Below is the code for a predictive modelling project. This model predicts medical insurance costs based on someone's age, location, number of children, smoking status, obesity levels etc.,

I was inspired to rerun this model, try to break the code, and see how this model runs under different assumptions. Mainly I wanted to check to see if this model accounts for aging and smoking status. And it does. It's a great introduction to machine learning and a nice model. Smoking status and age are two factors that can increase medical insurance costs significantly. My results, compared to the original results in the article, account for those two factors. Overall I loved this model.

Please see here: https://medium.com/analytics-vidhya/predicting-medical-insurance-costs-machine-learning-e1e4e7c4e8ed (https://medium.com/analytics-vidhya/predicting-medical-insurance-costs-machine-learning-e1e4e7c4e8ed)

Supervised machine learning project

Data set from Kaggle.com

In [1]:

```
import pandas as pd
import numpy as np
import sklearn as sk
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import cross_val_score, KFold
from sklearn import model_selection
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, mean_absolute_error
insurance = pd.read_csv("insurance.csv")
insurance.head()
```

Out[1]:

	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 [2]:

```
insurance=pd.read_csv("insurance.csv")
insurance.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 [3]:

```
# Replacing string values to numbers
insurance['sex'] = insurance['sex'].apply({'male':0,      'female':1}.get)
insurance['smoker'] = insurance['smoker'].apply({'yes':1, 'no':0}.get)
insurance['region'] = insurance['region'].apply({'southwest':1, 'southeast':2, 'northwe
st':3, 'northeast':4}.get)
```

In [4]:

```
insurance.head()
```

Out[4]:

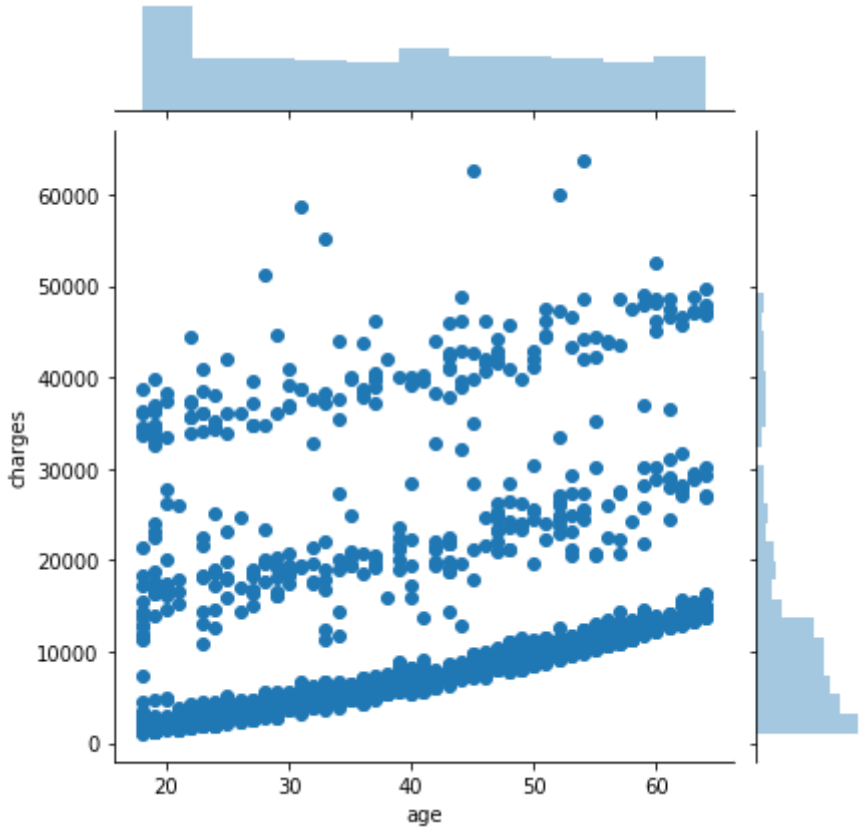
	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	1	16884.92400
1	18	0	33.770	1	0	2	1725.55230
2	28	0	33.000	3	0	2	4449.46200
3	33	0	22.705	0	0	3	21984.47061
4	32	0	28.880	0	0	3	3866.85520

In [5]:

```
import seaborn as sns
# Correlation betweeen 'charges' and 'age'
sns.jointplot(x=insurance['age'],y=insurance['charges'])
```

Out[5]:

```
<seaborn.axisgrid.JointGrid at 0x232ab7875c8>
```



In [6]:

```
# Code to distinguish obese from non-obese - 0 will be nonsmoker and 1 is smoker
def map_obese(column):
    mapped=[]
    for row in column:
        if row>30:
            mapped.append(1)
        else:
            mapped.append(0)
    return mapped
insurance["obese"]=map_obese(insurance["bmi"])
```

In [7]:

```
# Code to distinguish smoker vs. non-smoker - 0 will be not obese and 1 obese
def map_smoking(column):
    mapped=[]
    for row in column:
        if row=="yes":
            mapped.append(1)
        else:
            mapped.append(0)
    return mapped
insurance["smoker_norm"]=map_smoking(insurance["smoker"])
nonnum_cols=[col for col in insurance.select_dtypes(include=["object"])]
```

In [8]:

```
insurance.head(5)
```

Out[8]:

	age	sex	bmi	children	smoker	region	charges	obese	smoker_norm
0	19	1	27.900	0	1	1	16884.92400	0	0
1	18	0	33.770	1	0	2	1725.55230	1	0
2	28	0	33.000	3	0	2	4449.46200	1	0
3	33	0	22.705	0	0	3	21984.47061	0	0
4	32	0	28.880	0	0	3	3866.85520	0	0

In [9]:

```
# features
X = insurance[['age', 'sex', 'bmi', 'children','smoker','region']]
# predicted variable
y = insurance['charges']
```

In [10]:

```
# Train and test prediction model
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4)
```

In [11]:

```
len(X_test) # 402
len(X_train) # 936
len(insurance) # 1338
```

Out[11]:

```
1338
```

In [12]:

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train,y_train)
```

Out[12]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=F
alse)
```

In [13]:

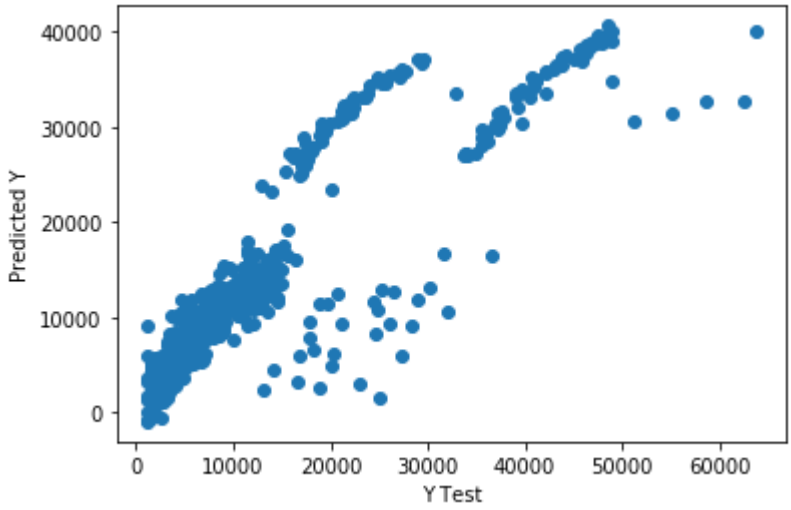
```
predictions = model.predict(X_test)
```

In [14]:

```
import matplotlib.pyplot as plt
plt.scatter(y_test,predictions)
plt.xlabel('Y Test')
plt.ylabel('Predicted Y')
```

Out[14]:

```
Text(0, 0.5, 'Predicted Y')
```



In [15]:

```
# Prediction of charges for new customer - Named "Random"
data = {'age' : 45,
        'sex' : 0,
        'bmi' : 45.50,
        'children' : 2,
        'smoker' : 0,
        'region' : 3}
index = [1]
random_df = pd.DataFrame(data,index)
random_df
```

Out[15]:

	age	sex	bmi	children	smoker	region
1	45	0	45.5	2	0	3

In [16]:

```
prediction_random = model.predict(random_df)
print("Medical Insurance cost for Random is : ",prediction_random)
```

```
Medical Insurance cost for Random is :  [15270.2437156]
```

In [ ]:

```

```