Exercise 1 - Part A

November 19, 2021

Machine Learning Lab

Lab 02

Exercise 1 - Part A

Importing Packages

[1]:
```
import pandas as pd       #Importing Pandas
import numpy as np        #Importing Numpy
```

Reading Training CSV data into the Dataframe

[2]:
```
train_df = pd.read_csv('train.csv',low_memory=False)
#Setting Store column as my Index in Dataframe
train_df.set_index('Store',inplace=True)
train_df.head()
```

[2]:

Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
1	5	2015-07-31	5263	555	1	1	0	1
2	5	2015-07-31	6064	625	1	1	0	1
3	5	2015-07-31	8314	821	1	1	0	1
4	5	2015-07-31	13995	1498	1	1	0	1
5	5	2015-07-31	4822	559	1	1	0	1

Reading Store CSV data into the Dataframe

[3]:
```
store_df = pd.read_csv('store.csv')
#Setting Store column as my Index in Dataframe
store_df.set_index('Store',inplace=True)
store_df.head()
```

[3]:
Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear	PromoInterval
1	c	a	1270.0	9.0	2008.0	0	NaN	NaN	NaN
2	a	a	570.0	11.0	2007.0	1	13.0	2010.0	Jan,Apr,Jul,Oct
3	a	a	14130.0	12.0	2006.0	1	14.0	2011.0	Jan,Apr,Jul,Oct
4	c	c	620.0	9.0	2009.0	0	NaN	NaN	NaN
5	a	a	29910.0	4.0	2015.0	0	NaN	NaN	NaN

Merging Training and Store Dataframes into a single Dataframe

[4]:
```
#Merging train and store dataframes on Store column
merged_df = pd.merge(train_df, store_df, how='inner', on='Store')
merged_df.head()
```

[4]:
Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2
1	5	2015-07-31	5263	555	1	1	0	1	c	a	1270.0	9.0	2008.0	0
1	4	2015-07-30	5020	546	1	1	0	1	c	a	1270.0	9.0	2008.0	0
1	3	2015-07-29	4782	523	1	1	0	1	c	a	1270.0	9.0	2008.0	0
1	2	2015-07-28	5011	560	1	1	0	1	c	a	1270.0	9.0	2008.0	0
1	1	2015-07-27	6102	612	1	1	0	1	c	a	1270.0	9.0	2008.0	0

```
1                          9.0                    2008.0       0
1                          9.0                    2008.0       0
1                          9.0                    2008.0       0
1                          9.0                    2008.0       0
1                          9.0                    2008.0       0

       Promo2SinceWeek  Promo2SinceYear PromoInterval
Store
1                  NaN              NaN           NaN
1                  NaN              NaN           NaN
1                  NaN              NaN           NaN
1                  NaN              NaN           NaN
1                  NaN              NaN           NaN
```

Cleaning and Preparing Dataframe for Analysis

[5]:
```
#Getting an overview which columns has null values
merged_df.isnull().sum()
```

[5]:
```
DayOfWeek                       0
Date                            0
Sales                           0
Customers                       0
Open                            0
Promo                           0
StateHoliday                    0
SchoolHoliday                   0
StoreType                       0
Assortment                      0
CompetitionDistance          2642
CompetitionOpenSinceMonth  323348
CompetitionOpenSinceYear   323348
Promo2                          0
Promo2SinceWeek            508031
Promo2SinceYear            508031
PromoInterval              508031
dtype: int64
```

[6]:
```
#For NA values in column Competition Distance, we are filling it with the␣
 ↪median value of the whole column
merged_df.CompetitionDistance.fillna(merged_df.CompetitionDistance.median(),␣
 ↪inplace = True)
#For NA in any other column, replacing NA with 0
merged_df.fillna(0, inplace = True)
```

Find the store that has the maximum sale recorded. Print the store id, date and the sales on that day

[7]:
```
#Filtering rows on the basis of maximum Sales value
store_max_sale = merged_df[merged_df.Sales == merged_df.Sales.max()]
print('Store with Id: {} has the maximum sales on {} with value: {}\n'.format(
    store_max_sale.index[0],store_max_sale.Date.values[0],store_max_sale.Sales.
 ↪values[0]))
store_max_sale
```

```
Store with Id: 909 has the maximum sales on 2015-06-22 with value: 41551
```

[7]:

Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear	PromoInterval
909	1	2015-06-22	41551	1721	1	0	0	0	a	c	1680.0	0.0	0.0	1	45.0	2009.0	Feb,May,Aug,Nov

Find the store(s) that has/ve the least possible and maximum possible competition distance(s).

[8]:
```
#Sorting the dataframe in descending on Competition Distance to get the Maximum␣
 ↪and Minimum values
competition_distance = merged_df.
 ↪sort_values('CompetitionDistance',ascending=False)
print('The Store with least Possible Competition Distance is :{} \n'.
 ↪format(competition_distance.iloc[-1].name))
print(competition_distance.iloc[-1])
print('\nThe Store with Maximum Possible Competition Distance is : {} \n'.
 ↪format(competition_distance.iloc[0].name))
print(competition_distance.iloc[0])
```

```
The Store with least Possible Competition Distance is :516

DayOfWeek                          7
Date                      2013-09-15
Sales                             0
Customers                         0
Open                              0
Promo                             0
StateHoliday                      0
```

```
SchoolHoliday                              0
StoreType                                  a
Assortment                                 c
CompetitionDistance                     20.0
CompetitionOpenSinceMonth                0.0
CompetitionOpenSinceYear                 0.0
Promo2                                     1
Promo2SinceWeek                         35.0
Promo2SinceYear                       2010.0
PromoInterval                 Mar,Jun,Sept,Dec
Name: 516, dtype: object

The Store with Maximum Possible Competition Distance is : 453

DayOfWeek                                  4
Date                              2013-05-02
Sales                                  12834
Customers                               1383
Open                                       1
Promo                                      1
StateHoliday                               0
SchoolHoliday                              0
StoreType                                  a
Assortment                                 c
CompetitionDistance                  75860.0
CompetitionOpenSinceMonth              0.0
CompetitionOpenSinceYear               0.0
Promo2                                     0
Promo2SinceWeek                        0.0
Promo2SinceYear                        0.0
PromoInterval                              0
Name: 453, dtype: object
```

What has been the maximum timeline a store has ran a "Promo" for? Which store was that, and what dates did the promotion covered?

[9]:
```
#Sorting the Merged Dataframe on Store and then Date columns and then grouping␣
 ↪it on the basis of Store Id
store_promo_df = merged_df.sort_values(by=['Store','Date']).groupby(by='Store')
store_promo_dict = {}

#For each Store Id we compute the maximum number of times 1 appears␣
 ↪consecutively
#and then storing the maximum times into a dictionary with store Id as a key
for group, value in store_promo_df:
    max_promo = np.where(
        value["Promo"].eq(1),
```

```
        value.groupby(value.Promo.ne(value.Promo.shift()).cumsum()).cumcount()␣
↪+ 1,
        0,
    ).max()
    store_promo_dict[group] = max_promo

#Converting the dictionary into the Dataframe with Store Id as Index
store_promo_df = pd.DataFrame.
↪from_dict(store_promo_dict,orient='index',columns=['Max Timeline for Promo'])
store_promo_df.index.name = 'Store'
store_promo_df
```

[9]:

Store	Max Timeline for Promo
1	5
2	5
3	5
4	5
5	5
…	…
1111	5
1112	5
1113	5
1114	5
1115	5

[1115 rows x 1 columns]

What is the difference in the mean of sales (across all stores) when offering a Promo and not?

[10]:
```
#Calculating mean of sales when Store was offering Promo
sales_with_promo = merged_df[merged_df.Promo == True].Sales.mean()
print('Mean of Sales when the Store was offering Promo is: {}'.
↪format(sales_with_promo))

#Calculating mean of sales when Store was NOT offering Promo
sales_without_promo = merged_df[merged_df.Promo == False].Sales.mean()
print('Mean of Sales when the Store was NOT offering Promo is: {}'.
↪format(sales_without_promo))

#Calculating Difference in mean of sales when Store was offering Promo and not
print('Difference in the mean of sales (across all stores) when offering a␣
↪Promo and not is: {}'.format(sales_with_promo - sales_without_promo))
```

```
Mean of Sales when the Store was offering Promo is: 7991.152045969903
Mean of Sales when the Store was NOT offering Promo is: 4406.050805160786
Difference in the mean of sales (across all stores) when offering a Promo and
```

not is: 3585.1012408091174

Are there any anomalies in the data as in where the store was "Open" but had no sales recorded? or vice versa?

[11]:
```
#For anamolies I am checking two conditions:
#-If store is open and there is no state holiday but the total sales is 0
#-If store is closed or there is state holiday but the total sales is greater than 0
anamolies = merged_df[((merged_df.Open == True) & (merged_df.Sales <= 0) & (merged_df.StateHoliday == '0'))
                      | (((merged_df.Open == False) | (merged_df.StateHoliday != '0')) & (merged_df.Sales > 0))]
anamolies
```

[11]:

Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2
2	5	2014-10-03	2689	389	1	1	a	0	a	a	570.0	11.0	2007.0	1
2	4	2013-10-03	2656	381	1	0	a	0	a	a	570.0	11.0	2007.0	1
5	4	2015-06-04	5807	625	1	1	a	0	a	a	29910.0	4.0	2015.0	0
5	4	2014-06-19	5307	574	1	1	a	0	a	a	29910.0	4.0	2015.0	0
5	4	2013-05-30	5065	576	1	1	a	0	a	a	29910.0	4.0	2015.0	0
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
1100	2	2014-04-29	0	3	1	1	0	0	a	a	540.0	11.0	2007.0	1
1107	4	2013-08-15	6138	649	1	1	a	1	a	a	1400.0	12.0	2006.0	1
1108	4	2015-06-04	7404	693	1	1	a	0	a	a	540.0	11.0	2007.0	1
1108	4	2014-06-19	7037	703	1	1	a	0	a	a	540.0	11.0	2007.0	1
1108	4	2013-05-30	6555	675	1	1	a	0	a	a	540.0	11.0	2007.0	1

```
5                          4.0                  2015.0       0
5                          4.0                  2015.0       0
…                          …                    …     …
1100                       0.0                     0.0       1
1107                       6.0                  2012.0       1
1108                       4.0                  2004.0       0
1108                       4.0                  2004.0       0
1108                       4.0                  2004.0       0

       Promo2SinceWeek  Promo2SinceYear    PromoInterval
Store
2                 13.0           2010.0  Jan,Apr,Jul,Oct
2                 13.0           2010.0  Jan,Apr,Jul,Oct
5                  0.0              0.0                0
5                  0.0              0.0                0
5                  0.0              0.0                0
…                  …                …                  …
1100              14.0           2011.0  Jan,Apr,Jul,Oct
1107              13.0           2010.0  Jan,Apr,Jul,Oct
1108               0.0              0.0                0
1108               0.0              0.0                0
1108               0.0              0.0                0

[964 rows x 17 columns]
```

Which store type ('a','b' etc.) has had the most sales?

[12]:
```
#To find Store type with most sales,
#-First we group our dataframe on the basis of store type
#-Then for each store type group, calculate its total sales
#-And lastly sort the resulting dataframe on the basis of sales
store_max_sales = merged_df.groupby(by='StoreType')
store_max_sale = store_max_sales.sum().sort_values(by='Sales',ascending=False)
store_max_sale
```

[12]:
```
           DayOfWeek       Sales  Customers    Open   Promo  SchoolHoliday  \
StoreType
a            2205558  3165334859  363541434  457077  210504          98413
d            1251195  1765392943  156904995  258774  119286          55739
c             547106   783221426   92129705  112978   52244          24653
b              63289   159231395   31465621   15563    6046           2916

           CompetitionDistance  CompetitionOpenSinceMonth  \
StoreType
a                2.847283e+09                  2799319.0
d                2.169958e+09                  1458372.0
c                4.812252e+08                   697454.0
b                1.687570e+07                    56520.0
```

StoreType	CompetitionOpenSinceYear	Promo2	Promo2SinceWeek	Promo2SinceYear
a	783488024.0	257886	6101222.0	518830924.0
d	394492060.0	178508	4183924.0	359095630.0
c	198749786.0	68258	1462148.0	137303804.0
b	17021940.0	4526	100816.0	9109896.0

[13]:
```
print('The Store type with the maximumn Sales is: {} with Sales amount: {}'.
↪format(store_max_sale.index[0],store_max_sale.Sales[0]))
```

```
The Store type with the maximumn Sales is: a with Sales amount: 3165334859
```