In [1]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
```

In [2]:

```
df=pd.read_csv(r"C:\Users\yasoda\Documents\202U1A05C1\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 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()
```

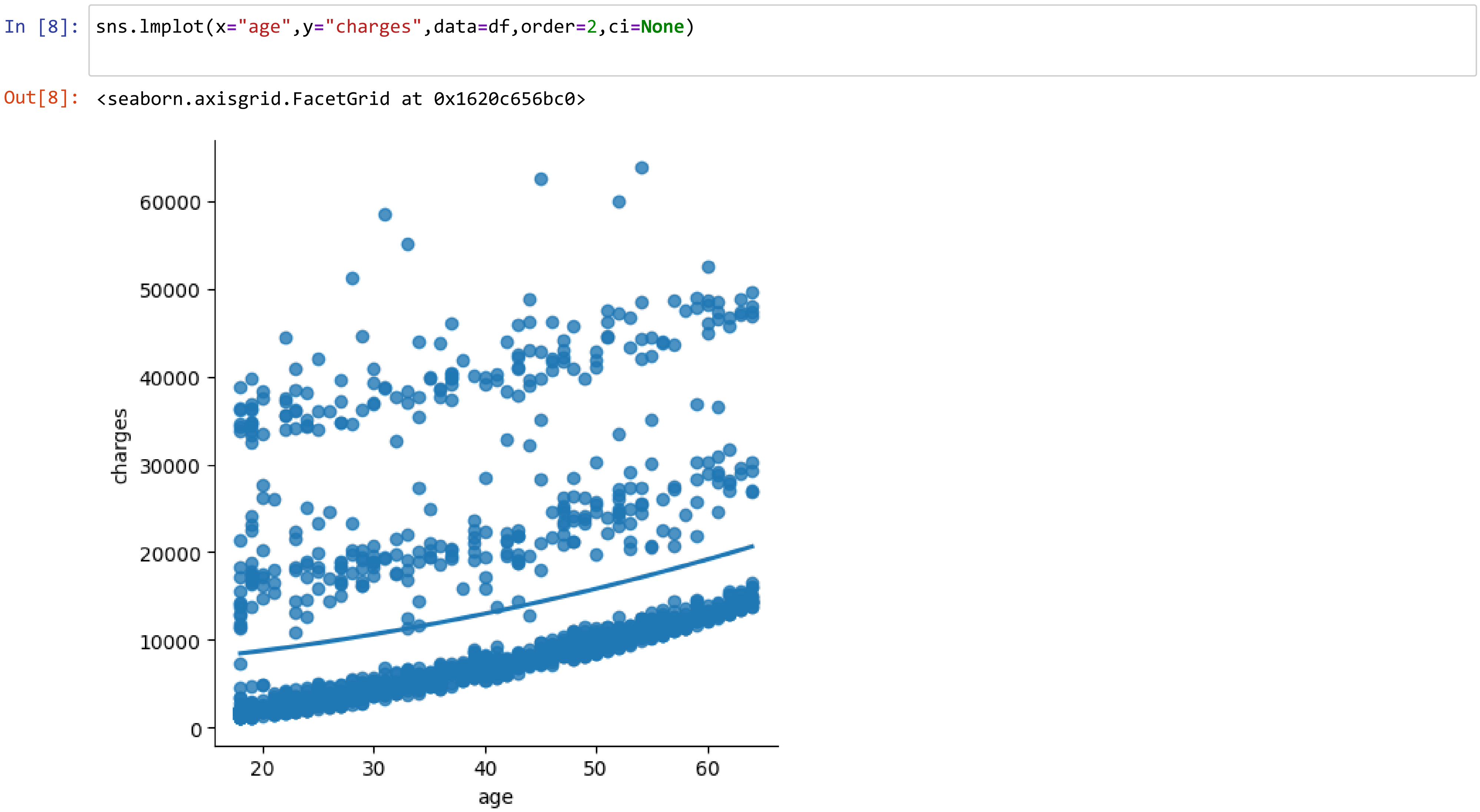
Out[6]:

| | age | bmi | children | charges |
|---|---|---|---|---|
| **count** | 1338.000000 | 1338.000000 | 1338.000000 | 1338.000000 |
| **mean** | 39.207025 | 30.663397 | 1.094918 | 13270.422265 |
| **std** | 14.049960 | 6.098187 | 1.205493 | 12110.011237 |
| **min** | 18.000000 | 15.960000 | 0.000000 | 1121.873900 |
| **25%** | 27.000000 | 26.296250 | 0.000000 | 4740.287150 |
| **50%** | 39.000000 | 30.400000 | 1.000000 | 9382.033000 |
| **75%** | 51.000000 | 34.693750 | 2.000000 | 16639.912515 |
| **max** | 64.000000 | 53.130000 | 5.000000 | 63770.428010 |

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]: sns.lmplot(x="age",y="charges",data=df,order=2,ci=None)

Out[8]: <seaborn.axisgrid.FacetGrid at 0x1620c656bc0>
```

**In the above scatter plot graph we can able to know that the aged peoples charges are low**

In [9]:
```
df.fillna(method='ffill',inplace=True)
```

In [10]:
```
x=np.array(df['age']).reshape(-1,1)
y=np.array(df['charges']).reshape(-1,1)
```

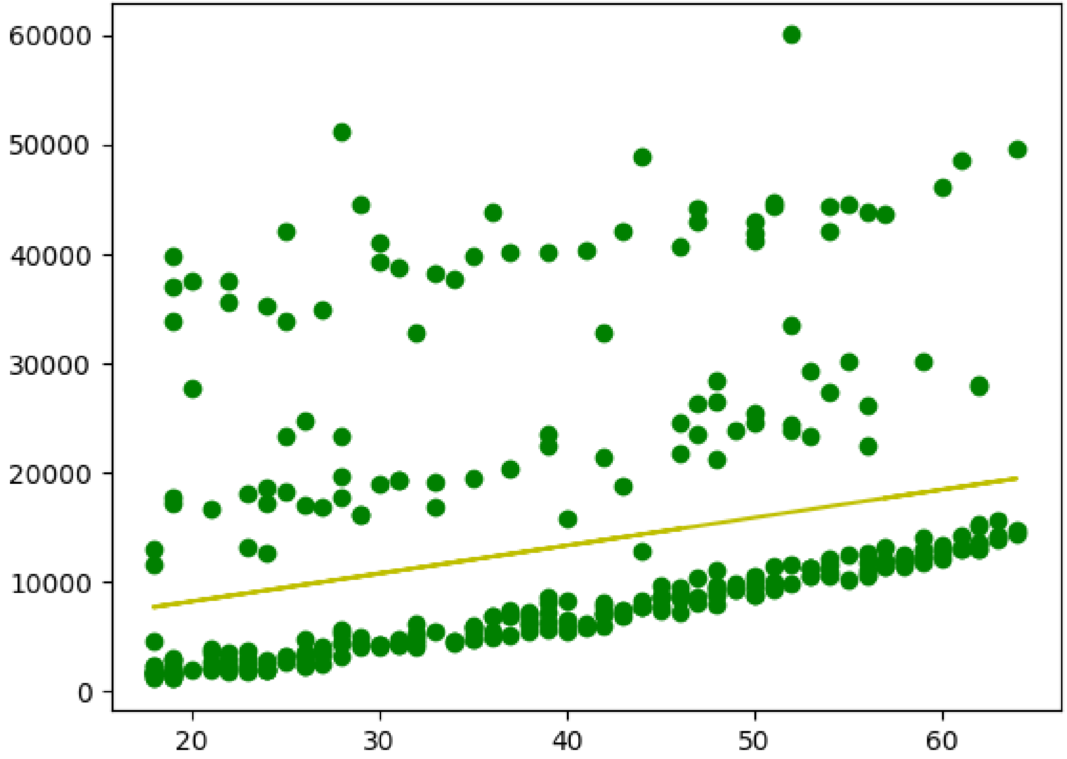
In [11]:
```
df.dropna(inplace=True)
```

In [12]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
regr=LinearRegression()
regr.fit(x_train,y_train)
print(regr.score(x_test,y_test))
```

```
0.08293882256151675
```

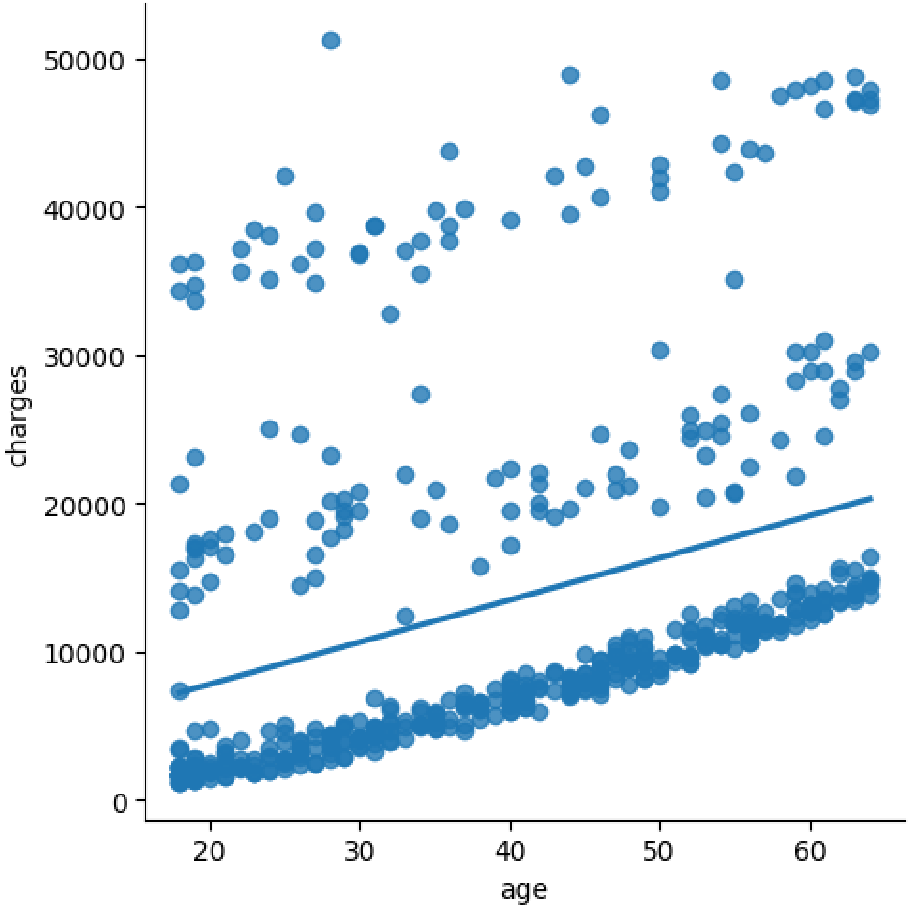
In [13]:

```
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='g')
plt.plot(x_test,y_pred,color='y')
plt.show()
```



In [14]:
```
df500=df[:][:500]
sns.lmplot(x="age",y="charges",data=df500,order=1,ci=None)
```

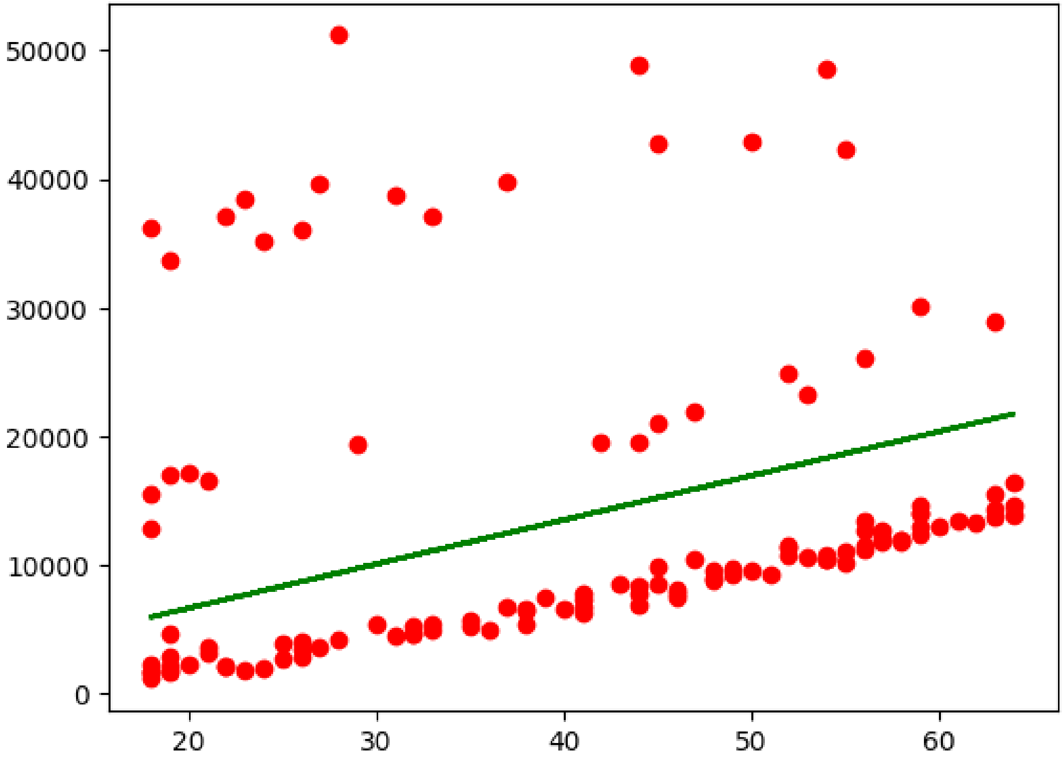
Out[14]:
```
<seaborn.axisgrid.FacetGrid at 0x1623d9fe830>
```



In [15]:

```
df500.fillna(method='ffill',inplace=True)
x=np.array(df500['age']).reshape(-1,1)
y=np.array(df500['charges']).reshape(-1,1)
df500.dropna(inplace=True)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
regr=LinearRegression()
regr.fit(x_train,y_train)
print("Regression:",regr.score(x_test,y_test))
y_pred=regr.predict(x_test)
plt.scatter(x_test,y_test,color='r')
plt.plot(x_test,y_pred,color='g')
plt.show()
```

```
Regression: -0.06442340865508056
```



In [16]:

```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
model=LinearRegression()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
r2=r2_score(y_test,y_pred)
print("R2 Score:",r2)
```

```
R2 Score: -0.06442340865508056
```

In [17]:

```
df.isnull().sum()
```

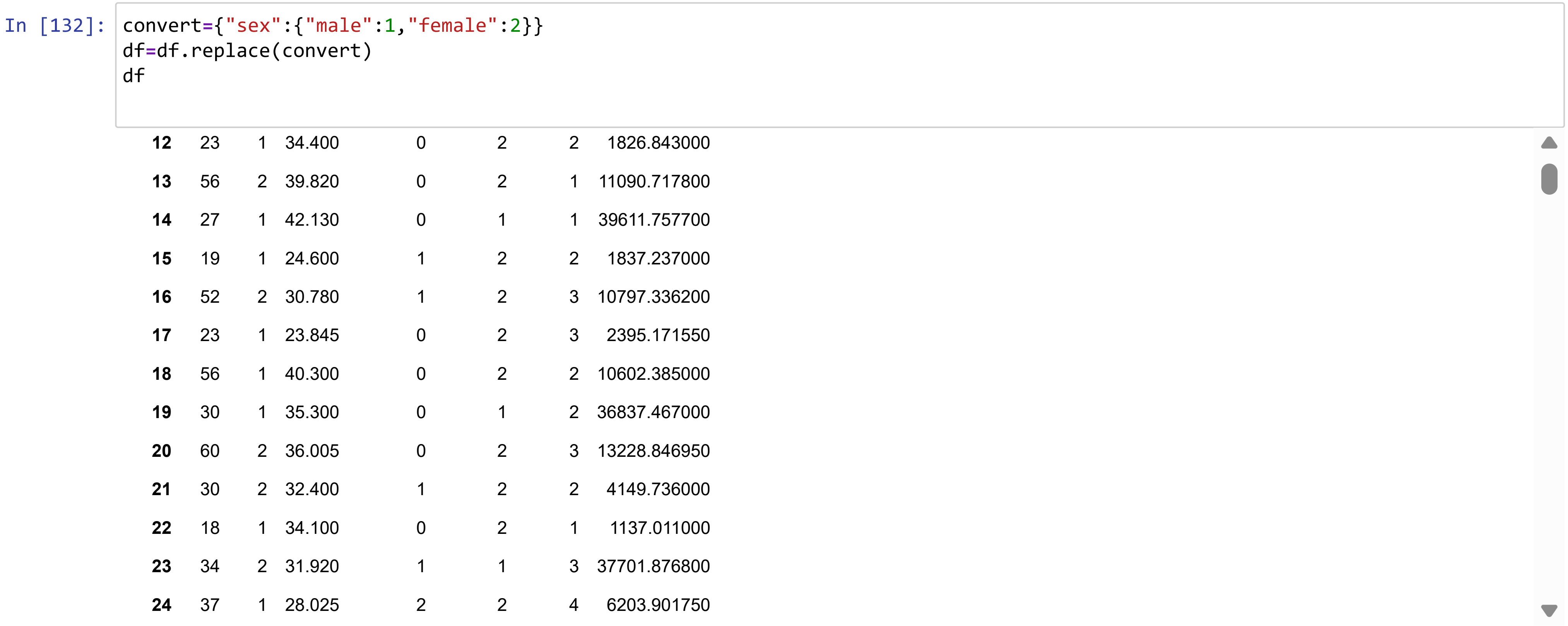
Out[17]:

```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

# Implementation of Ridge regression

In [157]:

```
from sklearn.linear_model import Ridge,RidgeCV,Lasso
from sklearn.preprocessing import StandardScaler
```



```
In [132]: convert={"sex":{"male":1,"female":2}}
          df=df.replace(convert)
          df
```

```
12   23   1   34.400    0    2    2    1826.843000
13   56   2   39.820    0    2    1   11090.717800
14   27   1   42.130    0    1    1   39611.757700
15   19   1   24.600    1    2    2    1837.237000
16   52   2   30.780    1    2    3   10797.336200
17   23   1   23.845    0    2    3    2395.171550
18   56   1   40.300    0    2    2   10602.385000
19   30   1   35.300    0    1    2   36837.467000
20   60   2   36.005    0    2    3   13228.846950
21   30   2   32.400    1    2    2    4149.736000
22   18   1   34.100    0    2    1    1137.011000
23   34   2   31.920    1    1    3   37701.876800
24   37   1   28.025    2    2    4    6203.901750
```

```
In [133]: convert={"smoker":{"yes":1,"no":2}}
          df=df.replace(convert)
          df
```

```
35  19  1  20.425  0  2  4   1625.433750
36  62  2  32.965  3  2  4  15612.193350
37  26  1  20.800  0  2  2   2302.300000
38  35  1  36.670  1  1  3  39774.276300
39  60  1  39.900  0  1  2  48173.361000
40  24  2  26.600  0  2  3   3046.062000
41  31  2  36.630  2  2  1   4949.758700
42  41  1  21.780  1  2  1   6272.477200
43  37  2  30.800  2  2  1   6313.759000
44  38  1  37.050  1  2  3   6079.671500
45  55  1  37.300  0  2  2  20630.283510
46  18  2  38.665  2  2  3   3393.356350
47  28  2  34.770  0  2  4   3556.922300
```

In [134]:

```
convert={"region":{"southeast":3,"southwest":4,"northeast":5,"northwest":6}}
df=df.replace(convert)
df
```

Out[134]:

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

In [136]:

```
features = df.columns[0:1]
target = df.columns[-1]
#X and y values
X = df[features].values
y = df[target].values
#splot
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=17)
print("The dimension of x_train is {}".format(X_train.shape))
print("The dimension of x_test is {}".format(X_test.shape))
#Scale features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
The dimension of x_train is (936, 1)
The dimension of x_test is (402, 1)
```

In [142]:

```
ridgeReg=Ridge(alpha=10)
ridgeReg.fit(X_train,y_train)
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 0.07446228994221393
The test score for ridge model is 0.10855133360950642
```

In [160]:

```
lr = LinearRegression()
#Fit model
lr.fit(X_train, y_train)
#predict
#prediction = lr.predict(X_test)
#actual
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 0.07447061146193878
The test score for lr model is 0.10891203216512224
```

In [ ]:

```
plt.figure(figsize=(10,10))
add plot for ridge regression
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,color='red',label=r'Ridge,$\alpha=grid$')
add plot for
```

In [149]:

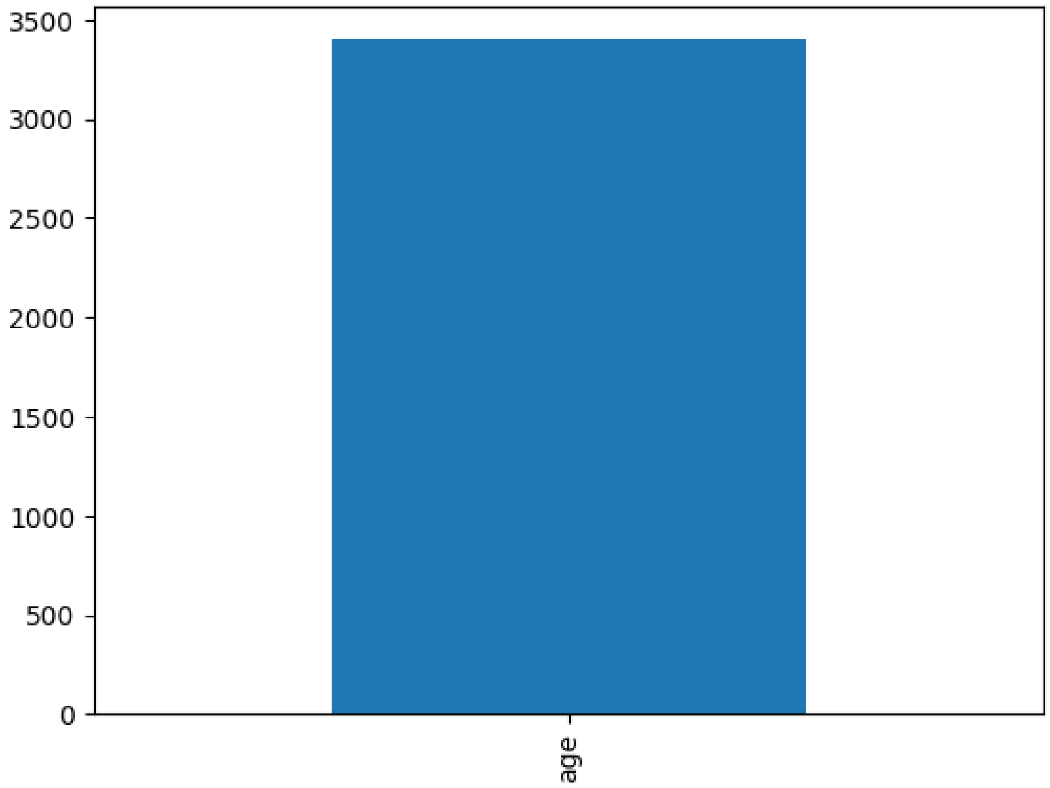
```
print("\nLasso Model: \n")
lasso = Lasso(alpha = 10)
lasso.fit(X_train,y_train)
train_score_ls =lasso.score(X_train,y_train)
test_score_ls =lasso.score(X_test,y_test)
print("The train score for ls model is {}".format(train_score_ls))
print("The test score for ls model is {}".format(test_score_ls))
```

```
Lasso Model: 

The train score for ls model is 0.07446997086306062
The test score for ls model is 0.10881427793326703
```

In [156]:

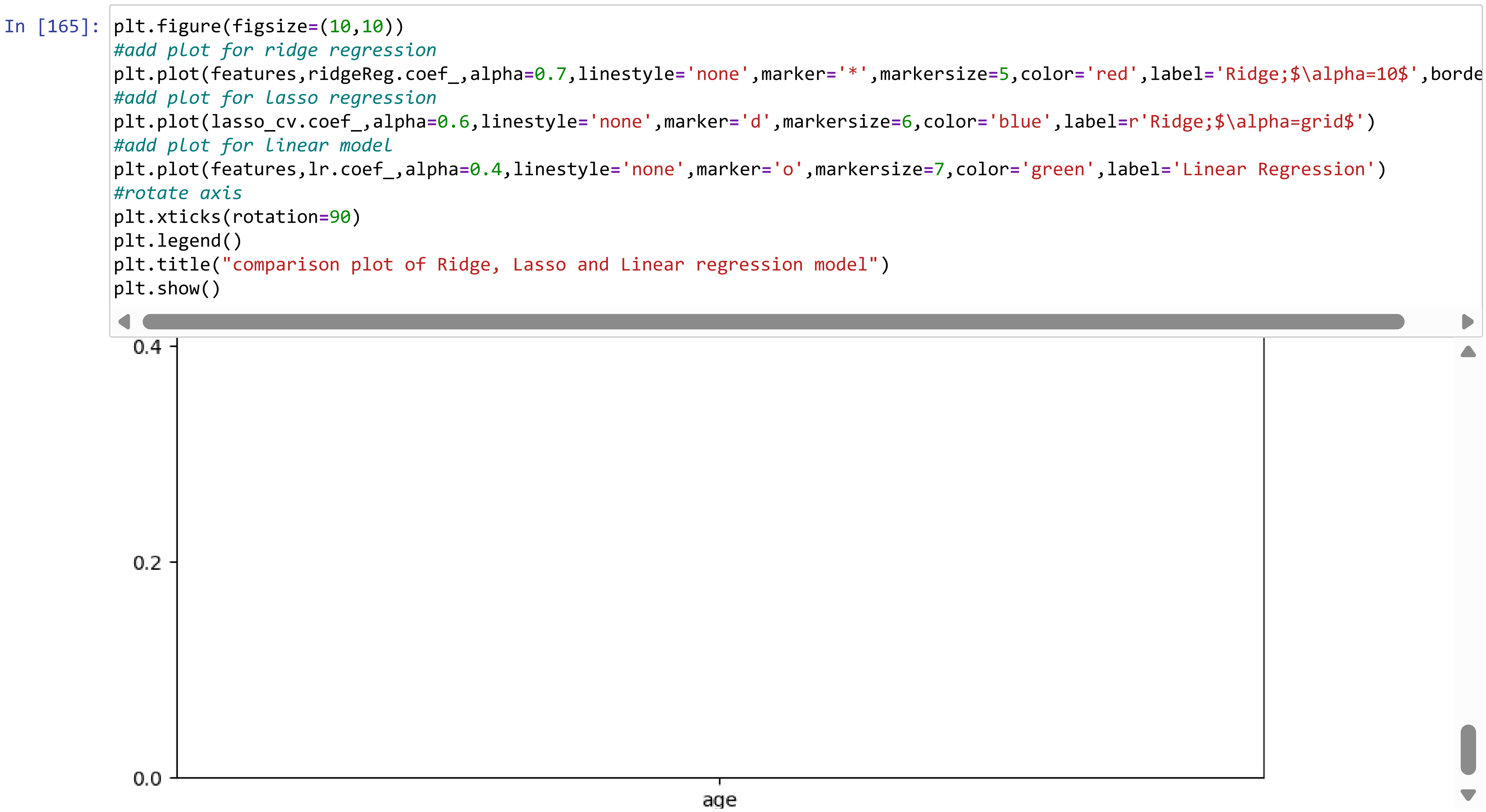
```
pd.Series(lasso.coef_, features).sort_values(ascending = True).plot(kind = "bar")
plt.show()
```



In [152]:

```
from sklearn.linear_model import LassoCV
lasso_cv=LassoCV(alphas=[0.0001,0.001,0.01,0.1,1,10],random_state=0).fit(X_train,y_train)
print(lasso_cv.score(X_train,y_train))
print(lasso_cv.score(X_test,y_test))
```

```
0.07446997086306062
0.10881427793326703
```



In [165]:

```
plt.figure(figsize=(10,10))
#add plot for ridge regression
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,color='red',label='Ridge;$\alpha=10$',borde
#add plot for lasso regression
plt.plot(lasso_cv.coef_,alpha=0.6,linestyle='none',marker='d',markersize=6,color='blue',label=r'Ridge;$\alpha=grid$')
#add plot for linear model
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,color='green',label='Linear Regression')
#rotate axis
plt.xticks(rotation=90)
plt.legend()
plt.title("comparison plot of Ridge, Lasso and Linear regression model")
plt.show()
```

In [169]:

```
from sklearn.linear_model import RidgeCV
#Ridge Cross validation
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))
```

```
0.07446228994221393
0.10855133360950775
```

# Elastic net regression

In [51]:

```
from sklearn.linear_model import ElasticNet
regr=ElasticNet()
regr.fit(x,y)
print(regr.coef_)
print(regr.intercept_)
```

```
[283.49314745]
[2117.95065707]
```

In [52]:

```
y_pred_elastic=regr.predict(x_train)
```

In [53]:

```
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print("Mean Squared Error on test set",mean_squared_error)
```

```
Mean Squared Error on test set 161136962.85931197
```

# Logistic regression

In [54]:

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [55]:

```
df=pd.read_csv(r"C:\Users\yasoda\Documents\202U1A05C1\insurance.csv")
df
```

Out[55]:

| | 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 [56]:

```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [57]:

```
print('This DataFrame has %d Rows and %d columns'%(df.shape))
```

```
This DataFrame has 1338 Rows and 7 columns
```

In [58]:

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

Out[58]:

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

In [59]:
```
convert={"sex":{"male":8,"female":9}}
df=df.replace(convert)
df
```

```
1305  24  9  27.720  0  2  southeast   2464.618800
1306  29  9  21.850  0  1  northeast  16115.304500
1307  32  8  28.120  4  1  northwest  21472.478800
1308  25  9  30.200  0  1  southwest  33900.653000
1309  41  8  32.200  2  2  southwest   6875.961000
1310  42  8  26.315  1  2  northwest   6940.909850
1311  33  9  26.695  0  2  northwest   4571.413050
1312  34  8  42.900  1  2  southwest   4536.259000
1313  19  9  34.700  2  1  southwest  36397.576000
1314  30  9  23.655  3  1  northwest  18765.875450
1315  18  8  28.310  1  2  northeast  11272.331390
1316  19  9  20.600  0  2  southwest   1731.677000
1317  18  8  53.130  0  2  southeast   1163.462700
```

In [60]:
```
features_matrix=df.iloc[:,0:4]
```

In [61]:
```
target_vector=df.iloc[:,-3]
```

In [170]:
```
print('The Features Matrix Has %d Rows And %d Column(s)'%(features_matrix.shape))
```

```
The Features Matrix Has 1338 Rows And 4 Column(s)
```

In [171]:
```
print('The Target Matrix Has %d Rows And %d Column(s)'%(np.array(target_vector).reshape(-1,1).shape))
```

```
The Target Matrix Has 1338 Rows And 1 Column(s)
```

```
In [64]: features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

```
In [177]: _scaling=1,class_weight=None,random_state=None,solver='lbfgs',max_iter=100,multi_class='auto',verbose=0,warm_start=False,n_jobs=
```

```
In [180]: Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_vector)
```

```
In [181]: observation=[[1,0,0.99539,-0.05889,]]
```

```
In [183]: 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 [2]
```

```
In [184]: 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: [1 2]
```

```
In [185]: print(" " "The model says she probability of the observation we passed belonging to class[0] Is %s" " "%(algorithm.predict_prob
```

```
 The model says she probability of the observation we passed belonging to class[0] Is 0.1942921563693959
```

```
In [71]: print()
```

```
In [186]: says the probabaility of the observation we passed belonging to class[1] Is %s" " "%(algorithm.predict_proba(observation)[0][0]
```

```
 The model says the probabaility of the observation we passed belonging to class[1] Is 0.1942921563693959
```

In [73]:

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

In [74]:

```
lerg=LogisticRegression()
lerg.fit(x,y)
print(lerg.score(x,y))
```

```
0.7952167414050823

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

# Decision tree regression

In [75]:

```
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 [76]:
```
df=pd.read_csv(r"C:\Users\yasoda\Documents\202U1A05C1\insurance.csv")
df
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **1080** | 18 | male | 21.780 | 2 | no | southeast | 11884.048580 |
| **1081** | 32 | male | 27.835 | 1 | no | northwest | 4454.402650 |
| **1082** | 38 | male | 19.950 | 1 | no | northwest | 5855.902500 |
| **1083** | 32 | male | 31.500 | 1 | no | southwest | 4076.497000 |
| **1084** | 62 | female | 30.495 | 2 | no | northwest | 15019.760050 |
| **1085** | 39 | female | 18.300 | 5 | yes | southwest | 19023.260000 |
| **1086** | 55 | male | 28.975 | 0 | no | northeast | 10796.350250 |
| **1087** | 57 | male | 31.540 | 0 | no | northwest | 11353.227600 |
| **1088** | 52 | male | 47.740 | 1 | no | southeast | 9748.910600 |
| **1089** | 56 | male | 22.100 | 0 | no | southwest | 10577.087000 |
| **1090** | 47 | male | 36.190 | 0 | yes | southeast | 41676.081100 |
| **1091** | 55 | female | 29.830 | 0 | no | northeast | 11286.538700 |
| **1092** | 23 | male | 32.700 | 3 | no | southwest | 3591.480000 |

In [77]:
```
df['region'].value_counts()
```

Out[77]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [78]:

```
df['bmi'].value_counts()
```

```
28.380     1
24.090     1
25.100     1
34.300     1
43.400     1
39.200     1
35.700     1
26.070     1
39.425     1
40.480     1
38.900     1
47.410     1
35.435     1
46.700     1
46.200     1
23.800     1
44.770     1
32.120     1
30.970     1
Name: count, dtype: int64
```

In [79]:

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

```
 3   33   1   22.705        0     no   northwest   21984.470610
 4   32   1   28.880        0     no   northwest    3866.855200
 5   31   0   25.740        0     no   southeast    3756.621600
 6   46   0   33.440        1     no   southeast    8240.589600
 7   37   0   27.740        3     no   northwest    7281.505600
 8   37   1   29.830        2     no   northeast    6406.410700
 9   60   0   25.840        0     no   northwest   28923.136920
10   25   1   26.220        0     no   northeast    2721.320800
11   62   0   26.290        0    yes   southeast   27808.725100
12   23   1   34.400        0     no   southwest    1826.843000
13   56   0   39.820        0     no   southeast   11090.717800
14   27   1   42.130        0    yes   southeast   39611.757700
15   19   1   24.600        1     no   southwest    1837.237000
```

In [80]:

```
x=["bmi","children"]
y=["yes","no"]
all_inputs=df[x]
all_classes=df["sex"]
```

In [187]:

```
(x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,test_size=0.03)
```

In [188]:

```
clf=DecisionTreeClassifier(random_state=0)
```

In [189]:
```
clf.fit(x_train,y_train)
```

Out[189]:
```
▾         DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

In [190]:
```
score=clf.score(x_test,y_test)
print(score)
```

```
0.4878048780487805
```

# Random forest

In [107]:
```
import pandas as pd
import numpy as ny
import matplotlib.pyplot as plt,seaborn as sns
```

In [108]:
```
df=pd.read_csv(r"C:\Users\yasoda\Documents\202U1A05C1\insurance.csv")
df
```

```
446  60    male  29.640  0  no  northeast  12730.999600
447  56  female  25.650  0  no  northwest  11454.021500
448  40  female  29.600  0  no  southwest   5910.944000
449  35    male  38.600  1  no  southwest   4762.329000
450  39    male  29.600  4  no  southwest   7512.267000
451  30    male  24.130  1  no  northwest   4032.240700
452  24    male  23.400  0  no  southwest   1969.614000
453  20    male  29.735  0  no  northwest   1769.531650
454  32    male  46.530  2  no  southeast   4686.388700
455  59    male  37.400  0  no  southwest  21797.000400
456  55  female  30.140  2  no  southeast  11881.969600
457  57  female  30.495  0  no  northwest  11840.775050
458  56    male  39.600  0  no  southwest  10601.412000
```

In [109]:

```
df['charges'].value_counts()
```

```
4738.268200     1
7512.267000     1
11840.775050    1
11881.969600    1
21797.000400    1
4686.388700     1
1769.531650     1
1969.614000     1
4032.240700     1
4762.329000     1
37079.372000    1
5910.944000     1
11454.021500    1
12730.999600    1
7345.084000     1
26109.329050    1
28287.897660    1
1149.395900     1
29141.360300    1
Name: count, dtype: int64
```

In [110]:

```
m={"region":{"southeast":1,"southwest":2,"northeast":3,"northwest":4}}
df=df.replace(m)
print(df)
```

```
370     61  female  21.090         0     no        4  13415.038100
371     57  female  22.230         0     no        3  12029.286700
372     42  female  33.155         1     no        3   7639.417450
373     26    male  32.900         2    yes        2  36085.219000
374     20    male  33.330         0     no        1   1391.528700
375     23  female  28.310         0    yes        4  18033.967900
376     39  female  24.890         3    yes        3  21659.930100
377     24    male  40.150         0    yes        1  38126.246500
378     64  female  30.115         3     no        4  16455.707850
379     62    male  31.460         1     no        1  27000.984730
380     27  female  17.955         2    yes        3  15006.579450
381     55    male  30.685         0    yes        3  42303.692150
382     55    male  33.000         0     no        1  20781.488920
383     35  female  43.340         2     no        1   5846.917600
384     44    male  22.135         2     no        3   8302.535650
385     19    male  34.400         0     no        2   1261.859000
386     58  female  39.050         0     no        1  11856.411500
387     50    male  25.365         2     no        4  30284.642940
388     26  female  22.610         0     no        4   3176.815900
389     24  female  30.210         3     no        4   4618.079900
```

In [111]:

```
df.shape
```

Out[111]:

```
(1338, 7)
```

In [191]:

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[191]:

```
▾ RandomForestClassifier

RandomForestClassifier()
```

In [194]:

```
rf=RandomForestClassifier()
```

In [196]:

```
params={'max_depth':[2,3,5,10,20],'min_samples_leaf':[5,10,20,50,100,200],'n_estimators':[10,25,30,50,100,200]}
```

In [ ]:

```
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)
```

In [203]:

```
grid_search.best_score_
```

Out[203]:

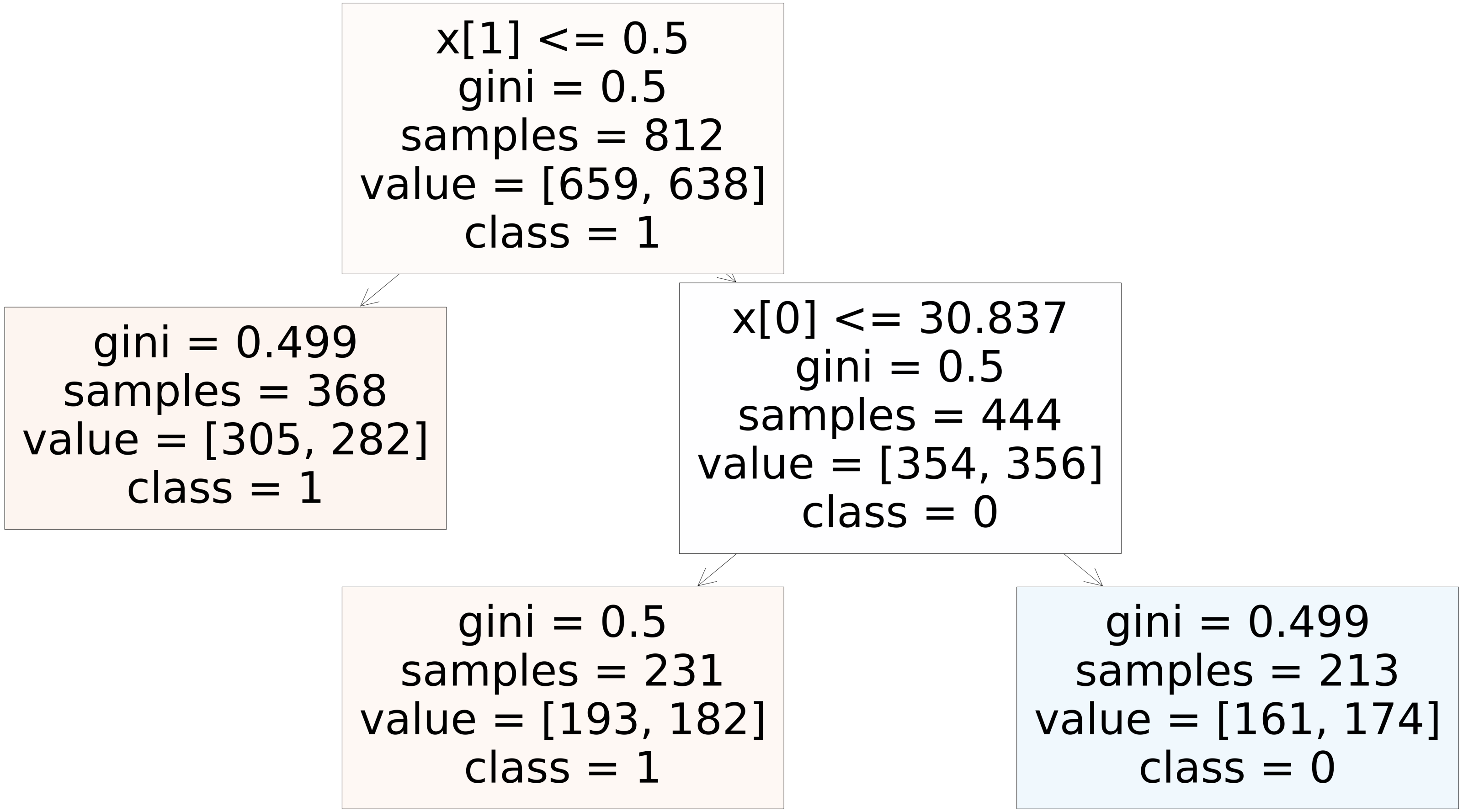
```
0.5250444653693241
```

In [204]:

```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=20, min_samples_leaf=200, n_estimators=25)
```

In [205]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[4],class_names=['1','0'],filled=True);
```



In [210]:
```
rf_best.feature_importances_
```

Out[210]:
```
array([0.64289857, 0.35710143])
```

In [209]:

```
score=rfc.score(x_test,y_test)
print(score)
```

```
0.439024390243902444
```

# Conclusion:In a given dataset we have performed Linear,Logistic,Decisiontree,Randomforest regression models have concluded the most accuracy it is occured in logistic regression.when we compare other regression models Logistic Regression suits best for the given data