dataset, So LogisticRegression is best fittedmodel for our datset.

In [ ]:

```python

```