In [18]:

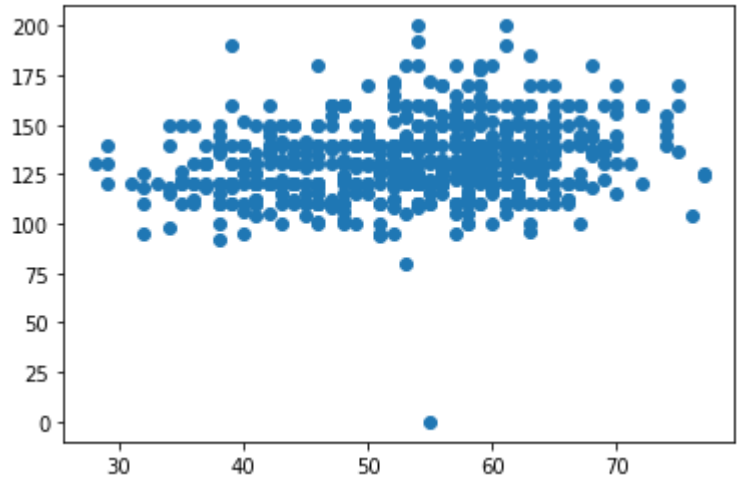
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
import statistics
import datetime
from statistics import mode,mean
os.chdir("C:/Data Science")



data = pd.read_csv('heart_disease_uci.csv')
data.head()
data.columns
data.info()
data = data[data['sex']=='Male']
data = data[["age","trestbps"]]
print(data.head())
plt.scatter(data["age"],data["trestbps"])
plt.show()
num = int(len(data)*0.9)
#training
train = data[:num]
#testing
test = data[num:]
print("Data:",len(data))
print("Train:",len(train))
print("Test:",len(test))

#Main function to find the coefficients of line:
def simple_linear_regression(input_feature, output):
 Xi = input_feature
 Yi = output

#Total number of data points:
 n = len(Xi)
#X bar:
 Xi_mean = Xi.mean( )

#Y bar:
 Yi_mean = Yi.mean ( )
#Sum of X:
 S_Xi = (Xi).sum( )
#Sum of Y:

 S_Yi = (Yi).sum( )
#Sum of (X*Y) multiplied by n:

 S_XiYi = ((Xi*Yi).sum( ) )*n
#Sum of X*Sum of Y:

 S_Xi_S_Yi = S_Xi*S_Yi
#Sum of (X*X) multiplied by n:
 S_XiXi = ((Xi*Xi).sum( ) )*n

#Square of sum of X:
 S_Xi_Square = S_Xi*S_Xi
#Slope:
```

```
 slope = (S_XiYi- S_Xi_S_Yi) / (S_XiXi-S_Xi_Square)
#Intercept:
 intercept = Yi_mean - slope * Xi_mean
 return slope, intercept

#Training the model with train data:
#Finding the coefficients of best fit line:
actual_slope, actual_intercept = simple_linear_regression(train["age"], train["trestbps"])

print ("Slope: " , actual_slope)
print ("Intercept: " , actual_intercept)

#Plot the regression Line with training data:
plt.scatter(train ["age"], train["trestbps"])
plt.plot(train["age"], actual_slope*train ["age"]+actual_intercept, color="green")
plt.xlabel("age")
plt.ylabel("trestbps")
plt.show( )

#Define the prediction function:
def get_regression_prediction(input_features, slope, intercept) :
 predicted_value = actual_slope*input_features + actual_intercept
 return predicted_value

#Predicting values based on prediction function:
age = 38

estimated_bps = get_regression_prediction(age, actual_slope, actual_intercept)
print ("Estimated BPS for age ",age ,'is ', estimated_bps)

#Predicting values for the whole dataset:
y_pred = get_regression_prediction (data["trestbps"], actual_slope, actual_intercept)
print (y_pred)

#Create a dataframe for Actual and Predicted values:
A_P_data = pd.DataFrame ({"Age" : data["age"],"Actual" : data["trestbps"] , "Predicted" :y_
print (A_P_data.head())
#Plot the bar graph for actual and predicted values:
#A_P_data.plot(kind='bar',figsize=(12, 6))
A_P_data.head(20).plot(kind='bar',figsize=(14, 7))
plt.title('Actual Vs Predicted trestbps for gender type Male with Linaer Regression')
plt.show( )


#Error calculation using Residual Sum of Squares:
def residual_sum_of_squares (input_feature, output, slope, intercept):
    prediction = slope*input_feature + intercept
    residual = (output - prediction)
    RSS = (residual*residual).sum()
    return (RSS )
print("RSs: ",residual_sum_of_squares(test["age"],test["trestbps"],actual_slope,actual_inte
```

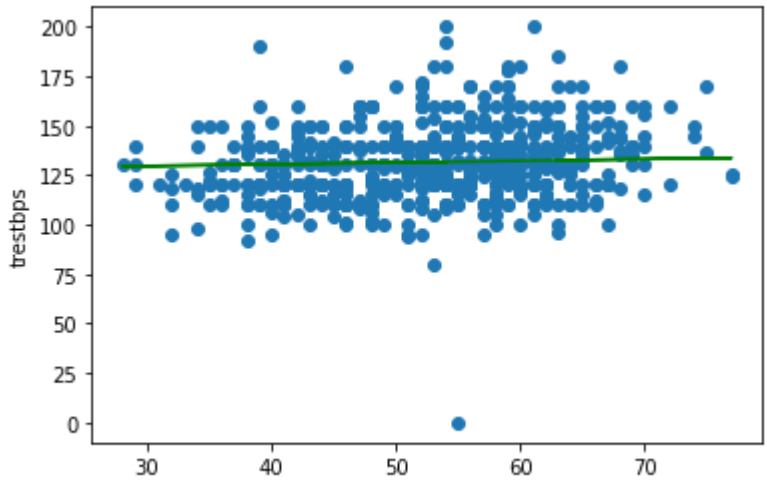
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 920 entries, 0 to 919
Data columns (total 16 columns):
 #   Column    Non-Null Count  Dtype
```

```
---  ------    --------------  -----
 0   id        920 non-null    int64
 1   age       920 non-null    int64
 2   sex       920 non-null    object
 3   dataset   920 non-null    object
 4   cp        920 non-null    object
 5   trestbps  861 non-null    float64
 6   chol      890 non-null    float64
 7   fbs       830 non-null    object
 8   restecg   918 non-null    object
 9   thalch    865 non-null    float64
 10  exang     865 non-null    object
 11  oldpeak   858 non-null    float64
 12  slope     611 non-null    object
 13  ca        309 non-null    float64
 14  thal      434 non-null    object
 15  num       920 non-null    int64
dtypes: float64(5), int64(3), object(8)
memory usage: 115.1+ KB
```

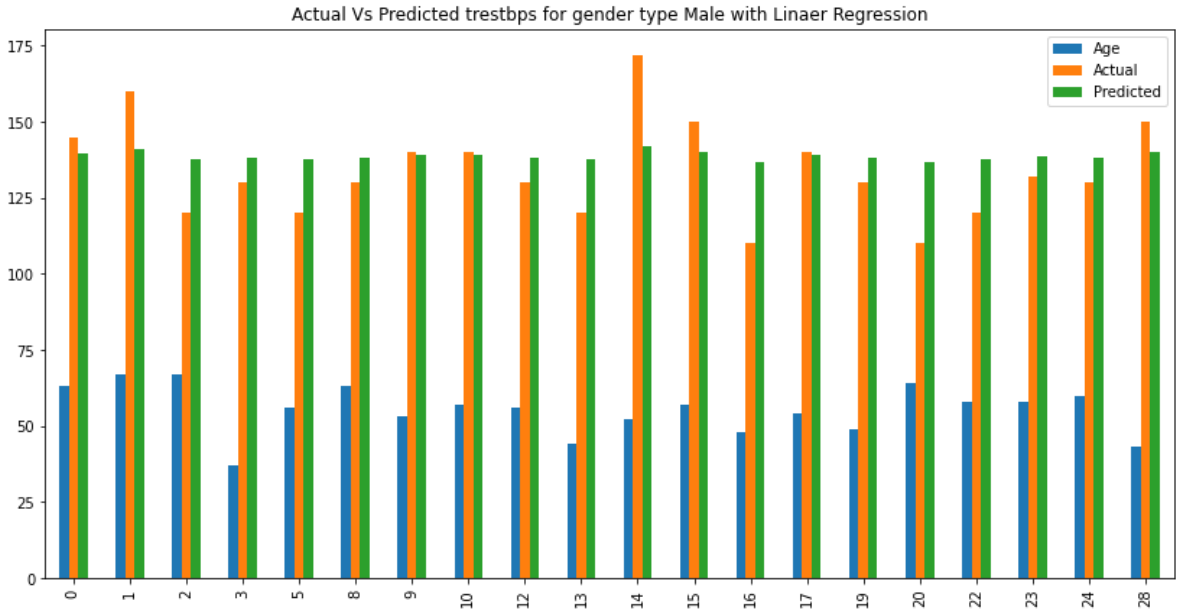
	age	trestbps
0	63	145.0
1	67	160.0
2	67	120.0
3	37	130.0
5	56	120.0




```
Data: 726
Train: 653
Test: 73
Slope:  0.08705707545222491
Intercept:  127.01109276114424
```




```
Estimated BPS for age  38 is  130.31926162832877
0      139.634369
1      140.940225
2      137.457942
3      138.328513
5      137.457942
          ...
914    138.676741
916           NaN
917    137.632056
918           NaN
919    137.457942
Name: trestbps, Length: 726, dtype: float64
   Age  Actual   Predicted
0   63   145.0  139.634369
1   67   160.0  140.940225
2   67   120.0  137.457942
3   37   130.0  138.328513
5   56   120.0  137.457942
```




```
RSs:  20857.950392872164
```

In [ ]: