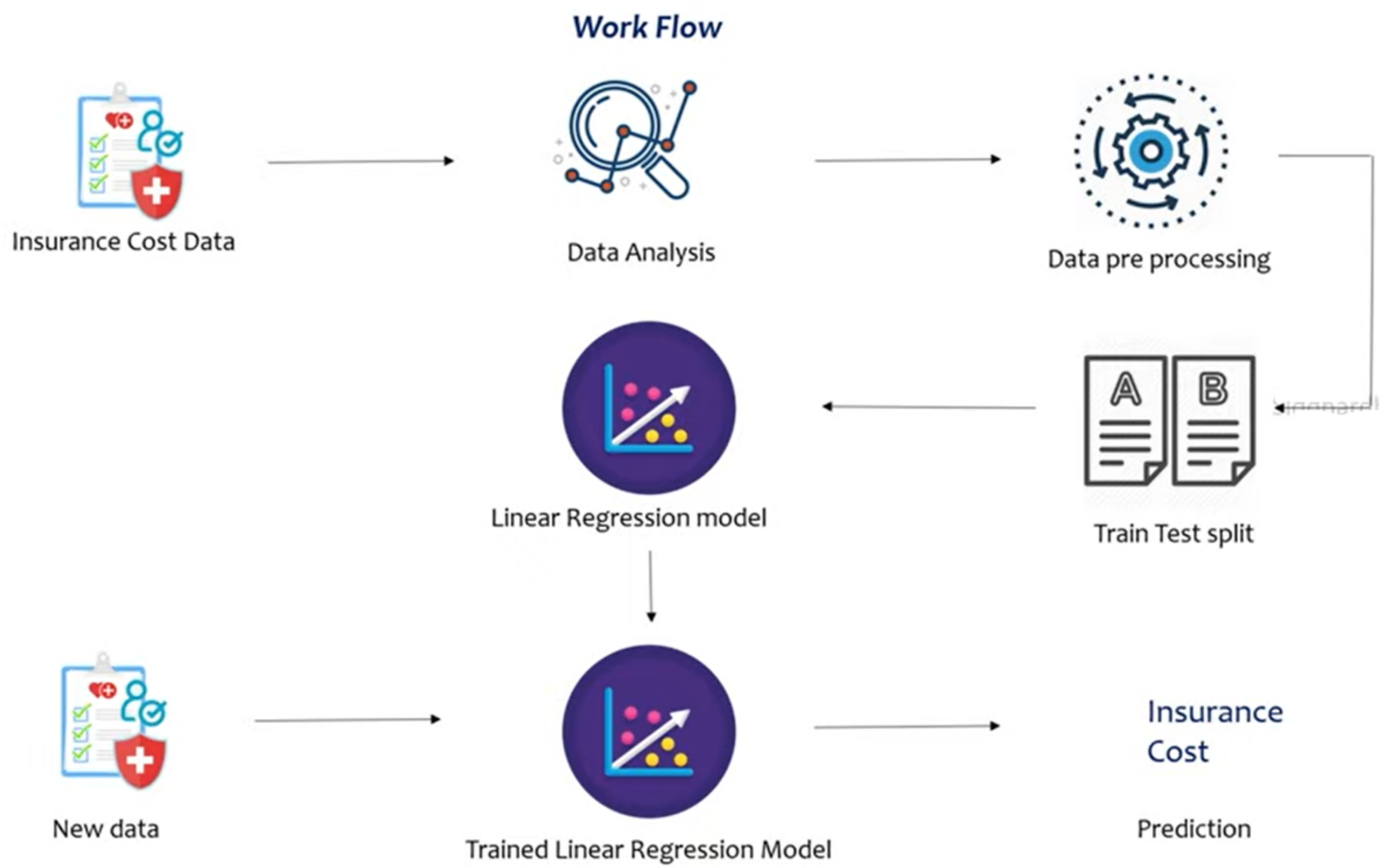


Work Flow::

- First Step is to collect data for this project
- Second step is data analysis: We analyse this data for better insights
- Next step is Data Pre-processing: So once we have a data we cannot feed it directly to our model,We have to do some pre-processing
- Next step is to split the data into training data and testing data....In Machine learning we train the model using training data and we test or evaluate the model using the testing data
- Once after splitting the data into training and testing data, we feed the training data into our model....Here we are using linear regression model

- Now we will get a trained linear regression model....so once we have that,we can feed new data....once we feed new data,this model can predict what will be the insurance cost

Importing Dependencies::

In [2]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt # for plots
import seaborn as sns # for plots
from sklearn.model_selection import train_test_split  # It help us to split the data into training and testing d
from sklearn.linear_model import LinearRegression
from sklearn import metrics  # This metrices are used to perform some evalaution on our model
```

Data Collection & Analysis::

In [14]:

```
# Loading the data from csv file to a Pandas DataFrame
insurance_dataset = pd.read_csv('insurance.csv')
```

In [15]:

```
# first 5 rows of the dataframe
insurance_dataset.head()
```

Out[15]:

	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

This dataset belong to united states and charges are in dollars

In [16]:

```
# number of rows and columns
insurance_dataset.shape
```

Out[16]:

```
(1338, 7)
```

In [17]:

```
# getting some informations about the dataset
insurance_dataset.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 this dataset ,Totally we have 3 categorical values

- Sex
- Smoker
- Region

In [18]:
```
# checking for missing values
insurance_dataset.isnull().sum()
```

Out[18]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

Data Analysis::

In [19]:
```
# statistical Measures of the dataset
insurance_dataset.describe()
```

Out[19]:

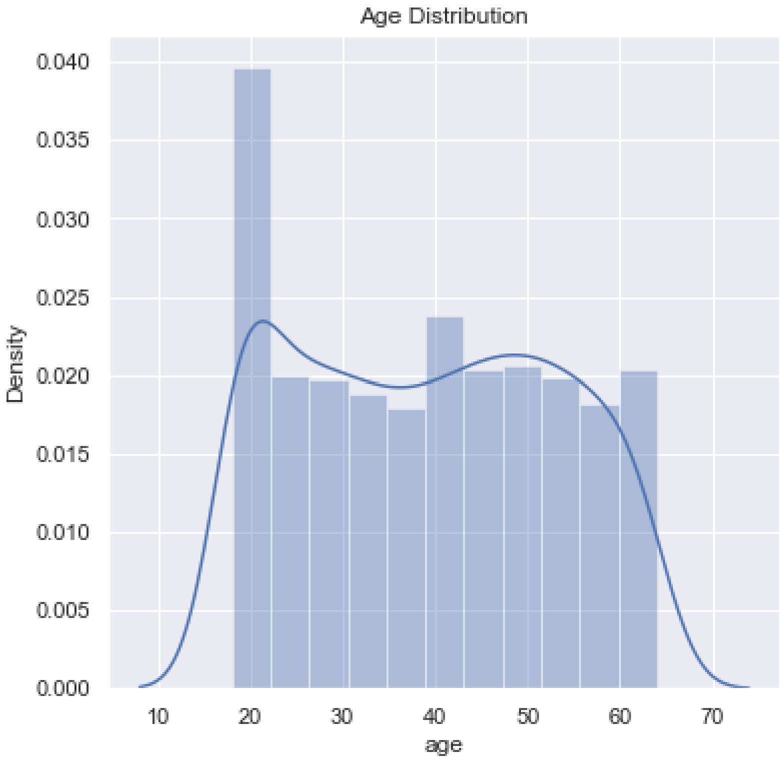
	age	bmi	children	charges
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.663397	1.094918	13270.422265
std	14.049960	6.098187	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.296250	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000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

Note :This statistical measure is only for non-categorical values...not for categorical values ie,sex,smoker,region

Now will find the distribution of dataset-----We do it column by column

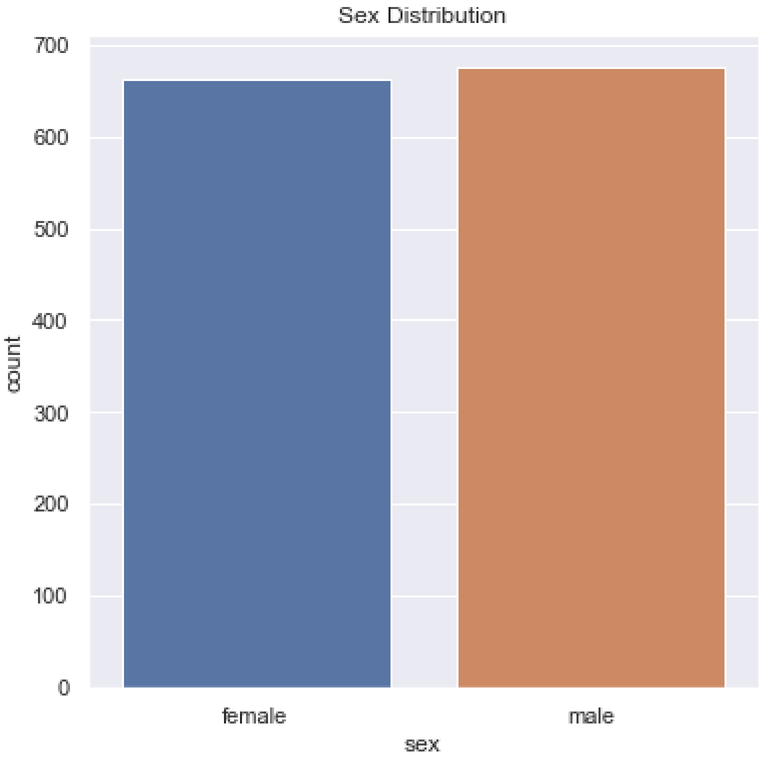
```
In [20]: # distribution of age value
         sns.set()
         plt.figure(figsize=(6,6))
         sns.distplot(insurance_dataset['age'])
         plt.title('Age Distribution')
         plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a depre
cated function and will be removed in a future version. Please adapt your code to use either `displot` (a figu
re-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Here we have maximum distribution at the age group of 20

In [23]:
```
# Gender column
sns.set()
plt.figure(figsize=(6,6))
sns.countplot(x='sex', data=insurance_dataset)
# countplot() method is used to Show the counts of observations in each categorical bin using bars.
plt.title('Sex Distribution')
plt.show()
```

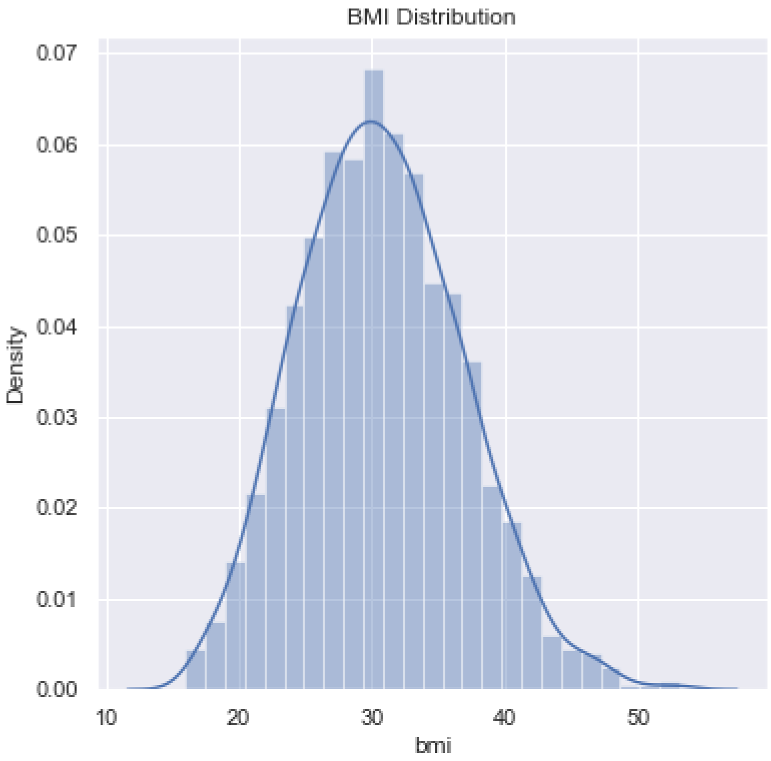


In [24]:
```
insurance_dataset['sex'].value_counts()
```

Out[24]:
```
male      676
female    662
Name: sex, dtype: int64
```

In [25]:
```
# bmi distribution
plt.figure(figsize=(6,6))
sns.distplot(insurance_dataset['bmi'])
plt.title('BMI Distribution')
plt.show()
```

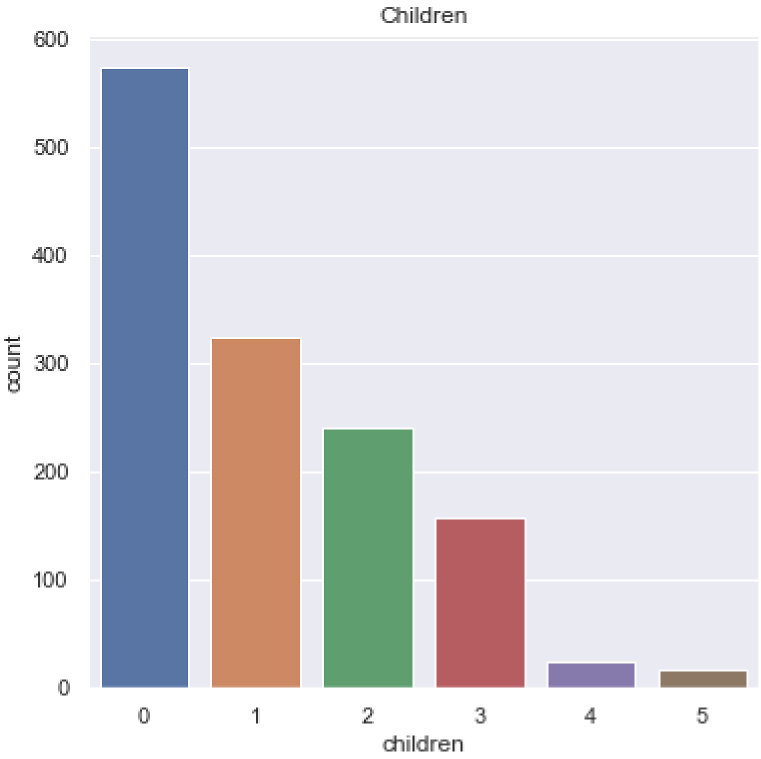
```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a depre
cated function and will be removed in a future version. Please adapt your code to use either `displot` (a figu
re-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



For a person Normal BMI Range --> 18.5 to 24.9

In [26]:

```
# children column
plt.figure(figsize=(6,6))
sns.countplot(x='children', data=insurance_dataset)
plt.title('Children')
plt.show()
```



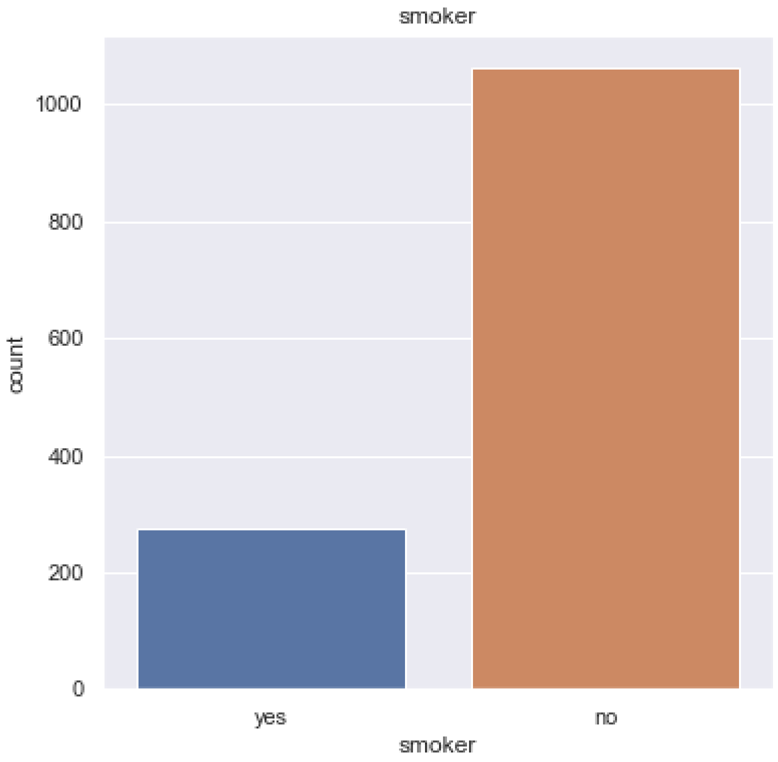
In [27]:

```
insurance_dataset['children'].value_counts()
```

Out[27]:

```
0    574
1    324
2    240
3    157
4     25
5     18
Name: children, dtype: int64
```

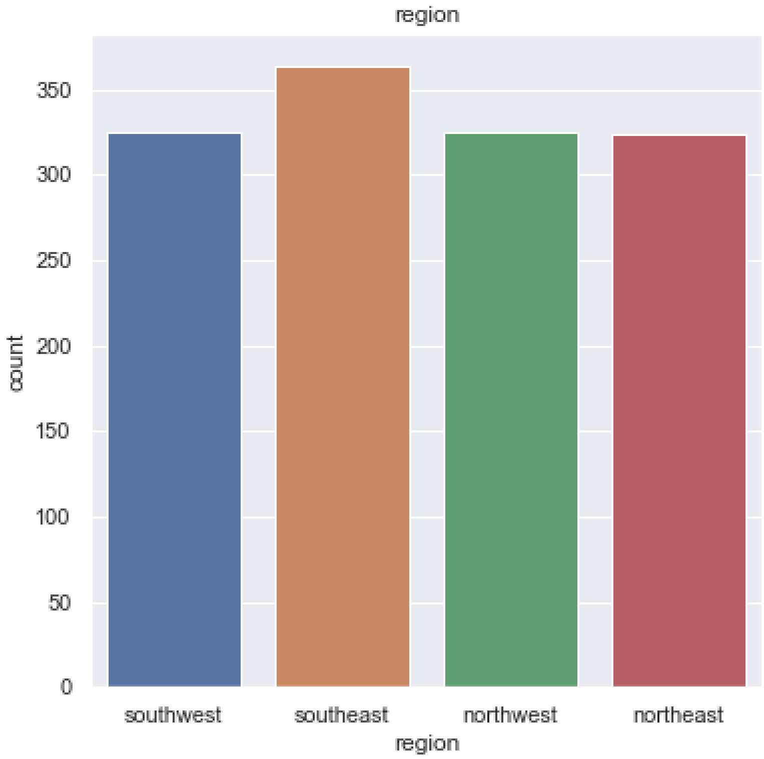
In [28]:
```
# smoker column
plt.figure(figsize=(6,6))
sns.countplot(x='smoker', data=insurance_dataset)
plt.title('smoker')
plt.show()
```



In [29]:
```
insurance_dataset['smoker'].value_counts()
```

Out[29]:
```
no     1064
yes     274
Name: smoker, dtype: int64
```

In [30]:
```
# region column
plt.figure(figsize=(6,6))
sns.countplot(x='region', data=insurance_dataset)
plt.title('region')
plt.show()
```



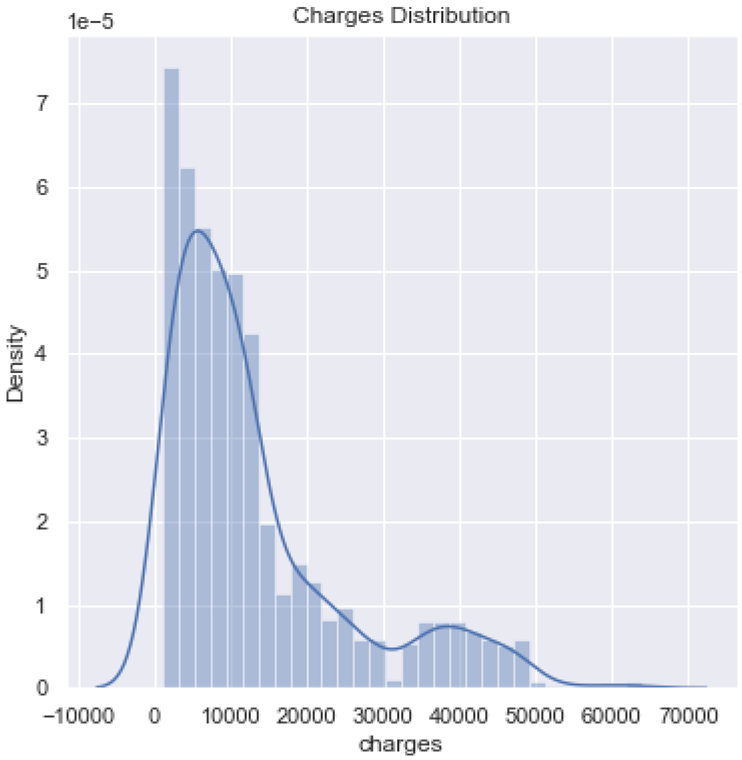
In [31]:
```
insurance_dataset['region'].value_counts()
```

Out[31]:
```
southeast    364
southwest    325
northwest    325
northeast    324
Name: region, dtype: int64
```

In [32]:

```
# distribution of charges value
plt.figure(figsize=(6,6))
sns.distplot(insurance_dataset['charges'])
plt.title('Charges Distribution')
plt.show()
```

```
C:\ProgramData\Anaconda3\lib\site-packages\seaborn\distributions.py:2557: FutureWarning: `distplot` is a depre
cated function and will be removed in a future version. Please adapt your code to use either `displot` (a figu
re-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



Data Pre-Processing::

- Encoding the categorical features

In [33]:

```
# encoding sex column
insurance_dataset.replace({'sex':{'male':0,'female':1}}, inplace=True)

3 # encoding 'smoker' column
insurance_dataset.replace({'smoker':{'yes':0,'no':1}}, inplace=True)

# encoding 'region' column
insurance_dataset.replace({'region':{'southeast':0,'southwest':1,'northeast':2,'northwest':3}}, inplace=True)
```

Splitting the Features and Target

In [34]:

```
X = insurance_dataset.drop(columns='charges', axis=1)
Y = insurance_dataset['charges']
```

In [35]:

```
print(X)
```

```
      age  sex     bmi  children  smoker  region
0      19    1  27.900         0       0       1
1      18    0  33.770         1       1       0
2      28    0  33.000         3       1       0
3      33    0  22.705         0       1       3
4      32    0  28.880         0       1       3
...   ...  ...     ...       ...     ...     ...
1333   50    0  30.970         3       1       3
1334   18    1  31.920         0       1       2
1335   18    1  36.850         0       1       0
1336   21    1  25.800         0       1       1
1337   61    1  29.070         0       0       3

[1338 rows x 6 columns]
```

In [36]:

```
print(Y)
```

```
0       16884.92400
1        1725.55230
2        4449.46200
3       21984.47061
4        3866.85520
           ...
1333    10600.54830
1334     2205.98080
1335     1629.83350
1336     2007.94500
1337    29141.36030
Name: charges, Length: 1338, dtype: float64
```

Splitting the data into Training data & Testing Data

In [37]:

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2, random_state=2)
```

Explanation:

- Here we have created 4 arrays:- X_train, X_test, Y_train, Y_test

- What happens here is : This X will be splitted into two arrays ie, X-train and X-test......And the Y ie,Charges will be splitted into Y-train and Y-test

- On the right hand side ,we have mentioned X,Y which means X and Y are used for splitting

- Test-size is 0.2 which means 20% of data is used for testing and 80% of data is used for training

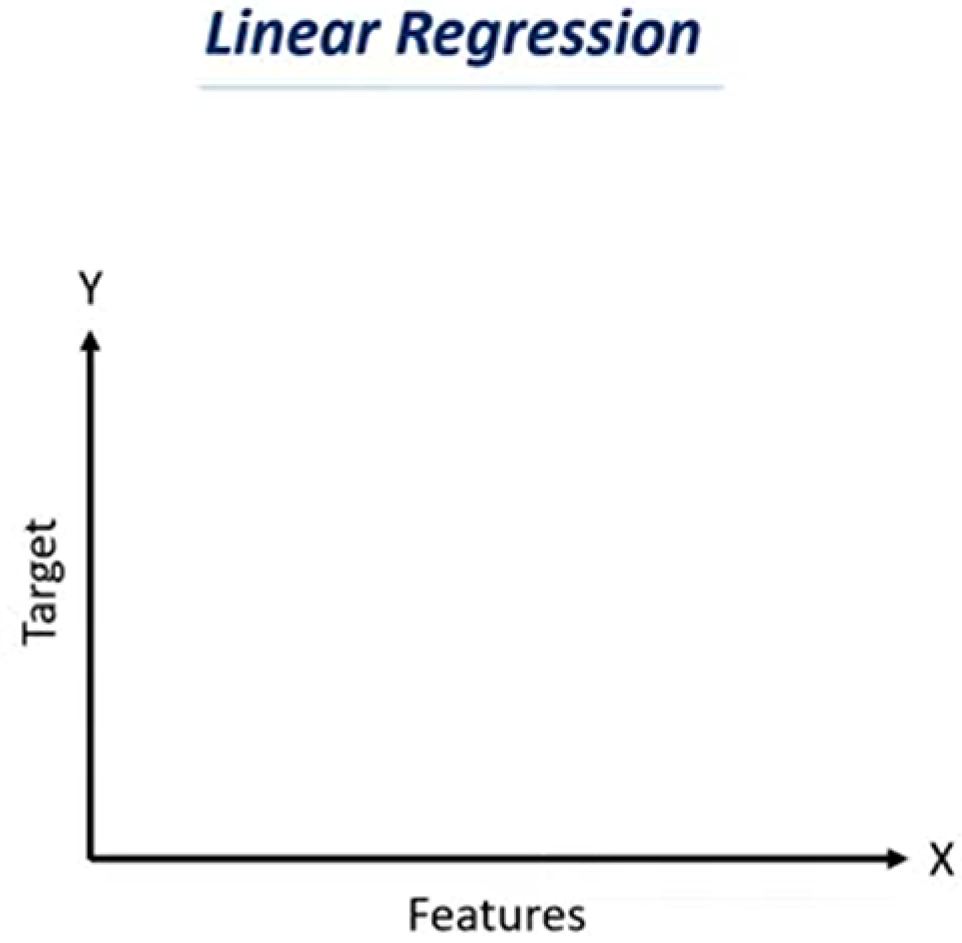
- Finally we have random state parameter,if you are mentioning the random state =2,then both of our data will be splitted in the same manner

Now lets check the shape of data after splitting

```
print(X.shape, X_train.shape, X_test.shape)
```

Model Training:

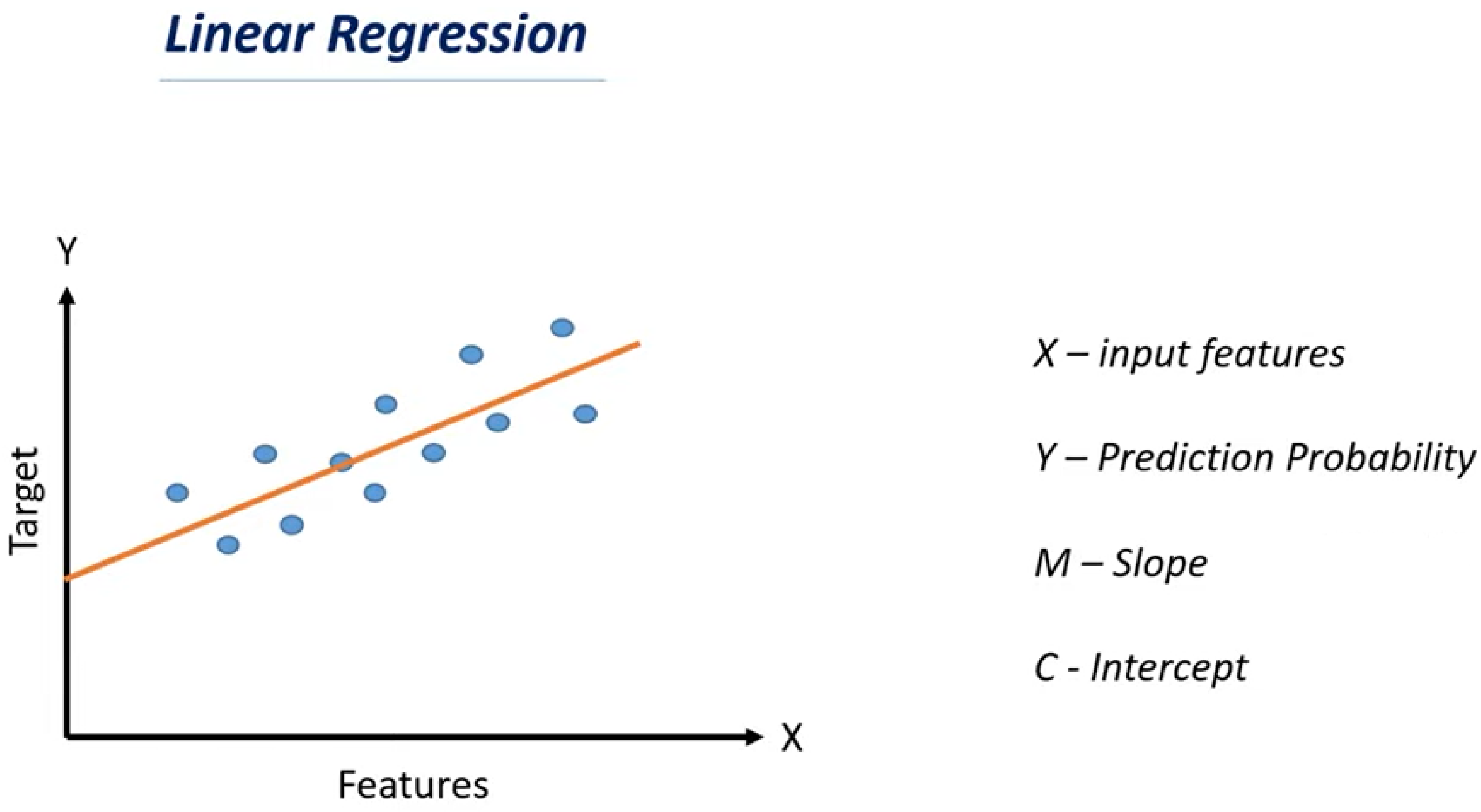
Linear Regression



EXplanation:

- Here we have 2 axis ie,X-axis and Y-axis....We are taking features in the X-axis and Target in the Y-axis

- Here the features are nothing but Age,Bmi,Number of children,Smoker and Region...These 6-columns are features....and the TARGET colmumn is the charges



Explanation:

- Lets say we have some data points..so when we use a linear regression model it tries to fit in this data points....

- The eqaution of line is Y=mx+c.....

- Here X-represents the input features...Y-represents the probaility of our prediction ,M-represents the slope and C-represents the intercept

Linear Regression

In [39]:
```
# Loading the Linear Regression model
regressor = LinearRegression()
```

Now fitting this model to our training and testing data

In [40]:
```
regressor.fit(X_train, Y_train)
```

Out[40]:
```
LinearRegression()
```

Now training of our model is done....thereafter we have to evaluate the model

In [41]:
```
# prediction on training data
training_data_prediction =regressor.predict(X_train)
```

In [42]:
```
# R squared value
r2_train = metrics.r2_score(Y_train, training_data_prediction)
# Here we are comparing the original Y-train values and predicted values
print('R squared vale : ', r2_train)
```

```
R squared vale :  0.751505643411174
```

In [43]:
```
# prediction on test data
test_data_prediction =regressor.predict(X_test)
```

In [44]:
```
# R squared value
r2_test = metrics.r2_score(Y_test, test_data_prediction)
print('R squared vale : ', r2_test)
```

```
R squared vale :  0.7447273869684077
```

Building a Predictive System

In [45]:

```
input_data = (31,1,25.74,0,1,0) # This input data is taken from our csv file...
# Here the input data is tuple...


# changing input_data ie,tuple to a numpy array....
# Why changing?? ---Because it is easy to do some processing on numpy arrays rather than tuples
input_data_as_numpy_array = np.asarray(input_data)

# reshape the array
# What we are doing in reshaping::
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = regressor.predict(input_data_reshaped)
print(prediction)

print('The insurance cost is USD ', prediction[0])
```

```
[3760.0805765]
The insurance cost is USD  3760.0805764960514

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\base.py:445: UserWarning: X does not have valid feature nam
es, but LinearRegression was fitted with feature names
  warnings.warn(
```

31,female,25.74,0,no,southeast,3756.6216

- This was our input data....here our model has made a very close prediction...hence ,we can say our model is performing better

In [ ]: