Title: Sales Data Analysis

Objectives: To analyze the sales data and answer to the following questions.

1. Which was the best month for sales? and amount earned in it?
2. Which city has the highest number of sales?
3. At what time should we display ads to maximize likelihood of customer buying the product?
4. Which product sold the most?
5. Which product is most often bought together by the customers?
6. Prediction of sales in a specific month using machine learning.

Details of Data Set Used: Electronic Shop Month Wise Sales by Kaggle.

Steps:

The dataset is in structured form, so the following steps are done:

a) Data Interpretation:

1. Data is integrated from multiple .csv files to single csv file.
2. Describing our data set:

```
        Order ID                Product Quantity Ordered Price Each  Order Date  Purchase Address
count      11646                  11646            11646      11646       11646             11646
unique     11208                     20                7         24        9495             11032
top     Order ID  USB-C Charging Cable                 1      11.95  Order Date  Purchase Address
freq          17                   1454            10490       1454          17                17
PS C:\Code> _
```

b) Data Preprocessing:

1. Adding month, sales, city, hour columns for convenience
2. Changing data types to time stamp
3. Missing values detected by isna() pandas function.
4. Cleaned missing values using dropna() pandas function.

Classifiers used for data modelling with its theory:

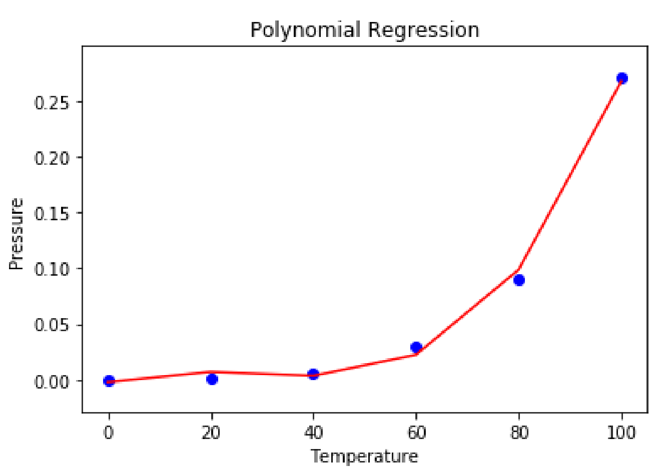
Polynomial Regression:

In statistics, polynomial regression is a form of regression analysis in which the relationship between the independent variable x and the dependent variable y is modelled as an nth degree polynomial in x. Polynomial regression fits a nonlinear relationship between the value of x and the corresponding conditional mean of y, denoted E(y |x). Although polynomial regression fits a nonlinear model to the data, as a statistical estimation problem it is linear, in the sense that the regression function E (y | x) is linear in the unknown parameters that are estimated from the data. For this reason, polynomial regression is a special case of multiple linear regression.

The explanatory (independent) variables resulting from the polynomial expansion of the "baseline" variables are known as higher-degree terms. Such variables are also used in classification settings

Why Polynomial Regression:

- There are some relationships that a researcher will hypothesize is curvilinear. Clearly, such type of cases will include a polynomial term.
- Inspection of residuals. If we try to fit a linear model to curved data, a scatter plot of residuals (Y axis) on the predictor (X axis) will have patches of many positive residuals in the middle. Hence in such situation it is not appropriate.
- An assumption in usual multiple linear regression analysis is that all the independent variables are independent. In polynomial regression model, this assumption is not satisfied.



The basic goal of regression analysis is to model the expected value of a dependent variable y in terms of the value of an independent variable x. In simple regression, we used following equation –

y = a + bx + e

Here y is dependent variable, a is y intercept, b is the slope and e is the error rate.

In many cases, this linear model will not work out For example if we analyzing the production of chemical synthesis in terms of temperature at which the synthesis take place in such cases we use quadratic model.

y = a + b1x + b2^2 + e

Here y is dependent variable on x, a is y intercept and e is the error rate.

In general, we can model it for nth value.

y = a + b1x + b2x^2 +....+ bnx^n

Since regression function is linear in terms of unknown variables, hence these models are linear from the point of estimation.

Hence through Least Square technique, let's compute the response value that is y.

Advantages of using Polynomial Regression:

- Broad range of function can be fit under it.
- Polynomial basically fits wide range of curvature.
- Polynomial provides the best approximation of the relationship between dependent and independent variable.

Disadvantages of using Polynomial Regression:

- These are too sensitive to the outliers.
- The presence of one or two outliers in the data can seriously affect the results of a nonlinear analysis.
- In addition, there are unfortunately fewer model validation tools for the detection of outliers in nonlinear regression than there are for linear regression.

Coding:

Project is made in python3 programming language. We used famous pandas library to deal with csv files. Graphs are plotted using plotting libraries.

Following are the libraries we used:

1. Pandas
2. Matplotlib
3. Pyplot
4. Itertools
5. Sklearn
6. Statisticalmodels
7. Collections

Main.py

```
import pandas as pd
import os
import matplotlib.pyplot as plt
from itertools import combinations
from collections import Counter

def get_city(address):
    return address.split(',')[1]

def get_state(address):
    return address.split(',')[2].split(' ')[1]
df=pd.read_csv("../Sales_Data/Sales_April_2019.csv")
print(df)

all_months_data=pd.DataFrame()
desc
files=[file for file in os.listdir("../Sales_Data")]
for file in files:
    df=pd.read_csv("../Sales_Data/"+file)
    all_months_data=pd.concat([all_months_data,df])

all_months_data.to_csv("all_data.csv",index=False)

all_data=pd.read_csv("all_data2.csv")
print(all_data.head())
```

```
#Cleaning the data
nan_df=all_data[all_data.isna().any(axis=1)]
print(nan_df.head())


all_data=all_data.dropna(how='all')
print(all_data.head())

all_data=all_data[all_data['Order Date'].str[0:2]!="Or"]
print(all_data.head())

all_data['Quantity Ordered']=pd.to_numeric(all_data['Quantity Ordered'])
all_data['Price Each']=pd.to_numeric(all_data['Price Each'])
all_data['Order Date']=pd.to_datetime(all_data['Order Date'])

#Adding columns for convinience

# Add Month Column

all_data['Month']=all_data['Order Date'].dt.month
print(all_data.head())


#Adding sales Column
all_data['Sales']=all_data['Quantity Ordered']*all_data['Price Each']
print(all_data.head())

# Adding City Column
all_data['City']=all_data['Purchase Address'].apply(lambda x: get_city(x) + '
('+get_state(x)+')')
print(all_data.head())

# Adding Hur Column
all_data['Hour']=all_data['Order Date'].dt.hour
all_data['Minute']=all_data['Order Date'].dt.minute

#What was the best month for sales? and amt earned in it

results=(all_data.groupby('Month').sum())
```

```
months=range(1,13)
plt.bar(months,results['Sales'])
plt.xticks(months)
plt.ylabel('Sales in USD ($)')
plt.xlabel('Month number')
plt.show()

# What city has the highest  number of sales
results=all_data.groupby('City').sum()
cities=[city for city,df in all_data.groupby('City')]
plt.bar(cities,results['Sales'])
plt.xticks(cities,rotation='vertical',size=8)
plt.ylabel('Sales in USD ($)')
plt.xlabel('Citiy name')
plt.show()


# What time should we display ads to maximize likelihood of customer buying
the product

hours =[hour for hour,df in all_data.groupby('Hour')]
plt.plot(hours,all_data.groupby(['Hour']).count())
plt.xticks(hours)
plt.xlabel('Hour')
plt.ylabel('Number of orders')
plt.grid()
plt.show()


#What product sold the most
product_group = all_data.groupby('Product')
quantity_ordered=product_group.sum()['Quantity Ordered']

products = [product for product,df in product_group]
prices=all_data.groupby('Product').mean()['Price Each']
fig,ax1 = plt.subplots()
ax2=ax1.twinx()
ax1.bar(products,quantity_ordered,color='g')
ax2.plot(products,prices,'b-')
```

```
ax1.set_xlabel('Products')
ax1.set_ylabel('Quantity Bought',color='g')
ax2.set_ylabel('Price in USD ($)',color='b')
ax1.set_xticklabels(products,rotation='vertical',size=8)
plt.show()

# What product are most often bought together by the customers
df=all_data[all_data['Order ID'].duplicated(keep=False)]
df['Grouped']=df.groupby('Order ID')['Product'].transform(lambda x:','.join(x))
df=df[['Order ID','Grouped']].drop_duplicates()
print(df.head())


count=Counter()
for row in df['Grouped']:
   row_list=row.split(',')
   count.update(Counter(combinations(row_list,2)))

for key,value in count.most_common(10):
   print(key,value)
```

Polynomial Model:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import PolynomialFeatures
from sklearn import linear_model
from sklearn.model_selection import train_test_split
import statsmodels.api as sm

#read the csv file in df as dataframe
df = pd.read_csv('F:\college ppts\data science\dayproductsum.csv')
#extract the required columns here we have taken month,Sales,day
df=df[['month','Sales','day']]
print("\nCorrelation matrix")
print(df.corr())
print("\nNUll Values")
print(df.isnull().sum())
#define the dependent(Y) and independent(X) variable
X=df[['month','day']]
Y=df['Sales']

#spliting the dataset in 25% (75% to train the model and 25% to test the model)
X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size =
0.25,random_state = 40)

#creating the polynomial model with degree 6
poly = PolynomialFeatures(degree=6)
X_ = poly.fit_transform(X_train)
predict_ = poly.fit_transform(X_test)
clf = linear_model.LinearRegression()
#fitting the model with train dataset
clf.fit(X_, Y_train)
#predicting the value with the trained dataset
Y_predict = clf.predict(predict_)


#calculating rsq value and the standard error
err = Y_test-Y_predict

est = sm.OLS(Y,X)
```

```
est2 = est.fit()
print(est2.summary())

#residue plot with the independent variable
plt.scatter(Y_predict,err,alpha=0.7)
plt.ylabel("Residue")
plt.xlabel("Predicted Value")
plt.title("Residual plot")


print("\nPredict the sale")
predict_ = poly.fit_transform([[12,25]])
Y_predict = clf.predict(predict_)
print(Y_predict)
```

Results:

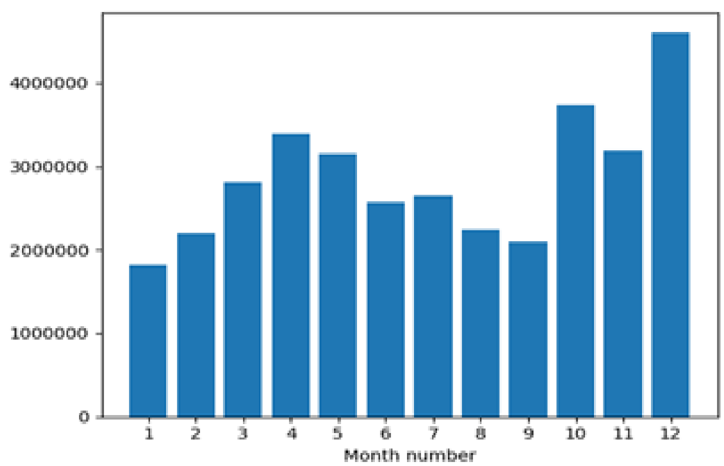
1. Data Interpretation:

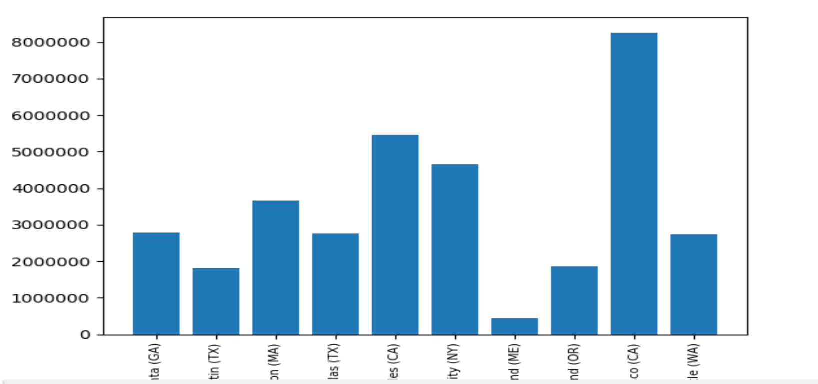
```
PS C:\Code> python .\main.py
      Order ID                 Product  ...     Order Date                       Purchase Address
0       176558        USB-C Charging Cable  ...  04/19/19 08:46          917 1st St, Dallas, TX 75001
1          NaN                     NaN  ...            NaN                                    NaN
2       176559  Bose SoundSport Headphones  ...  04/07/19 22:30     682 Chestnut St, Boston, MA 02215
3       176560            Google Phone  ...  04/12/19 14:38   669 Spruce St, Los Angeles, CA 90001
4       176560        Wired Headphones  ...  04/12/19 14:38   669 Spruce St, Los Angeles, CA 90001
...        ...                     ...  ...            ...                                    ...
18378   194090            Google Phone  ...  04/08/19 17:11  177 Jackson St, Los Angeles, CA 90001
18379   194091     AA Batteries (4-pack)  ...  04/15/19 16:02        311 Forest St, Austin, TX 73301
18380   194092    AAA Batteries (4-pack)  ...  04/28/19 14:36 347 Sunset St, San Francisco, CA 94016
18381   194093     AA Batteries (4-pack)  ...  04/14/19 15:09         835 Lake St, Portland, OR 97035
18382   194094   Lightning Charging Cable  ...  04/18/19 11:08          354 North St, Boston, MA 02215

[18383 rows x 6 columns]
  Order ID                 Product  ...        Order Date                      Purchase Address
0   176558        USB-C Charging Cable  ...    04/19/19 08:46          917 1st St, Dallas, TX 75001
1   176559  Bose SoundSport Headphones  ...  04/07/2019 22:30     682 Chestnut St, Boston, MA 02215
2   176560            Google Phone  ...  04/12/2019 14:38  669 Spruce St, Los Angeles, CA 90001
3   176560        Wired Headphones  ...  04/12/2019 14:38  669 Spruce St, Los Angeles, CA 90001
4   176561        Wired Headphones  ...    04/30/19 09:27     333 8th St, Los Angeles, CA 90001

[5 rows x 6 columns]
PS C:\Code>
```

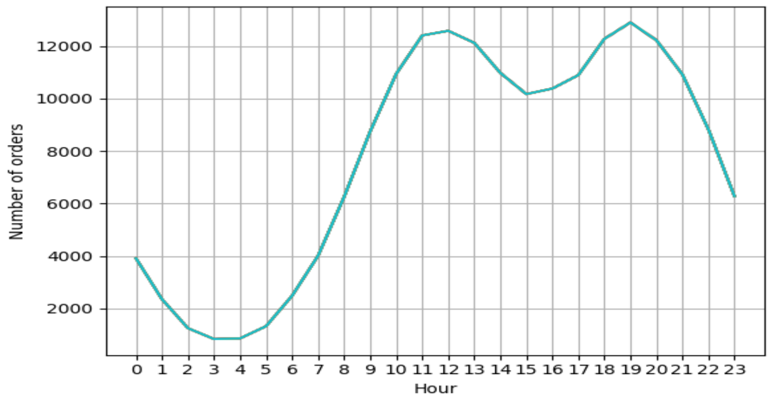
2. Data Preprocessing:

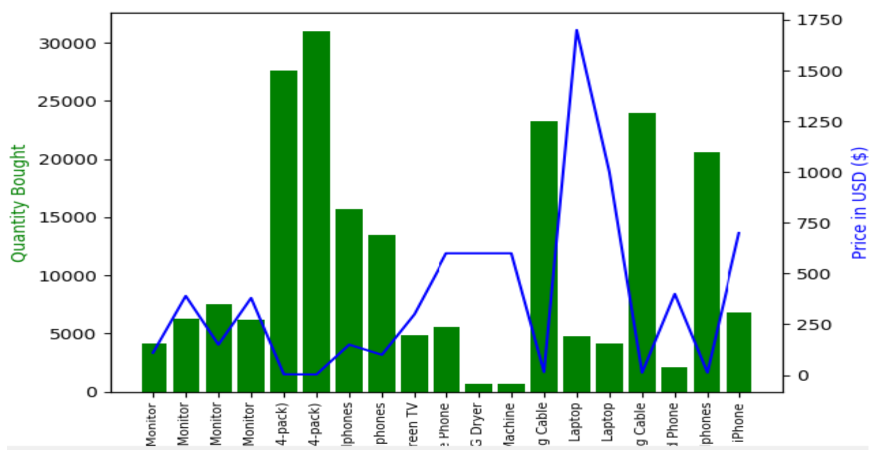
```
[5 rows x 6 columns]
     Order ID Product Quantity Ordered Price Each Order Date Purchase Address
355       NaN     NaN              NaN        NaN        NaN              NaN
734       NaN     NaN              NaN        NaN        NaN              NaN
1432      NaN     NaN              NaN        NaN        NaN              NaN
1552      NaN     NaN              NaN        NaN        NaN              NaN
1570      NaN     NaN              NaN        NaN        NaN              NaN
PS C:\Code> _
```

3. Data Visualization:









Observations/Inferences: We were successfully able to analyze the data set and able to model in visual form through graphs. All objectives are successfully represented in this project.

Residual Plot: