

```
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\91628\Downloads\insurance2.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 and Preprocessing

```
In [3]: 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]: 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]: df.shape
```

Out[5]: (1338, 7)

In [6]: df.describe

```
Out[6]: <bound method NDFrame.describe of
   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 [7]: 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 [8]: df.isnull().any()

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

In [9]: df.isna().sum()

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

In [10]: df['region'].value_counts()

```
Out[10]: region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

```
In [11]: convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[11]:

| | 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 [12]: convert={"region":{"southeast":0,"southwest":1,"northwest":3,"northeast":4}}
df=df.replace(convert)
df
```

Out[12]:

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

1338 rows × 7 columns

```
In [13]: convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

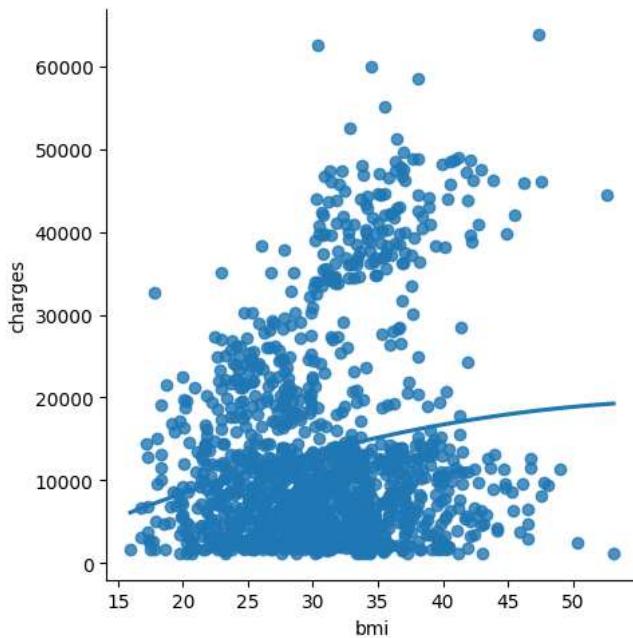
Out[13]:

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

1338 rows × 7 columns

Data Visualization

```
In [14]: sns.lmplot(x='bmi',y='charges',order=2,data=df,ci=None)
plt.show()
```

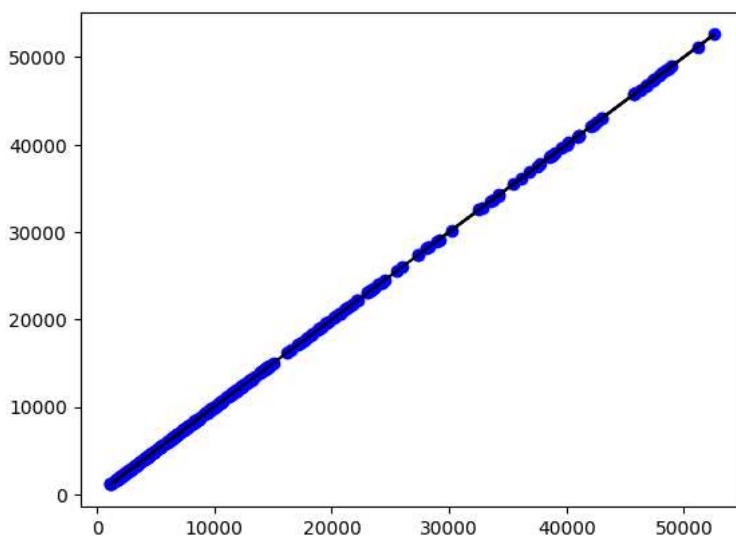


```
In [15]: x=np.array(df['bmi']).reshape(-1,1)
y=x=np.array(df['charges']).reshape(-1,1)
```

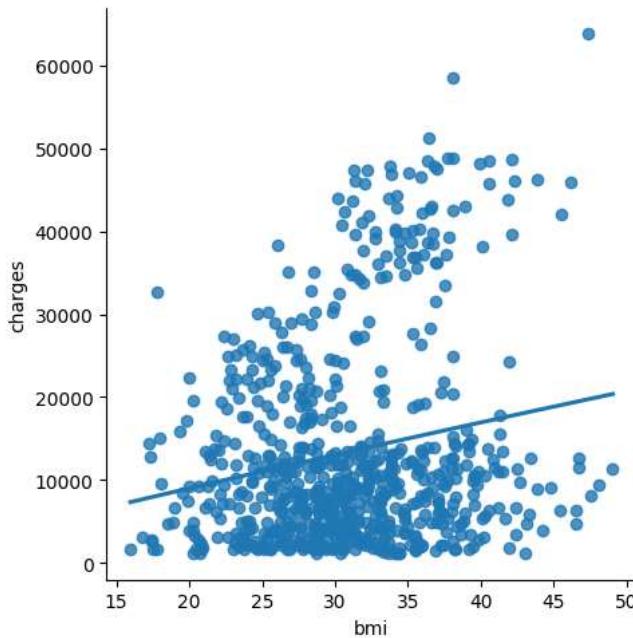
```
In [16]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=0)
lr=LinearRegression()
lr.fit(x_train,y_train)
print(lr.score(x_test,y_test))
```

1.0

```
In [17]: y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



```
In [18]: df700=df[:][:700]
sns.lmplot(x='bmi',y='charges',order=2,ci=None,data=df700)
plt.show()
```



```
In [19]: df700.fillna(method='ffill',inplace=True)
```

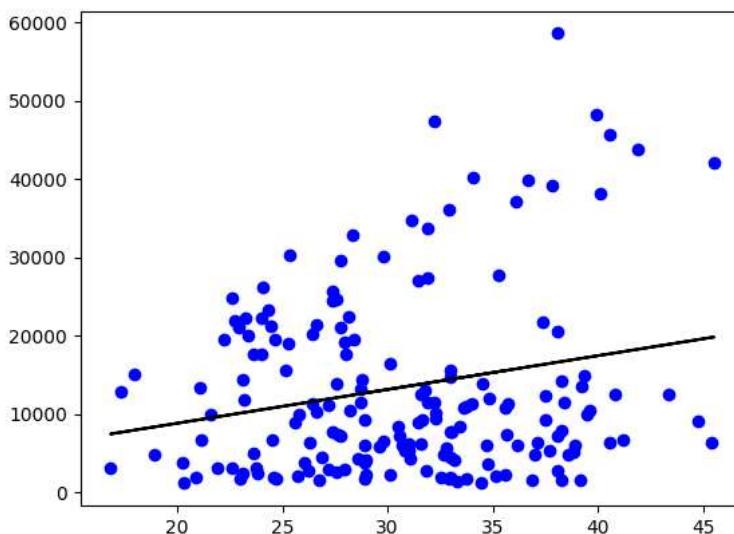
```
In [20]: x=np.array(df700["bmi"]).reshape(-1,1)
y=np.array(df700['charges']).reshape(-1,1)
```

```
In [21]: df700.dropna(inplace=True)
```

```
In [22]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
lr=LinearRegression()
lr.fit(x_train,y_train)
print(lr.score(x_test,y_test))
```

0.015100548342128928

```
In [23]: y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



Evaluation of model

```
In [24]: from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

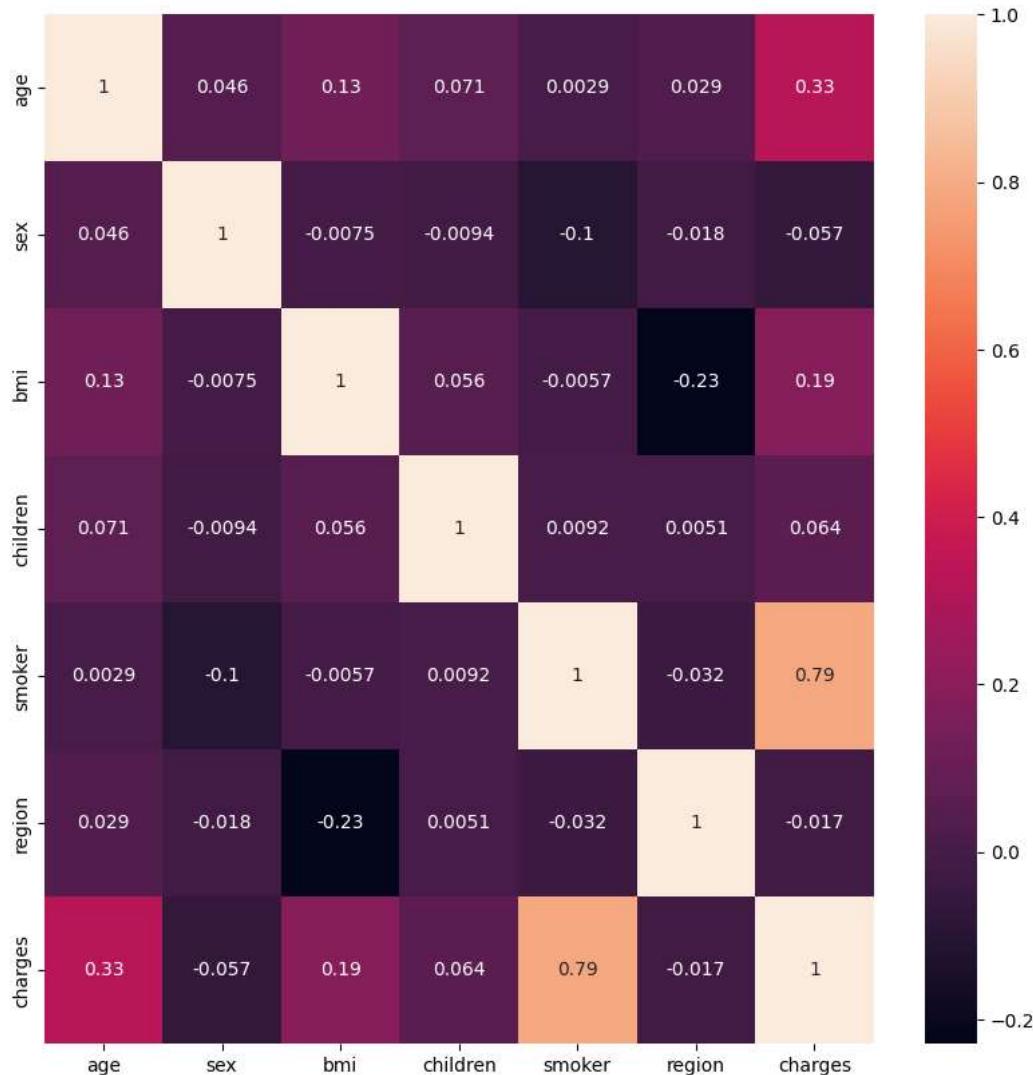
```
In [25]: lr=LinearRegression()
lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
r2=r2_score(y_test,y_pred)
print(r2)
```

0.015100548342128928

Ridge Regression

```
In [26]: from sklearn.linear_model import Lasso,Ridge
from sklearn.preprocessing import StandardScaler
```

```
In [27]: plt.figure(figsize=(10,10))
sns.heatmap(df700.corr(),annot=True)
plt.show()
```



```
In [28]: features=df.columns[0:1]
target=df.columns[-1]
```

```
In [29]: x=df[features].values
y=df[target].values
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=1)
print("The dimension of X_train is {}".format(x_train.shape))
print("The dimension of X_test is {}".format(x_test.shape))

The dimension of X_train is (936, 1)
The dimension of X_test is (402, 1)
```

```
In [30]: lr = LinearRegression()
#Fit model
lr.fit(x_train, y_train)
#predict
actual = y_test
train_score_lr = lr.score(x_train, y_train)
test_score_lr = lr.score(x_test, y_test)
print("\nLinear Regression Model:\n")
print("The train score for lr model is {}".format(train_score_lr))
print("The test score for lr model is {}".format(test_score_lr))
```

Linear Regression Model:

The train score for lr model is 0.0910963973805714
 The test score for lr model is 0.08490473916580776

```
In [31]: ridgeReg = Ridge(alpha=10)
ridgeReg.fit(x_train,y_train)
#train and test scorefor ridge regression
train_score_ridge = ridgeReg.score(x_train, y_train)
test_score_ridge = ridgeReg.score(x_test, y_test)
print("\nRidge Model:\n")
print("The train score for ridge model is {}".format(train_score_ridge))
print("The test score for ridge model is {}".format(test_score_ridge))
```

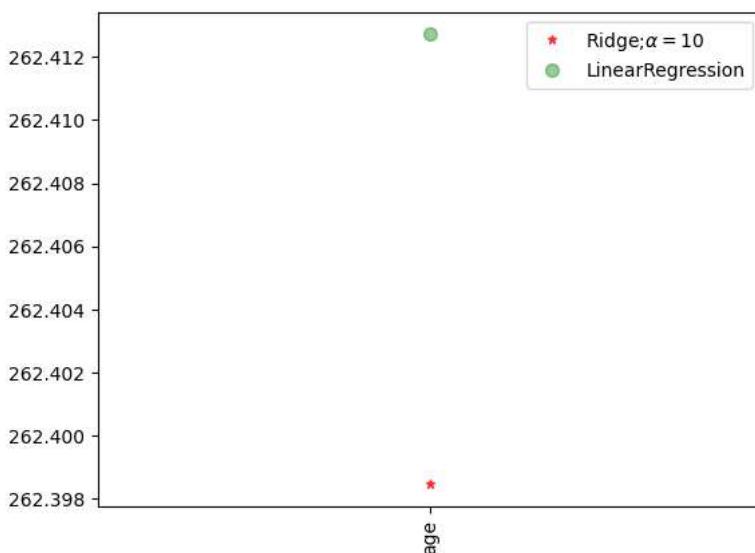
Ridge Model:

The train score for ridge model is 0.09109639711159634
 The test score for ridge model is 0.08490538609860176

```
In [32]: plt.figure(figsize=(10,10))
```

```
Out[32]: <Figure size 1000x1000 with 0 Axes>
<Figure size 1000x1000 with 0 Axes>
```

```
In [33]: .plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker="*",markersize=5,color='red',label=r'Ridge;\$\alpha=10$',zorder=7)
plot(features,lr.coef_,alpha=0.4,linestyle='none',marker="o",markersize=7,color='green',label='LinearRegression')
xticks(rotation=90)
legend()
show()
```



Lasso Regression

```
In [34]: lasso= Lasso(alpha=10)
lasso.fit(x_train,y_train)
#train and test score for ridge regression
train_score_ls = lasso.score(x_train, y_train)
test_score_ls= lasso.score(x_test, y_test)
print("\nRidge Model:\n")
print("The train score for lasso model is {}".format(train_score_ls))
print("The test score for lasso model is {}".format(test_score_ls))
```

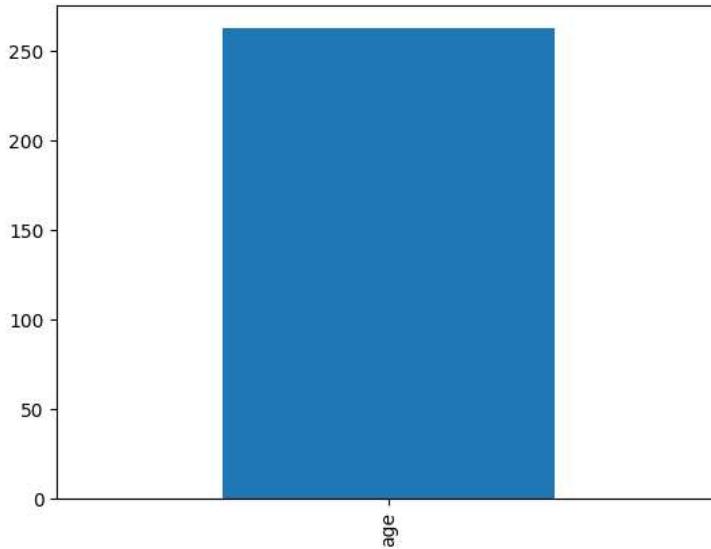
Ridge Model:

The train score for lasso model is 0.09109639395809044
 The test score for lasso model is 0.08490704421828055

```
In [35]: plt.figure(figsize=(10,10))
```

```
Out[35]: <Figure size 1000x1000 with 0 Axes>
<Figure size 1000x1000 with 0 Axes>
```

```
In [37]: pd.Series(lasso.coef_,features).sort_values(ascending=True).plot(kind="bar")
plt.show()
```



```
In [38]: from sklearn.linear_model import LassoCV
```

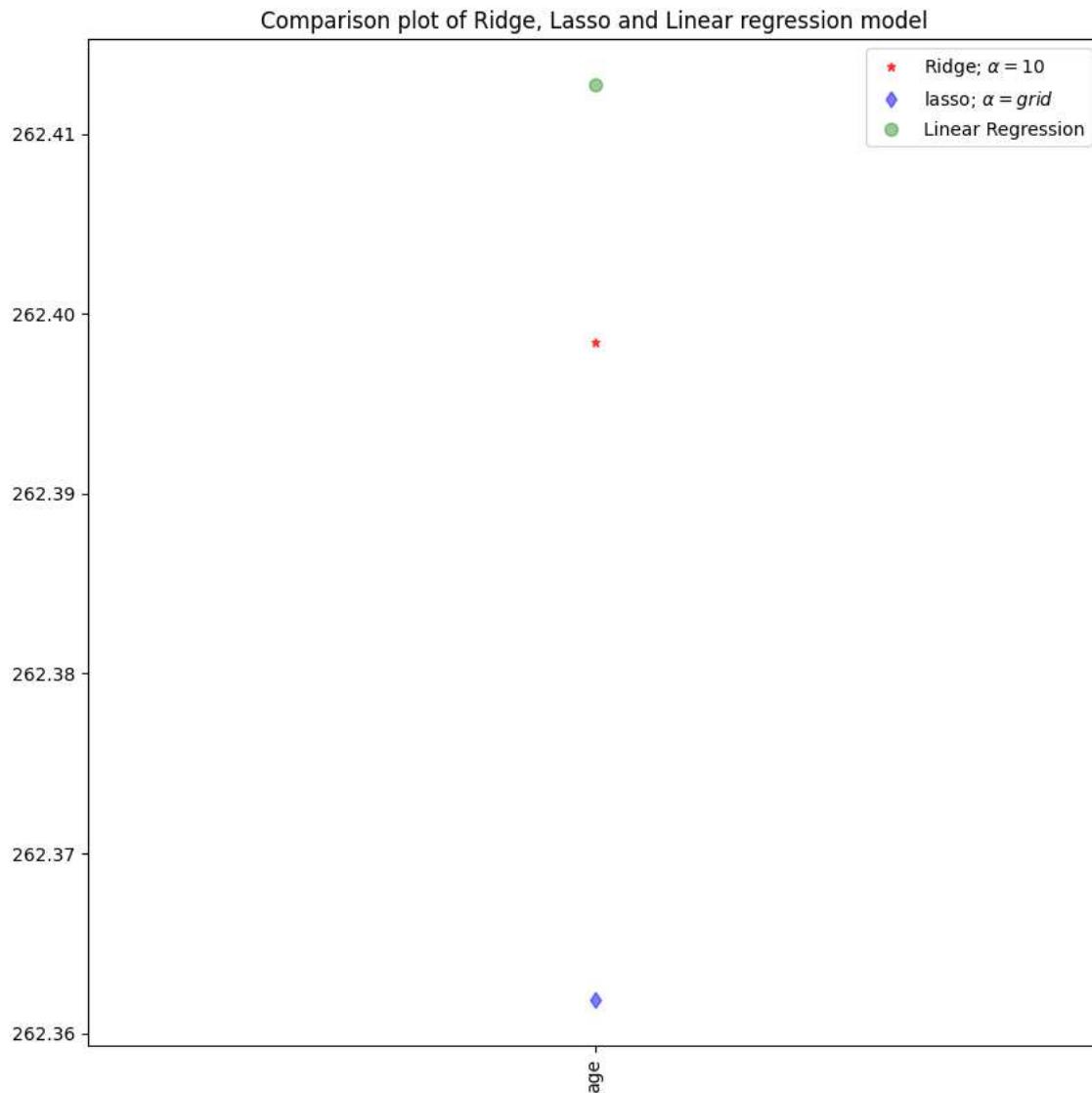
```
In [39]: from sklearn.linear_model import RidgeCV
ridge_cv=RidgeCV(alphas =[0.0001,0.001,0.01,0.1,1,10]).fit(x_train,y_train)
#score
print(ridge_cv.score(x_train,y_train))
print(ridge_cv.score(x_test,y_test))
```

0.09109639711159612
 0.08490538609884613

```
In [40]: #using the Linear cv model
from sklearn.linear_model import LassoCV
#cross validation
lasso_cv=LassoCV(alphas =[0.0001,0.001,0.01,0.1,1,10]).fit(x_train,y_train)
#score
print(lasso_cv.score(x_train,y_train))
print(lasso_cv.score(x_test,y_test))
```

0.09109639395809044
 0.08490704421828055

```
In [41]: .figure(figsize = (10, 10))
    plot for ridge regression
    plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,color='red',label=r'Ridge; $\alpha=10$',zorder=7)
    plot for lasso regression
    plot(lasso_cv.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,color='blue',label=r'lasso; $\alpha = \text{grid}$')
    plot for linear model
    plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,color='green',label='Linear Regression')
    ate axis
    xticks(rotation = 90)
    legend()
    title("Comparison plot of Ridge, Lasso and Linear regression model")
    show()
```



ElasticNet Regression

```
In [42]: from sklearn.linear_model import ElasticNet
```

```
In [43]: el=ElasticNet()
el.fit(x_train,y_train)
print(el.coef_)
print(el.intercept_)
```

```
[261.74450967]
3115.0831774262424
```

```
In [44]: y_pred_elastic=el.predict(x_train)
```

```
In [45]: mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print(mean_squared_error)
```

135077142.70714515

```
In [46]: el=ElasticNet()
el.fit(x_train,y_train)
print(el.score(x_train,y_train))
```

0.09109580670592365

Logistic Regression

```
In [47]: import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

```
In [48]: df=pd.read_csv(r"C:\Users\91628\Downloads\insurance2.csv")
df
```

Out[48]:

| | 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 [49]: df.shape
```

Out[49]: (1338, 7)

```
In [50]: pd.set_option('display.max_rows',1000000000)
pd.set_option('display.max_columns',1000000000)
pd.set_option('display.width',95)
```

```
In [51]: print('This Dataset has %d rows and %d columns'%(df.shape))
```

This Dataset has 1338 rows and 7 columns

```
In [52]: df.head()
```

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 |

In [53]: df.describe

```
Out[53]: <bound method NDFrame.describe of
   0    19  female  27.900      0  yes  southwest  16884.924000
   1    18    male  33.770      1  no  southeast  1725.552300
   2    28    male  33.000      3  no  southeast  4449.462000
   3    33    male  22.705      0  no  northwest  21984.470610
   4    32    male  28.880      0  no  northwest  3866.855200
   5    31  female  25.740      0  no  southeast  3756.621600
   6    46  female  33.440      1  no  southeast  8240.589600
   7    37  female  27.740      3  no  northwest  7281.505600
   8    37    male  29.830      2  no  northeast  6406.410700
   9    60  female  25.840      0  no  northwest  28923.136920
  10   25    male  26.220      0  no  northeast  2721.320800
  11   62  female  26.290      0  yes  southeast  27808.725100
  12   23    male  34.400      0  no  southwest  1826.843000
  13   56  female  39.820      0  no  southeast  11090.717800
  14   27    male  42.130      0  yes  southeast  39611.757700
  15   19    male  24.600      1  no  southwest  1837.237000
  16   52  female  30.780      1  no  northeast  10797.336200
  17   23    male  23.845      0  no  northeast  2395.171550
  18   ...    ...  ...  ...
  1337  ...    ...  ...  ...
  1338  ...    ...  ...  ...
```

In [54]: 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 [55]: df.isnull().sum()

```
Out[55]: age      0
sex      0
bmi      0
children 0
smoker   0
region   0
charges  0
dtype: int64
```

```
In [56]: convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

Out[56]:

| | age | sex | bmi | children | smoker | region | charges |
|----|-----|--------|--------|----------|--------|-----------|--------------|
| 0 | 19 | female | 27.900 | 0 | 1 | southwest | 16884.924000 |
| 1 | 18 | male | 33.770 | 1 | 0 | southeast | 1725.552300 |
| 2 | 28 | male | 33.000 | 3 | 0 | southeast | 4449.462000 |
| 3 | 33 | male | 22.705 | 0 | 0 | northwest | 21984.470610 |
| 4 | 32 | male | 28.880 | 0 | 0 | northwest | 3866.855200 |
| 5 | 31 | female | 25.740 | 0 | 0 | southeast | 3756.621600 |
| 6 | 46 | female | 33.440 | 1 | 0 | southeast | 8240.589600 |
| 7 | 37 | female | 27.740 | 3 | 0 | northwest | 7281.505600 |
| 8 | 37 | male | 29.830 | 2 | 0 | northeast | 6406.410700 |
| 9 | 60 | female | 25.840 | 0 | 0 | northwest | 28923.136920 |
| 10 | 25 | male | 26.220 | 0 | 0 | northeast | 2721.320800 |

```
In [57]: convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[57]:

| | age | sex | bmi | children | smoker | region | charges |
|----|-----|-----|--------|----------|--------|-----------|--------------|
| 0 | 19 | 1 | 27.900 | 0 | 1 | southwest | 16884.924000 |
| 1 | 18 | 0 | 33.770 | 1 | 0 | southeast | 1725.552300 |
| 2 | 28 | 0 | 33.000 | 3 | 0 | southeast | 4449.462000 |
| 3 | 33 | 0 | 22.705 | 0 | 0 | northwest | 21984.470610 |
| 4 | 32 | 0 | 28.880 | 0 | 0 | northwest | 3866.855200 |
| 5 | 31 | 1 | 25.740 | 0 | 0 | southeast | 3756.621600 |
| 6 | 46 | 1 | 33.440 | 1 | 0 | southeast | 8240.589600 |
| 7 | 37 | 1 | 27.740 | 3 | 0 | northwest | 7281.505600 |
| 8 | 37 | 0 | 29.830 | 2 | 0 | northeast | 6406.410700 |
| 9 | 60 | 1 | 25.840 | 0 | 0 | northwest | 28923.136920 |
| 10 | 25 | 0 | 26.220 | 0 | 0 | northeast | 2721.320800 |

```
In [58]: convert={"region":{"southeast":1,"southwest":2,"northeast":3,"northwest":4}}
df=df.replace(convert)
df
```

| | | | | | | | |
|----|----|---|--------|---|---|---|--------------|
| 34 | 28 | 0 | 36.400 | 1 | 1 | 2 | 51194.559140 |
| 35 | 19 | 0 | 20.425 | 0 | 0 | 4 | 1625.433750 |
| 36 | 62 | 1 | 32.965 | 3 | 0 | 4 | 15612.193350 |
| 37 | 26 | 0 | 20.800 | 0 | 0 | 2 | 2302.300000 |
| 38 | 35 | 0 | 36.670 | 1 | 1 | 3 | 39774.276300 |
| 39 | 60 | 0 | 39.900 | 0 | 1 | 2 | 48173.361000 |
| 40 | 24 | 1 | 26.600 | 0 | 0 | 3 | 3046.062000 |
| 41 | 31 | 1 | 36.630 | 2 | 0 | 1 | 4949.758700 |
| 42 | 41 | 0 | 21.780 | 1 | 0 | 1 | 6272.477200 |
| 43 | 37 | 1 | 30.800 | 2 | 0 | 1 | 6313.759000 |
| 44 | 38 | 0 | 37.050 | 1 | 0 | 3 | 6079.671500 |
| 45 | 55 | 0 | 37.300 | 0 | 0 | 2 | 20630.283510 |
| 46 | 18 | 1 | 38.665 | 2 | 0 | 3 | 3393.356350 |

In [61]: features_matrix=df.iloc[:,0:4]

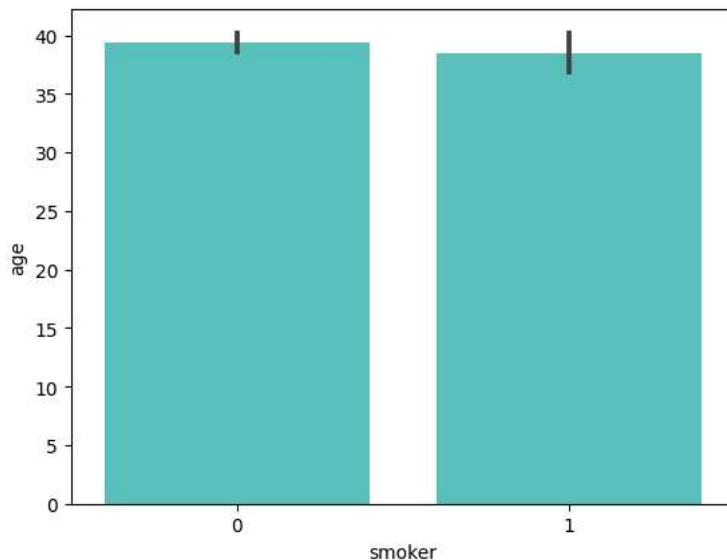
In [62]: target_vector=df.iloc[:,-3]

```
In [65]: print('The Feature Matrix has %d Rows and %d columns(s)'%(features_matrix.shape))
print('The Target Matrix has %d Rows and %d columns(s)'%(np.array(target_vector).reshape(-1,1).shape))
```

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

```
In [66]: import matplotlib.pyplot as plt
import seaborn as sns
```

```
In [67]: sns.barplot(x='smoker', y='age', data=df, color="mediumturquoise")
plt.show()
```



```
In [68]: features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

```
In [69]: algorithm=LogisticRegression(max_iter=10000)
```

```
In [70]: Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_vector)
```

```
In [71]: observation=[[1,0,0.99539,-0.0588]]
```

```
In [72]: predictions=Logistic_Regression_Model.predict(observation)
print('The model predicted the observation to belong to class %s'%(predictions))
```

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

```
In [73]: print('The algorithm was trained to predict one of the two classes:%s'%(algorithm.classes_))
```

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

```
In [75]: del says the probability of the observation we passed belonging to class[0] %s" "%(algorithm.predict_proba(observation)[0][0]))
```

```
del says the probability of the observation we passed belonging to class[1] %s" "%(algorithm.predict_proba(observation)[0][1]))
```

The Model says the probability of the observation we passed belonging to class[0] 0.8057075871331396

The Model says the probability of the observation we passed belonging to class[1] 0.19429241286686041

```
In [76]: x=np.array(df['age']).reshape(-1,1)
y=np.array(df['smoker']).reshape(-1,1)
```

```
In [77]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.05)
lo=LogisticRegression()
lo.fit(x_train,y_train)
print(lo.score(x_test,y_test))
```

0.7761194029850746

C:\Users\91628\AppData\Local\Programs\Python\Python311\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 [78]: import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

```
In [79]: df=pd.read_csv(r"C:\Users\91628\Downloads\insurance2.csv")
df
```

| | age | sex | bmi | children | smoker | region | charges | outlook |
|----|-----|--------|--------|----------|--------|-----------|--------------|-------------|
| 11 | 62 | female | 26.290 | 0 | yes | southeast | 27808.725100 | 2721.826000 |
| 12 | 23 | male | 34.400 | 0 | no | southwest | 1826.843000 | |
| 13 | 56 | female | 39.820 | 0 | no | southeast | 11090.717800 | |
| 14 | 27 | male | 42.130 | 0 | yes | southeast | 39611.757700 | |
| 15 | 19 | male | 24.600 | 1 | no | southwest | 1837.237000 | |
| 16 | 52 | female | 30.780 | 1 | no | northeast | 10797.336200 | |
| 17 | 23 | male | 23.845 | 0 | no | northeast | 2395.171550 | |
| 18 | 56 | male | 40.300 | 0 | no | southwest | 10602.385000 | |
| 19 | 30 | male | 35.300 | 0 | yes | southwest | 36837.467000 | |
| 20 | 60 | female | 36.005 | 0 | no | northeast | 13228.846950 | |
| 21 | 30 | female | 32.400 | 1 | no | southwest | 4149.736000 | |
| 22 | 18 | male | 34.100 | 0 | no | southeast | 1137.011000 | |
| 23 | 34 | female | 31.920 | 1 | yes | northeast | 37701.876800 | |

```
In [80]: df.shape
```

```
Out[80]: (1338, 7)
```

```
In [81]: df.isnull().any()
```

```
Out[81]: age      False
sex      False
bmi      False
children  False
smoker   False
region   False
charges  False
dtype: bool
```

```
In [82]: df['region'].value_counts()
```

```
Out[82]: region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

```
In [83]: convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

| | age | sex | bmi | children | smoker | region | charges | outlook |
|----|-----|-----|--------|----------|--------|-----------|--------------|---------|
| 20 | 60 | 1 | 36.005 | 0 | no | northeast | 13228.846950 | |
| 21 | 30 | 1 | 32.400 | 1 | no | southwest | 4149.736000 | |
| 22 | 18 | 0 | 34.100 | 0 | no | southeast | 1137.011000 | |
| 23 | 34 | 1 | 31.920 | 1 | yes | northeast | 37701.876800 | |
| 24 | 37 | 0 | 28.025 | 2 | no | northwest | 6203.901750 | |
| 25 | 59 | 1 | 27.720 | 3 | no | southeast | 14001.133800 | |
| 26 | 63 | 1 | 23.085 | 0 | no | northeast | 14451.835150 | |
| 27 | 55 | 1 | 32.775 | 2 | no | northwest | 12268.632250 | |
| 28 | 23 | 0 | 17.385 | 1 | no | northwest | 2775.192150 | |
| 29 | 31 | 0 | 36.300 | 2 | yes | southwest | 38711.000000 | |
| 30 | 22 | 0 | 35.600 | 0 | yes | southwest | 35585.576000 | |
| 31 | 18 | 1 | 26.315 | 0 | no | northeast | 2198.189850 | |
| 32 | 19 | 1 | 28.600 | 5 | no | southwest | 4687.797000 | |

```
In [84]: convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

| | | | | | | | |
|----|----|---|--------|---|---|-----------|--------------|
| 9 | 60 | 1 | 25.840 | 0 | 0 | northwest | 28923.136920 |
| 10 | 25 | 0 | 26.220 | 0 | 0 | northeast | 2721.320800 |
| 11 | 62 | 1 | 26.290 | 0 | 1 | southeast | 27808.725100 |
| 12 | 23 | 0 | 34.400 | 0 | 0 | southwest | 1826.843000 |
| 13 | 56 | 1 | 39.820 | 0 | 0 | southeast | 11090.717800 |
| 14 | 27 | 0 | 42.130 | 0 | 1 | southeast | 39611.757700 |
| 15 | 19 | 0 | 24.600 | 1 | 0 | southwest | 1837.237000 |
| 16 | 52 | 1 | 30.780 | 1 | 0 | northeast | 10797.336200 |
| 17 | 23 | 0 | 23.845 | 0 | 0 | northeast | 2395.171550 |
| 18 | 56 | 0 | 40.300 | 0 | 0 | southwest | 10602.385000 |
| 19 | 30 | 0 | 35.300 | 0 | 1 | southwest | 36837.467000 |
| 20 | 60 | 1 | 36.005 | 0 | 0 | northeast | 13228.846950 |
| 21 | 30 | 1 | 32.400 | 1 | 0 | southwest | 4149.736000 |

```
In [85]: x=["bmi","children"]
y=["Yes","No"]
all_inputs=df[x]
all_classes=df["sex"]
```

```
In [86]: (x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,test_size=0.03)
```

```
In [87]: clf=DecisionTreeClassifier(random_state=0)
```

```
In [88]: clf.fit(x_train,y_train)
```

```
Out[88]: DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

```
In [89]: score=clf.score(x_test,y_test)
print(score)
```

0.43902439024390244

Random Forest

```
In [90]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt,seaborn as sns
```

```
In [91]: df=pd.read_csv(r"C:\Users\91628\Downloads\insurance2.csv")
df
```

```
Out[91]:
```

| | age | sex | bmi | children | smoker | region | charges |
|----|-----|--------|--------|----------|--------|-----------|--------------|
| 0 | 19 | female | 27.900 | 0 | yes | southwest | 16884.924000 |
| 1 | 18 | male | 33.770 | 1 | no | southeast | 1725.552300 |
| 2 | 28 | male | 33.000 | 3 | no | southeast | 4449.462000 |
| 3 | 33 | male | 22.705 | 0 | no | northwest | 21984.470610 |
| 4 | 32 | male | 28.880 | 0 | no | northwest | 3866.855200 |
| 5 | 31 | female | 25.740 | 0 | no | southeast | 3756.621600 |
| 6 | 46 | female | 33.440 | 1 | no | southeast | 8240.589600 |
| 7 | 37 | female | 27.740 | 3 | no | northwest | 7281.505600 |
| 8 | 37 | male | 29.830 | 2 | no | northeast | 6406.410700 |
| 9 | 60 | female | 25.840 | 0 | no | northwest | 28923.136920 |
| 10 | 25 | male | 26.220 | 0 | no | northeast | 2721.320800 |

```
In [92]: df.shape
```

Out[92]: (1338, 7)

```
In [93]: df['region'].value_counts()
```

```
Out[93]: region
southeast    364
southwest   325
northwest   325
northeast   324
Name: count, dtype: int64
```

```
In [94]: df['bmi'].value_counts()
```

```
Out[94]: bmi
32.300    13
28.310     9
30.495     8
30.875     8
31.350     8
30.800     8
34.100     8
28.880     8
33.330     7
35.200     7
25.800     7
32.775     7
27.645     7
32.110     7
38.060     7
25.460     7
30.590     7
27.360     7
24.220     7
```

```
In [95]: m={"sex":{"female":1,"male":0}}
df=df.replace(m)
print(df)
```

| | age | sex | bmi | children | smoker | region | charges |
|-----|-----|-----|--------|----------|--------|-----------|--------------|
| 123 | 44 | 0 | 31.350 | 1 | yes | northeast | 39556.494500 |
| 124 | 47 | 1 | 33.915 | 3 | no | northwest | 10115.008850 |
| 125 | 26 | 1 | 28.785 | 0 | no | northeast | 3385.399150 |
| 126 | 19 | 1 | 28.300 | 0 | yes | southwest | 17081.080000 |
| 127 | 52 | 1 | 37.400 | 0 | no | southwest | 9634.538000 |
| 128 | 32 | 1 | 17.765 | 2 | yes | northwest | 32734.186300 |
| 129 | 38 | 0 | 34.700 | 2 | no | southwest | 6082.405000 |
| 130 | 59 | 1 | 26.505 | 0 | no | northeast | 12815.444950 |
| 131 | 61 | 1 | 22.040 | 0 | no | northeast | 13616.358600 |
| 132 | 53 | 1 | 35.900 | 2 | no | southwest | 11163.568000 |
| 133 | 19 | 0 | 25.555 | 0 | no | northwest | 1632.564450 |
| 134 | 20 | 1 | 28.785 | 0 | no | northeast | 2457.211150 |
| 135 | 22 | 1 | 28.050 | 0 | no | southeast | 2155.681500 |
| 136 | 19 | 0 | 34.100 | 0 | no | southwest | 1261.442000 |
| 137 | 22 | 0 | 25.175 | 0 | no | northwest | 2045.685250 |
| 138 | 54 | 1 | 31.900 | 3 | no | southeast | 27322.733860 |
| 139 | 22 | 1 | 36.000 | 0 | no | southwest | 2166.732000 |
| 140 | 34 | 0 | 22.420 | 2 | no | northeast | 27375.904780 |
| 141 | 26 | 0 | 32.490 | 1 | no | northeast | 3490.549100 |
| 142 | 74 | 0 | 25.200 | 0 | yes | southeast | 18077.105000 |

```
In [96]: 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 | 16884.924000 |
| 1 | 18 | 0 | 33.770 | 1 | 0 | southeast | 1725.552300 |
| 2 | 28 | 0 | 33.000 | 3 | 0 | southeast | 4449.462000 |
| 3 | 33 | 0 | 22.705 | 0 | 0 | northwest | 21984.470610 |
| 4 | 32 | 0 | 28.880 | 0 | 0 | northwest | 3866.855200 |
| 5 | 31 | 1 | 25.740 | 0 | 0 | southeast | 3756.621600 |
| 6 | 46 | 1 | 33.440 | 1 | 0 | southeast | 8240.589600 |
| 7 | 37 | 1 | 27.740 | 3 | 0 | northwest | 7281.505600 |
| 8 | 37 | 0 | 29.830 | 2 | 0 | northeast | 6406.410700 |
| 9 | 60 | 1 | 25.840 | 0 | 0 | northwest | 28923.136920 |
| 10 | 25 | 0 | 26.220 | 0 | 0 | northeast | 2721.320800 |
| 11 | 62 | 1 | 26.290 | 0 | 1 | southeast | 27808.725100 |
| 12 | 23 | 0 | 34.400 | 0 | 0 | southwest | 1826.843000 |
| 13 | 56 | 1 | 39.820 | 0 | 0 | southeast | 11890.717800 |
| 14 | 27 | 0 | 42.130 | 0 | 1 | southeast | 39611.757700 |
| 15 | 19 | 0 | 24.600 | 1 | 0 | southwest | 1837.237000 |
| 16 | 52 | 1 | 30.780 | 1 | 0 | northeast | 10797.336200 |
| 17 | 23 | 0 | 23.845 | 0 | 0 | northeast | 2395.171550 |
| 18 | 56 | 0 | 26.200 | 0 | 0 | southeast | 18077.105000 |

```
In [97]: from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

```
Out[97]: RandomForestClassifier()
RandomForestClassifier()
```

```
In [98]: 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 [101]: from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rf,param_grid=params, cv=2, scoring="accuracy")
grid_search.fit(x_train,y_train)
```

```
Out[101]: GridSearchCV()
estimator: RandomForestClassifier()
RandomForestClassifier()
```

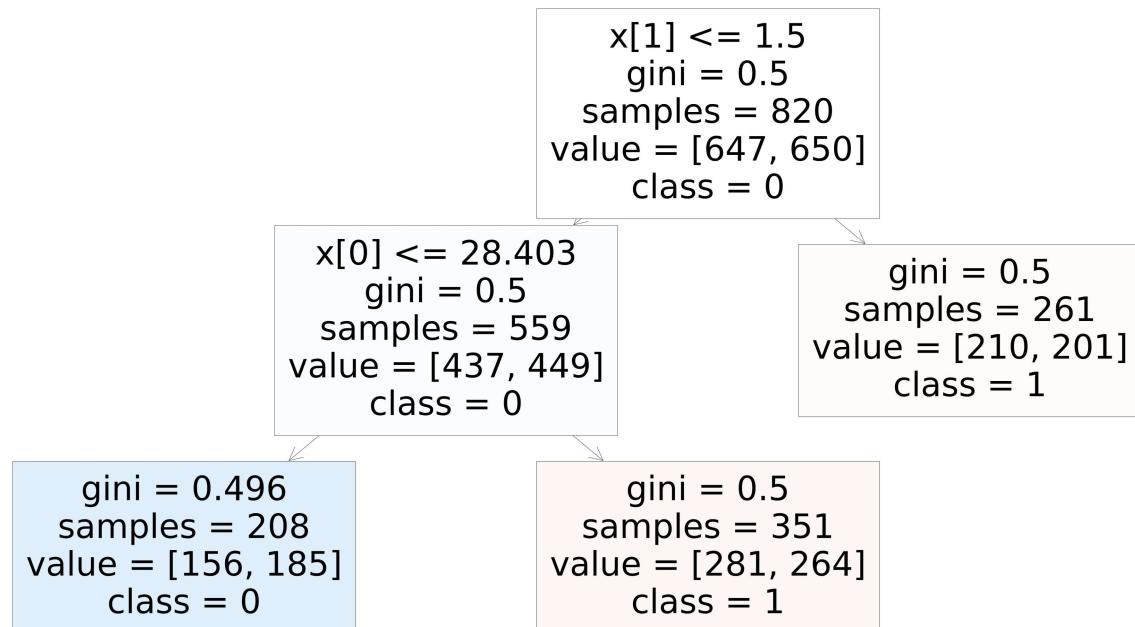
```
In [102]: grid_search.best_score_
```

```
Out[102]: 0.5242883163080903
```

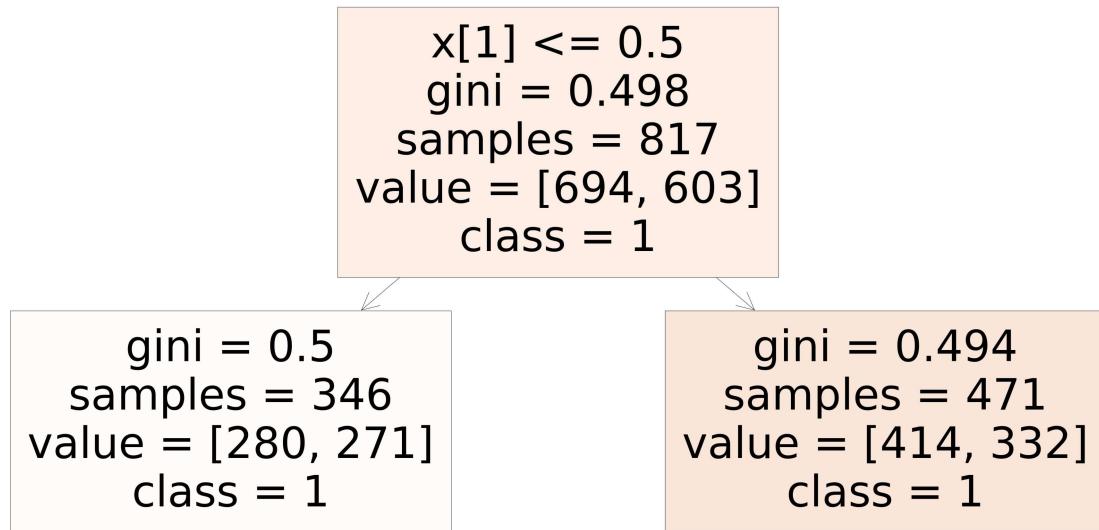
```
In [103]: rf_best=grid_search.best_estimator_
print(rf_best)
```

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

```
In [104]: from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[4], class_names=['1', '0'], filled=True);
```



```
In [105]: from sklearn.tree import plot_tree
plt.figure(figsize=(70,30))
plot_tree(rf_best.estimators_[6],class_names=["1","0"],filled=True);
```



```
In [106]: rf_best.feature_importances_
```

```
Out[106]: array([0.63952523, 0.36047477])
```

```
In [107]: rf=RandomForestClassifier(random_state=0)
```

```
In [108]: rf.fit(x_train,y_train)
```

```
Out[108]: RandomForestClassifier(random_state=0)
```

```
In [109]: score=rf.score(x_test,y_test)
print(score)
```

```
0.43902439024390244
```

Conclusion:

For the given insurance data set have performed linear,logistic,random forest and decision tree models of regression and classifications.

#and have conclude that the most accuracy is occurred in logistic regression,i.e 77percent #when compare to other regression models. #and concluded that "Logistic Regression" model is fits for the data.

```
In [ ]:
```