HW_03 by Andrew Lee

Load Data from google drive

In [113]:

```
from google.colab import drive
drive.mount("/content/gdrive")
```

```
Drive already mounted at /content/gdrive; to attempt to forcibly remount,
call drive.mount("/content/gdrive", force_remount=True).
```

In [114]:

```
import pandas as pd
from sklearn.linear_model import LinearRegression

# import data from csv

df = pd.read_csv('/content/gdrive/MyDrive/Colab Notebooks/dataset/wentworth_applied_ana
lytics - kpisetting.csv.csv')

print(df.head())
print(df.columns)
```

```
         date  visitors  downloads  installations  28dactive
0  01/14/2015     16489       1826            570        270
1  01/15/2015     16362        936            266        104
2  01/16/2015     16463        188             61         67
3  01/17/2015     15972        474            112         40
4  01/18/2015     16659        186            109         32
Index(['date', 'visitors', 'downloads', 'installations', '28dactive'], dty
pe='object')
```

If Company X built a linear regression model to predict for installations using visitors - how many installations might Company X expect to generate when it acquires 230,000 visitors in a single day?

In [115]:

```
# Independent variable, or input
x = df[['visitors']]
# Deendent variable, or output
y = df['installations']
# Setting up model inputs
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
```

In [116]:

```
# Training of the model
model = LinearRegression()
model.fit(x_train, y_train)
predictions = model.predict(x_test)
# Get 230000 vistors prediction
data = [230000]
inputpredict = pd.DataFrame(data,columns=['visitors'])
outputprdict = model.predict(inputpredict)
print(round(outputprdict[0]))
# getting r2 and mse
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(y_test, predictions)
r2 = r2_score(y_test, predictions)
```

```
4160
```

Using vistors feature in the linear regression, we expect 4243 installations when there is 230000 visitors in a single day.

In [117]:

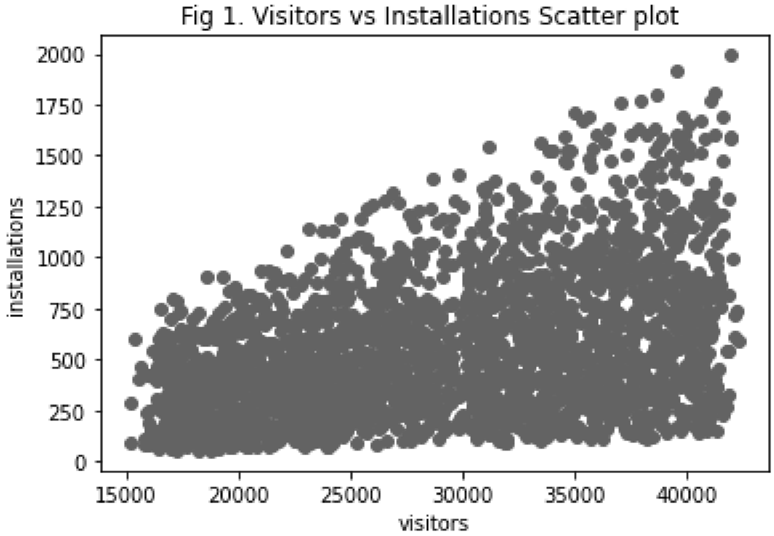
```
print("mse: ",mse)
print("r2:",r2)
```

```
mse:  90706.3592037313
r2: 0.19869484627746992
```

After we train the model with 80% of data, we found out the r^2 is around 0.15 which shows the model doesn't fit our observations. We also find out the mean squared error is around 87830.

In [118]:

```
import matplotlib.pyplot as plt
# Create the scatter plot
plt.scatter(x, y)
plt.xlabel("visitors")
plt.ylabel("installations")
plt.title("Fig 1. Visitors vs Installations Scatter plot ")
# Show the plot
plt.show()
```



In Fig 1 scatter plot, although we can see there is a positive trend between installation and vistors, the data points speard out in a huge area.

In [119]:

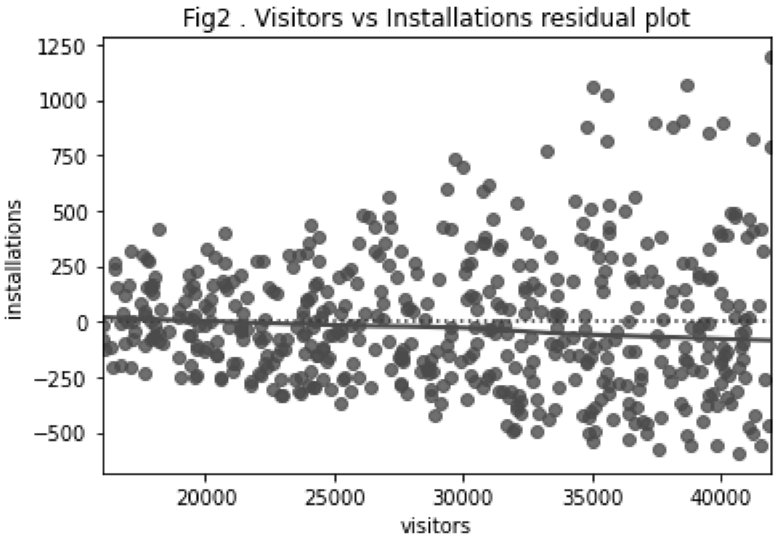
```
import seaborn as sns

# Make predictions and calculate residuals
residuals = y_test - predictions

# Plot the residuals

sns.residplot(x_test, y_test, lowess=True, color="g",x="visitors", y="installations")
plt.title("Fig2 . Visitors vs Installations residual plot")
plt.show()
```

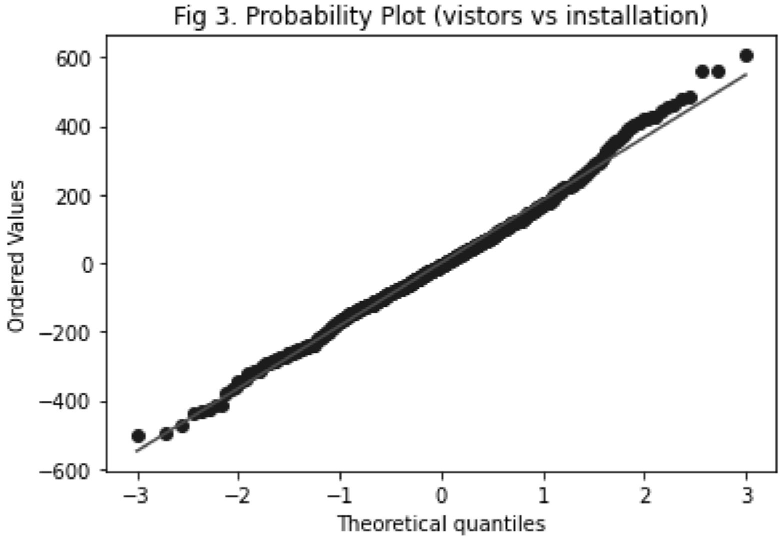
```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWa
rning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other a
rguments without an explicit keyword will result in an error or misinterpr
etation.
  warnings.warn(
```



In Fig 2, the residual plot is the sightly heteroscedasticity. As the vistor increase, the difference between the mean and the observate increase. The difference between the mean and the observate are also huge.

In [128]:

```
import scipy.stats as stats
# Plot the residuals
stats.probplot(residuals, plot=plt)
plt.title("Fig 3. Probability Plot (vistors vs installation)")
plt.show()
```



In the Probability Plot above , we can see the data points on both tails are off the distribution. We can conclue the residuals are sightly not normally distributed.

Using vistors feature in the linear regression, we expect 4243 installations when there is 230000 visitors in a single day. However, it seems linear regression model is not the best fit to predict the result the based on lower value on r2, and non-normality distributed.

If Company X built a linear regression model to predict for installations using downloads - how many installations might Company X expect to generate when it acquires 195,000 downloads in a single day?

In [121]:

```
# Independent variable, or input
x = df[['downloads']]
# Deendent variable, or output
y = df['installations']
# Setting up model inputs

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2)
```

In [122]:

```
# Training of the model
model = LinearRegression()
model.fit(x_train, y_train)
predictions = model.predict(x_test)
# getting r2 and mse
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(y_test, predictions)
r2 = r2_score(y_test, predictions)
# Get 230000 vistors prediction
data = [195000]
inputpredict = pd.DataFrame(data,columns=['downloads'])
outputprdict = model.predict(inputpredict)
print(round(outputprdict[0]))
```

```
56911
```

Using downloads feature in the linear regression, we expect 57603 installations when there is 195000 downloads in a single day.

In [123]:

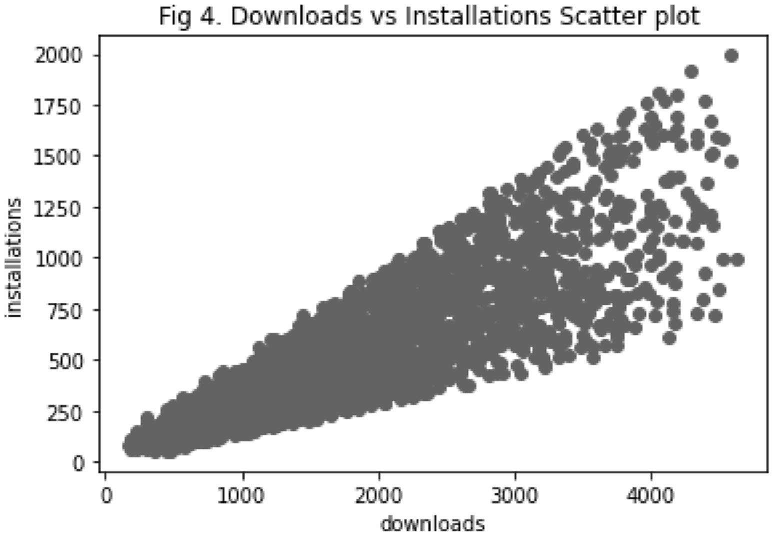
```
print("mse: ",mse)
print("r2:",r2)
```

```
mse:  33409.89760588258
r2: 0.7233954061447316
```

After we train the model with 80% of data, we found out the r^2 is around 0.72 which shows the model is fit our observations. We also find out the mean squared error is around 33175.

In [124]:

```
import matplotlib.pyplot as plt
# Create the scatter plot
plt.scatter(x, y)
plt.xlabel("downloads")
plt.ylabel("installations")
plt.title("Fig 4. Downloads vs Installations Scatter plot ")
# Show the plot
plt.show()
```



In Fig 4 scatter plot, we can see there is a positive trend between installation and vistors, the data points doesn't spread out a lot.

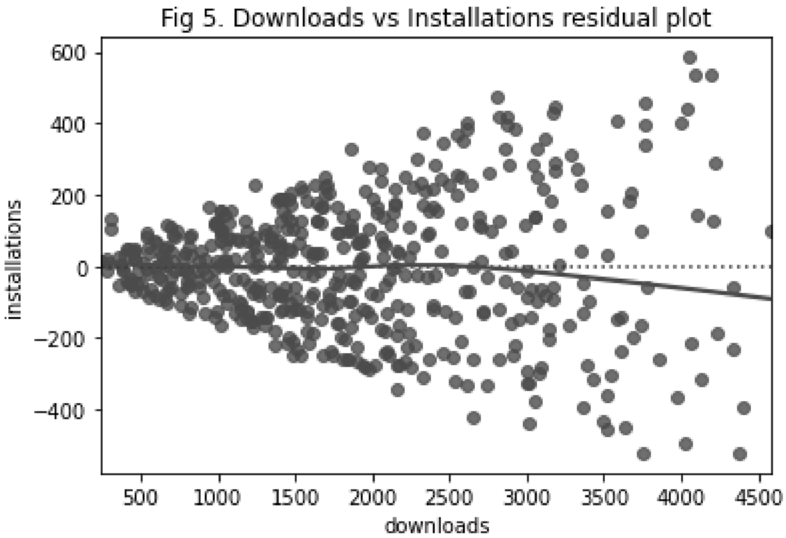
In [125]:

```
import seaborn as sns

residuals = y_test - predictions

# Plot the residuals
sns.residplot(x_test, y_test, lowess=True, color="g",x="downloads", y="installations")
plt.title("Fig 5. Downloads vs Installations residual plot")
plt.show()
```

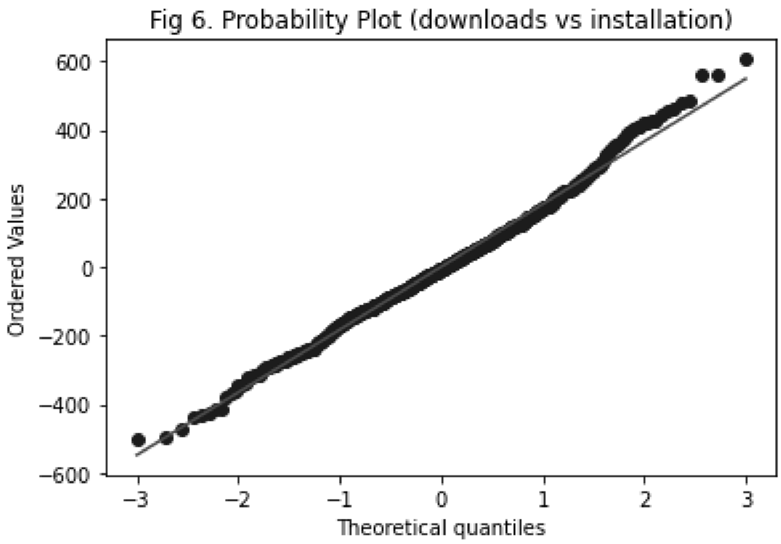
```
/usr/local/lib/python3.8/dist-packages/seaborn/_decorators.py:36: FutureWa
rning: Pass the following variables as keyword args: x, y. From version 0.
12, the only valid positional argument will be `data`, and passing other a
rguments without an explicit keyword will result in an error or misinterpr
etation.
  warnings.warn(
```



In Fig 2, the residual plot is the heteroscedasticity and have a fan shape. As the vistor increase, the difference between the mean and the observate increase.

In [127]:

```
import scipy.stats as stats
# Plot the residuals
stats.probplot(residuals, plot=plt)
plt.title("Fig 6. Probability Plot (downloads vs installation)")
plt.show()
```



In the Probability QQ Plot above , we can see most data points are on the distribution. We can conclue the residuals are normally distributed.

Using downloads feature in the linear regression, we expect 57603 installations when there is 195000 downloads in a single day. Since there is high r2 value and the residual is normally distriubted, I would say the linear regression is reliable for this prediction.