Data Source : https://www.kaggle.com/c/rossmann-store-sales/data?select=sample_submission.csv

let's start import library

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

let's read the data

In [2]:

```
train = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')
store = pd.read_csv('store.csv')
```

We got the training shape,test shape and store shape

In [3]:

```
print("Traing Data Shape",train.shape)
print("Test Data shape",test.shape)
print("Store Data shape",store.shape)
```

```
Traing Data Shape (1017209, 9)
Test Data shape (41088, 8)
Store Data shape (1115, 10)
```

let's get top 5 rows

In [4]:

```
train.head()
```

Out[4]:

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

let's get sum of null value

In [5]:

```
train.isnull().sum()
```

Out[5]:

```
Store            0
DayOfWeek        0
```

```
DayOfWeek          0
Date               0
Sales              0
Customers          0
Open               0
Promo              0
StateHoliday       0
SchoolHoliday      0
dtype: int64
```

let' read top 5 test data

In [6]:

```
test.head()
```

Out[6]:

	Id	Store	DayOfWeek	Date	Open	Promo	StateHoliday	SchoolHoliday
0	1	1	4	2015-09-17	1.0	1	0	0
1	2	3	4	2015-09-17	1.0	1	0	0
2	3	7	4	2015-09-17	1.0	1	0	0
3	4	8	4	2015-09-17	1.0	1	0	0
4	5	9	4	2015-09-17	1.0	1	0	0

In [7]:

```
store.head()
```

Out[7]:

	Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2	Promo2Since
0	1	c	a	1270.0	9.0	2008.0	0	
1	2	a	a	570.0	11.0	2007.0	1	
2	3	a	a	14130.0	12.0	2006.0	1	
3	4	c	c	620.0	9.0	2009.0	0	
4	5	a	a	29910.0	4.0	2015.0	0	

In [8]:

```
not_open = train[(train['Open']==0)&(train['Sales']!=0)]
print("No closed store with sales:", str(not_open.size==0))

no_sales = train[(train['Open']==1)&(train['Sales']<=0)]
print("No open store with no sales:", str(no_sales.size==0))
```

```
No closed store with sales: True
No open store with no sales: False
```

In [9]:

```
train.shape
```

Out[9]:

```
(1017209, 9)
```

let's take train data,for which Sales' values will be greater than Zero

In [10]:

```
train =  train.loc[train['Sales']>0]
```

Get the shap of train data now

In [11]:

```
print("New training Data shape",train.shape)
```

```
New training Data shape (844338, 9)
```

Data Visualization

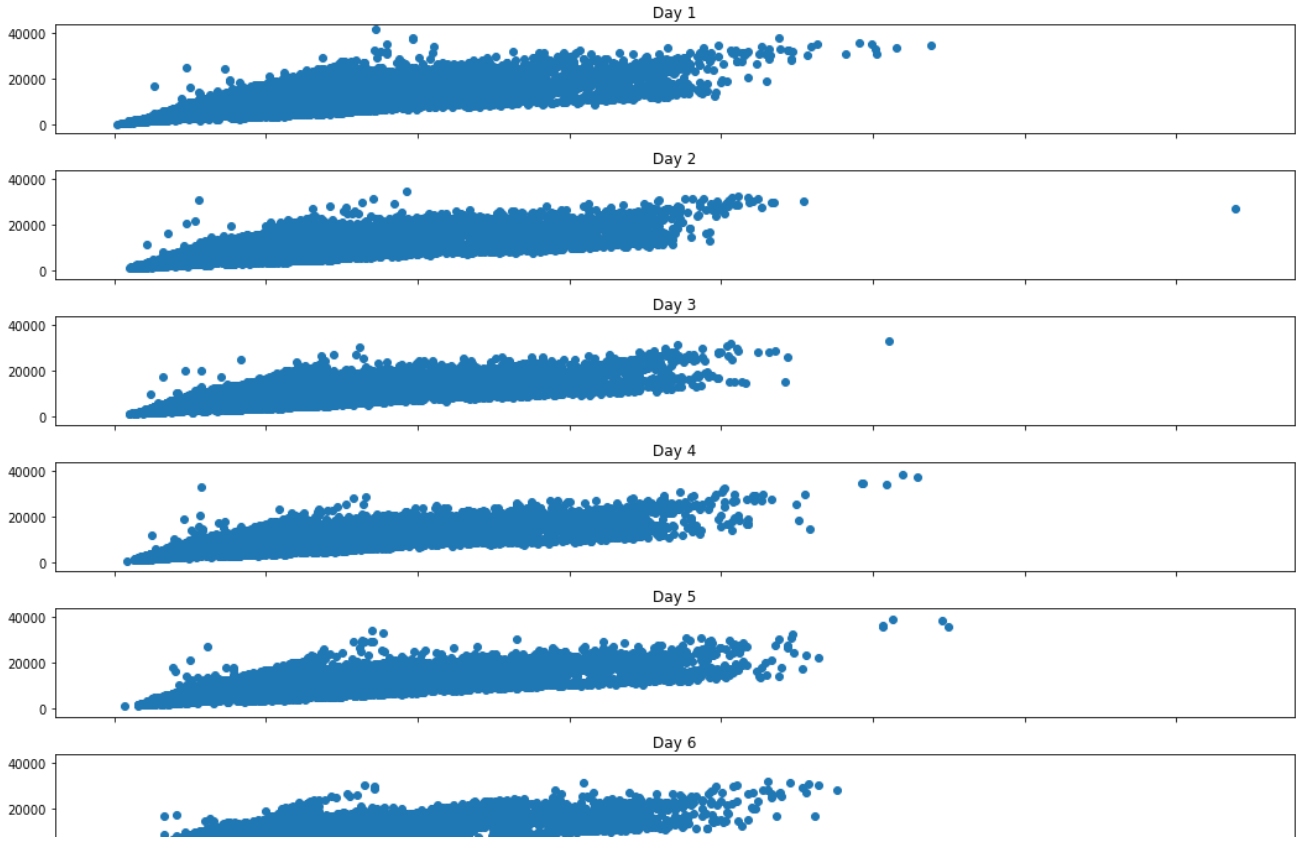
In [12]:

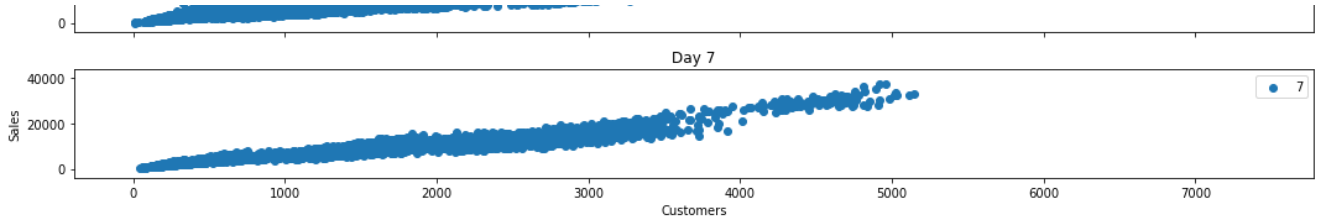
```
dates = pd.to_datetime(train['Date']).sort_values()
dates = dates.unique()
start_date = dates[0]
end_date = dates[-1]
print("start date",start_date)
print("end date",end_date)
date_range = pd.date_range(start_date,end_date).values
```

```
start date 2013-01-01T00:00:00.000000000
end date 2015-07-31T00:00:00.000000000
```

In [13]:

```
plt.rcParams['figure.figsize'] = [15.0,12.0]
f,ax = plt.subplots(7,sharex=True,sharey=True)
for i in range(1,8):
    data = train[train['DayOfWeek']==i]
    ax[i-1].set_title("Day {0}".format(i))
    ax[i-1].scatter(data['Customers'],data['Sales'],label=i)
plt.legend()
plt.xlabel("Customers")
plt.ylabel("Sales")
plt.tight_layout()
plt.show()
```





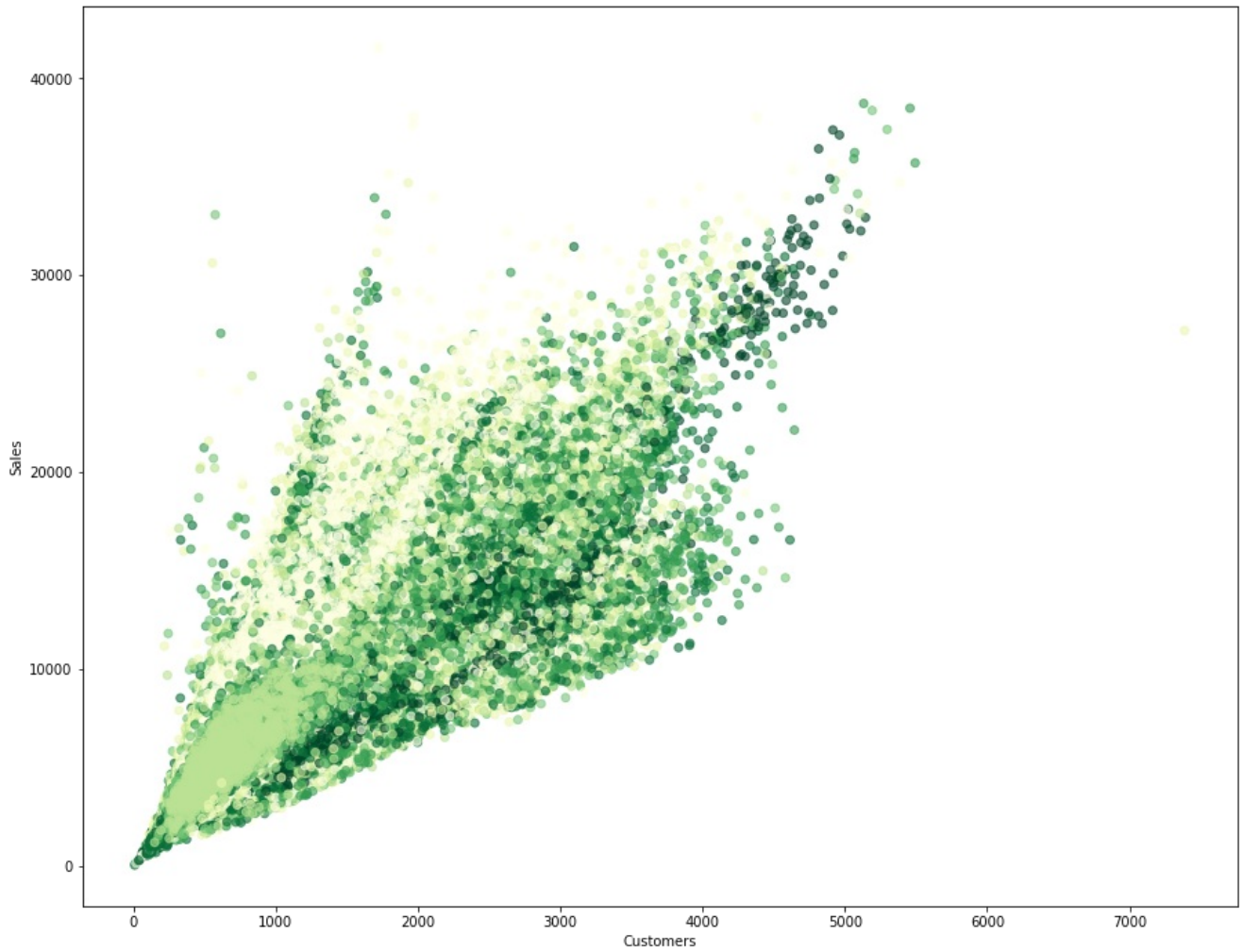
General Correction between customer and sales observed in above plot

In [14]:

```
#plotting customer Vs sales for each day of week

plt.scatter(train['Customers'],train['Sales'],c= train['DayOfWeek'],alpha=0.6,cmap = plt.cm.get_cma
p('YlGn'))

plt.xlabel('Customers')
plt.ylabel('Sales')
plt.show()
```



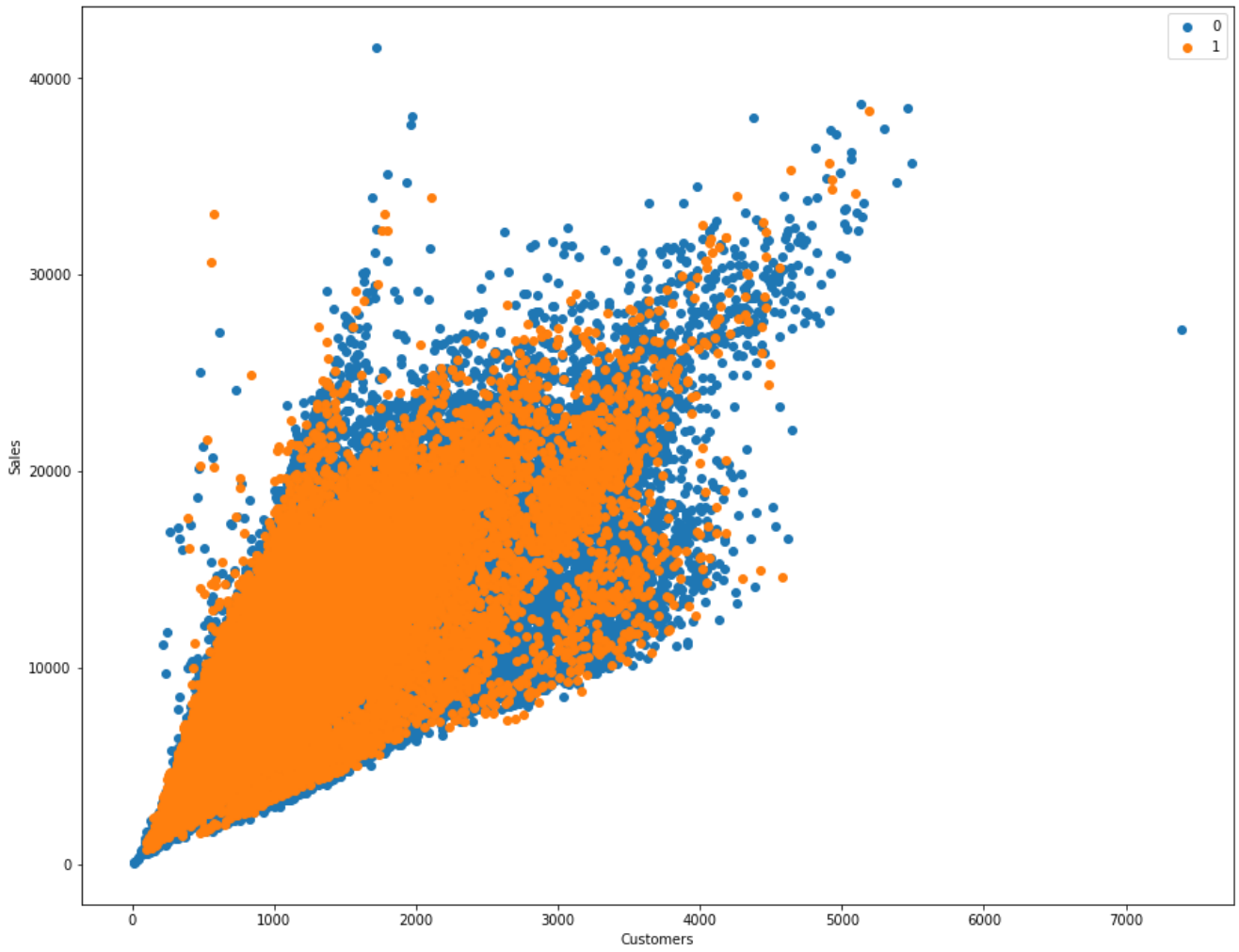
We visualize the data when we have school holiday

In [15]:

```
for i in [0,1]:
    data = train[train['SchoolHoliday']==i]
    if (len(data)==0):
        continue
    plt.scatter(data['Customers'],data['Sales'],label=i)

plt.legend()
plt.xlabel("Customers")
plt.ylabel("Sales")
```

```
plt.ylabel("Sales")
plt.show()
```



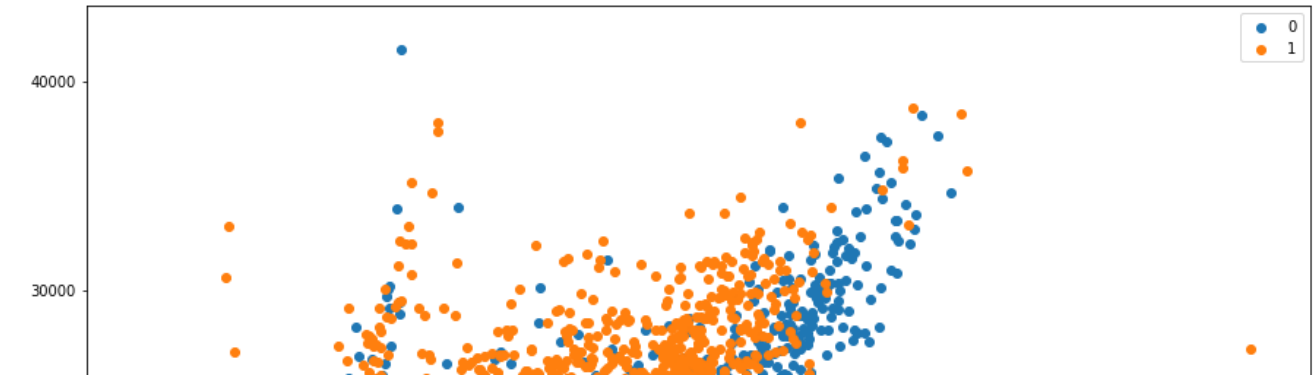
Observation : School holiday represents the orange point and sales represents the blue point . We can observe that school holiday is not much impacting on sales

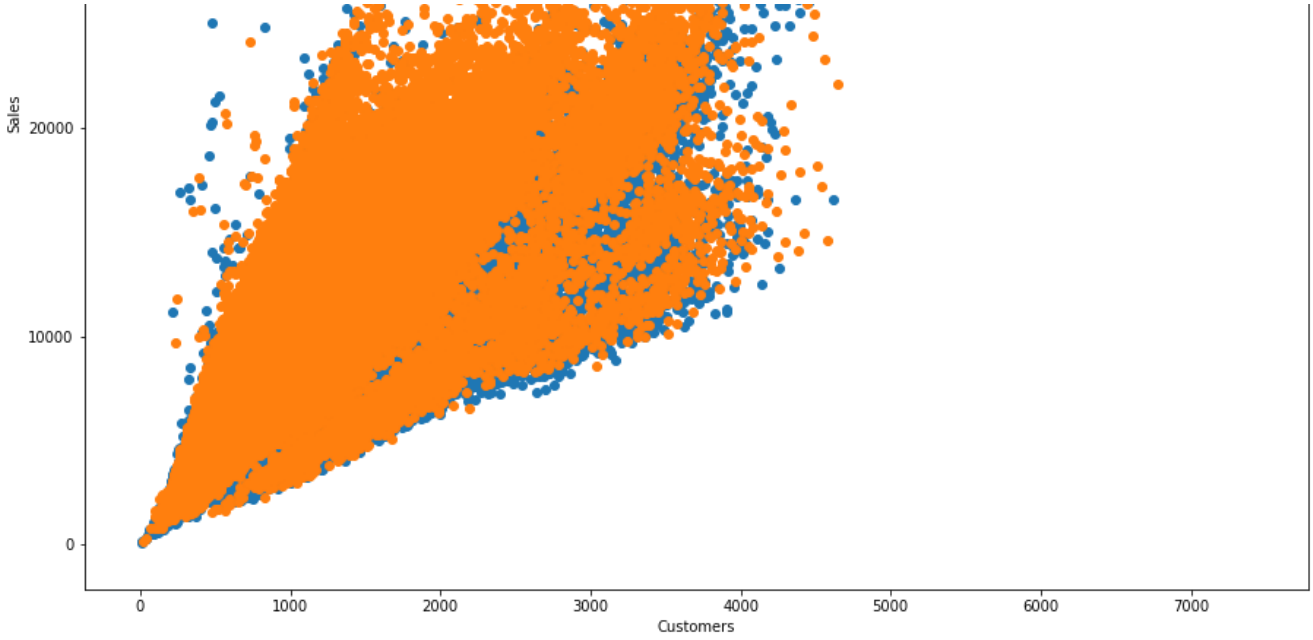
We visualize the data when we have Promo

In [16]:

```
for i in [0,1]:
    data = train[train['Promo']==i]
    if (len(data)==0):
        continue
    plt.scatter(data['Customers'],data['Sales'],label=i)

plt.legend()
plt.xlabel("Customers")
plt.ylabel("Sales")
plt.show()
```





Observation : Promo is impacting on Sales a lot

In [17]:

```
train.head()
```

Out[17]:

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

In [18]:

```
train['SalesPerCustomer'] = train['Sales']/train['Customers']

avg_store = train.groupby('Store')[['Sales','Customers','SalesPerCustomer']].mean()
avg_store.rename(columns=lambda x:'Avg'+x,inplace=True)
store = pd.merge(avg_store.reset_index(),store,on = 'Store')
store.head()
```

Out[18]:

	Store	AvgSales	AvgCustomers	AvgSalesPerCustomer	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth
0	1	4759.096031	564.049936	8.393038	c	a	1270.0	9.
1	2	4953.900510	583.998724	8.408443	a	a	570.0	11.
2	3	6942.568678	750.077022	9.117599	a	a	14130.0	12.
3	4	9638.401786	1321.752551	7.249827	c	c	620.0	9.
4	5	4676.274711	537.340180	8.611229	a	a	29910.0	4.

In [19]:

```
avg_store.head()
```

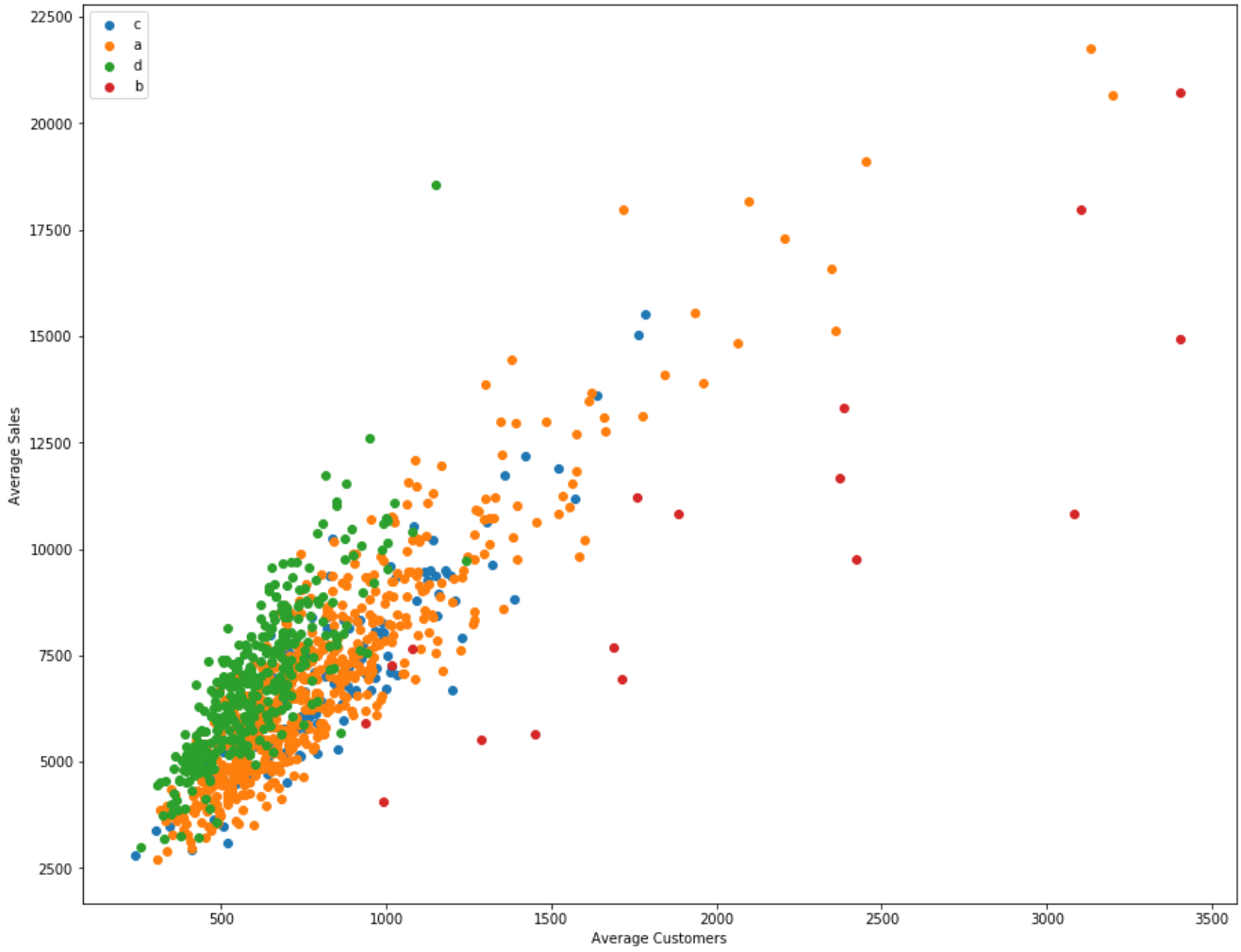
Out[19]:

	AvgSales	AvgCustomers	AvgSalesPerCustomer

Store	AvgSales	AvgCustomers	AvgSalesPerCustomer
1	4759.096031	564.049936	8.393038
2	4953.900510	583.998724	8.408443
3	6942.568678	750.077022	9.117599
4	9638.401786	1321.752551	7.249827
5	4676.274711	537.340180	8.611229

In [20]:

```
for i in store.StoreType.unique():
    data = store[store['StoreType']==i]
    if (len(data)==0):
        continue
    plt.scatter(data['AvgCustomers'],data['AvgSales'],label=i)
plt.legend()
plt.xlabel('Average Customers')
plt.ylabel('Average Sales')
plt.show()
```



In [21]:

```
store.Assortment.unique()
```

Out[21]:

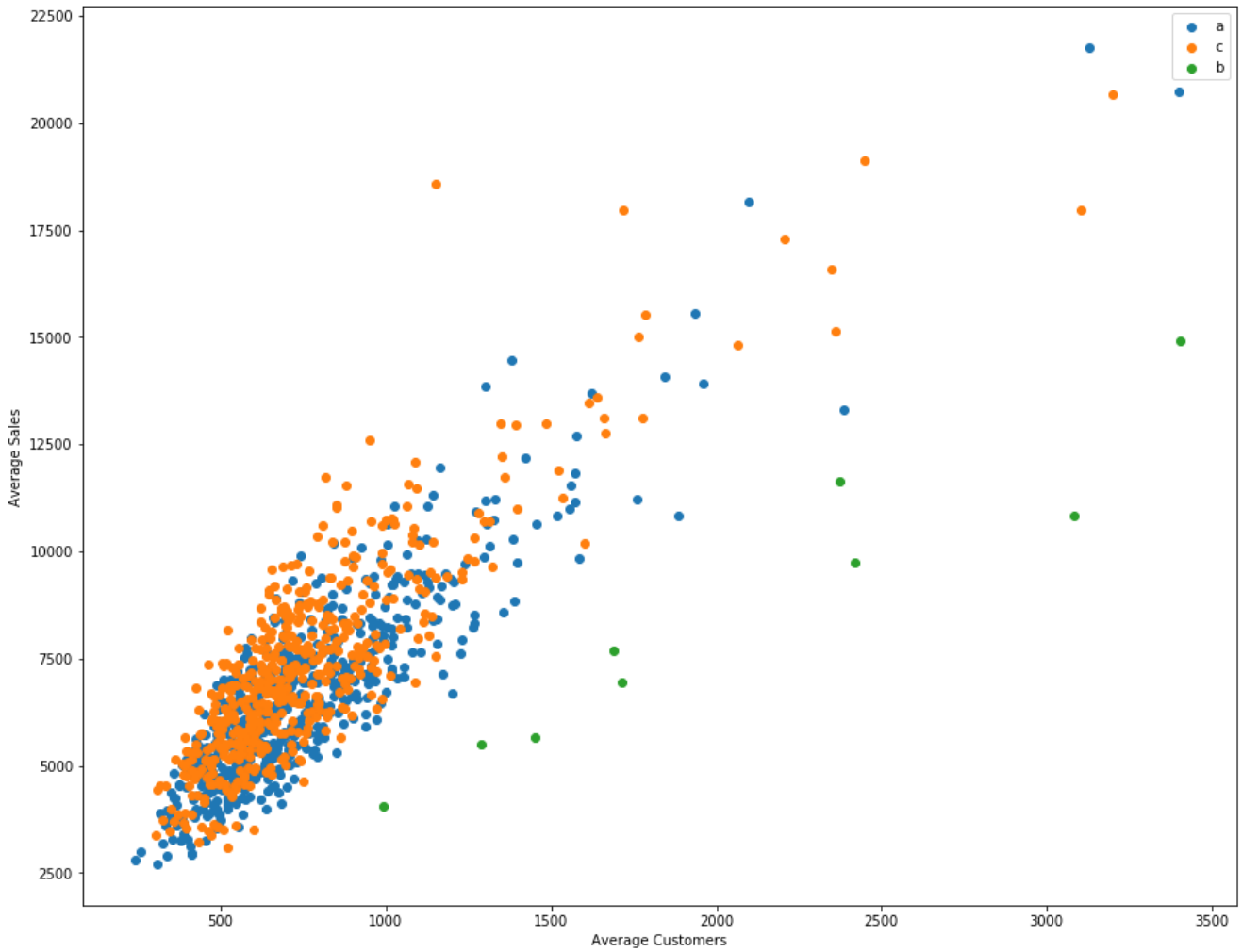
```
array(['a', 'c', 'b'], dtype=object)
```

In [22]:

```
for i in store.Assortment.unique():
```

```
    data = store[store['Assortment']==i]
    if (len(data)==0):
        continue
    plt.scatter(data['AvgCustomers'],data['AvgSales'],label=i)

plt.legend()
plt.xlabel('Average Customers')
plt.ylabel('Average Sales')
plt.show()
```



In [23]:

```
store.Promo2.unique()
```

Out[23]:

```
array([0, 1], dtype=int64)
```

In [24]:

```
for i in store.Promo2.unique():
    data = store[store['Promo2']==i]
    if (len(data)==0):
        continue
    plt.scatter(data['AvgCustomers'],data['AvgSales'],label=i)

plt.legend()
plt.xlabel('Avg Customers')
plt.ylabel('Avg Sales')
plt.show()
```

Feature Engineering

In [25]:

```
store.isnull().sum()
```

Out[25]:

```
Store                        0
AvgSales                     0
AvgCustomers                 0
AvgSalesPerCustomer          0
StoreType                    0
Assortment                   0
CompetitionDistance          3
CompetitionOpenSinceMonth  354
CompetitionOpenSinceYear   354
Promo2                       0
Promo2SinceWeek            544
Promo2SinceYear            544
PromoInterval              544
dtype: int64
```

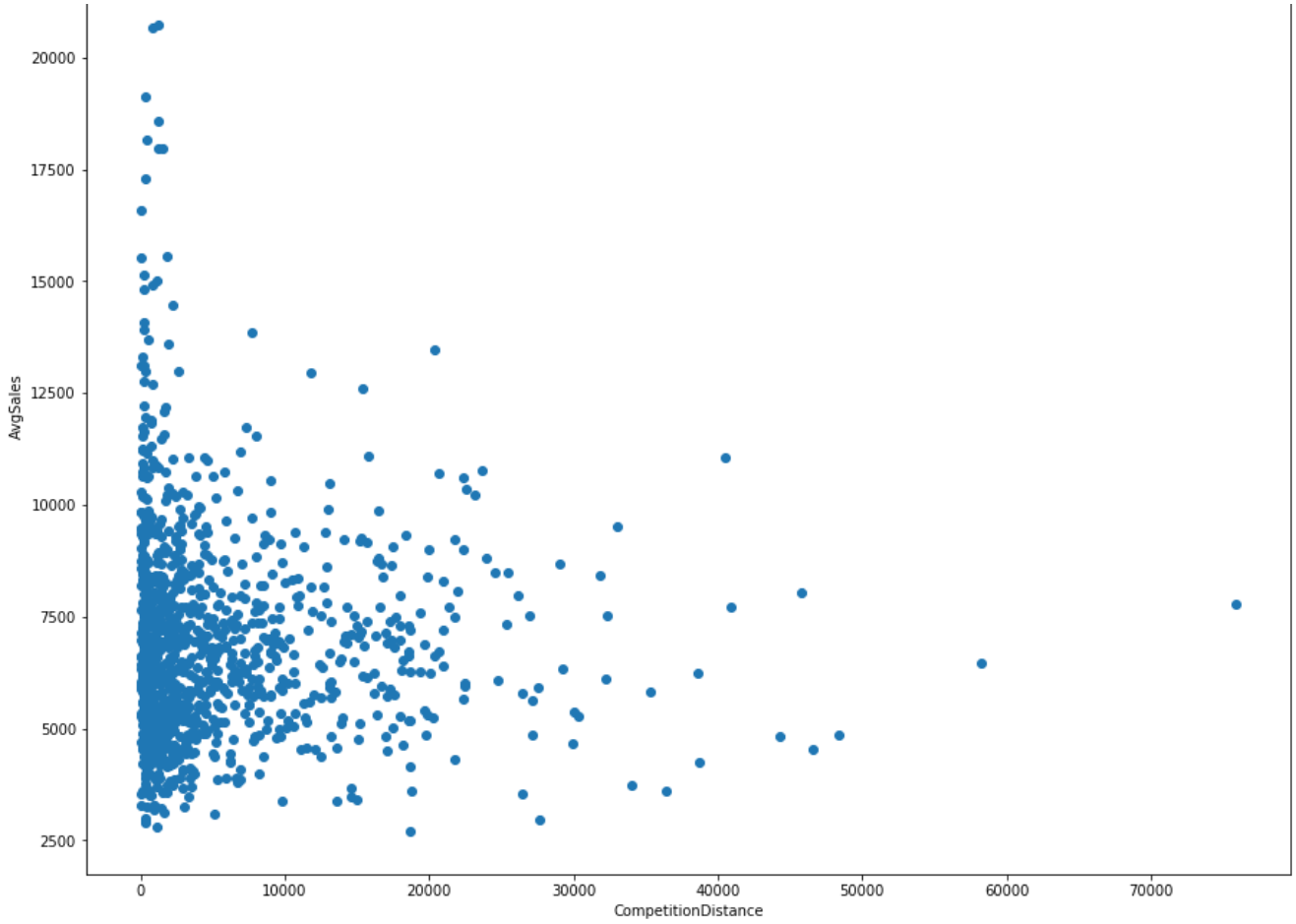
In [26]:

```
#Fill NaN values

store["CompetitionDistance"].fillna(-1)

plt.scatter(store['CompetitionDistance'],store['AvgSales'])

plt.xlabel('CompetitionDistance')
plt.ylabel('AvgSales')
plt.show()
```



Observation : If competition distance is nearby then AvgSales will be high

In [27]:

```
store.head()
```

Out[27]:

	Store	AvgSales	AvgCustomers	AvgSalesPerCustomer	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth
0	1	4759.096031	564.049936	8.393038	c	a	1270.0	9.
1	2	4953.900510	583.998724	8.408443	a	a	570.0	11.
2	3	6942.568678	750.077022	9.117599	a	a	14130.0	12.
3	4	9638.401786	1321.752551	7.249827	c	c	620.0	9.
4	5	4676.274711	537.340180	8.611229	a	a	29910.0	4.

In [28]:

```
store['StoreType'] = store['StoreType'].astype('category').cat.codes
store['Assortment'] = store['Assortment'].astype('category').cat.codes
train['StateHoliday'] = train['StateHoliday'].astype('category').cat.codes
store.head()
```

Out[28]:

	Store	AvgSales	AvgCustomers	AvgSalesPerCustomer	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth
0	1	4759.096031	564.049936	8.393038	2	0	1270.0	9.
1	2	4953.900510	583.998724	8.408443	0	0	570.0	11.
2	3	6942.568678	750.077022	9.117599	0	0	14130.0	12.
3	4	9638.401786	1321.752551	7.249827	2	2	620.0	9.
4	5	4676.274711	537.340180	8.611229	0	0	29910.0	4.

In [29]:

```
train.head()
```

Out[29]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	SalesPerCustomer
0	1	5	2015-07-31	5263	555	1	1	1	1	9.482883
1	2	5	2015-07-31	6064	625	1	1	1	1	9.702400
2	3	5	2015-07-31	8314	821	1	1	1	1	10.126675
3	4	5	2015-07-31	13995	1498	1	1	1	1	9.342457
4	5	5	2015-07-31	4822	559	1	1	1	1	8.626118

In [30]:

```
merged = pd.merge(train,store,on='Store',how='left')
merged.head()
```

Out[30]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	SalesPerCustomer	...	AvgSalesPerCusto
0	1	5	2015-07-31	5263	555	1	1	1	1	9.482883	...	8.393
1	2	5	2015-07-31	6064	625	1	1	1	1	9.702400	...	8.408
2	3	5	2015-07-31	8314	821	1	1	1	1	10.126675	...	9.117
3	4	5	2015-07-31	13995	1498	1	1	1	1	9.342457	...	7.249
4	5	5	2015-07-31	4822	559	1	1	1	1	8.626118	...	8.611

5 rows × 22 columns

In [31]:

```
merged.shape
```

Out[31]:

```
(844338, 22)
```

In [32]:

```
merged.isnull().sum()
```

Out[32]:

```
Store                         0
DayOfWeek                     0
Date                          0
Sales                         0
Customers                     0
Open                          0
Promo                         0
StateHoliday                  0
SchoolHoliday                 0
SalesPerCustomer              0
AvgSales                      0
AvgCustomers                  0
AvgSalesPerCustomer           0
StoreType                     0
Assortment                    0
CompetitionDistance        2186
CompetitionOpenSinceMonth  268600
```

```
CompetitionOpenSinceYear     268600
Promo2                            0
Promo2SinceWeek              423292
Promo2SinceYear              423292
PromoInterval                423292
dtype: int64
```

In [33]:

```
#remove NaNs

merged.fillna(0,inplace=True)
merged.isnull().sum()
```

Out[33]:

```
Store                        0
DayOfWeek                    0
Date                         0
Sales                        0
Customers                    0
Open                         0
Promo                        0
StateHoliday                 0
SchoolHoliday                0
SalesPerCustomer             0
AvgSales                     0
AvgCustomers                 0
AvgSalesPerCustomer          0
StoreType                    0
Assortment                   0
CompetitionDistance          0
CompetitionOpenSinceMonth    0
CompetitionOpenSinceYear     0
Promo2                       0
Promo2SinceWeek              0
Promo2SinceYear              0
PromoInterval                0
dtype: int64
```

Observation : We have removed the null value and filled with value "0"

In [34]:

```
merged['Date'] = pd.to_datetime(merged['Date'])
merged.dtypes
```

Out[34]:

```
Store                                int64
DayOfWeek                            int64
Date                        datetime64[ns]
Sales                                int64
Customers                            int64
Open                                 int64
Promo                                int64
StateHoliday                          int8
SchoolHoliday                        int64
SalesPerCustomer                   float64
AvgSales                           float64
AvgCustomers                       float64
AvgSalesPerCustomer                float64
StoreType                             int8
Assortment                            int8
CompetitionDistance                float64
CompetitionOpenSinceMonth          float64
CompetitionOpenSinceYear           float64
Promo2                               int64
Promo2SinceWeek                    float64
Promo2SinceYear                    float64
PromoInterval                       object
dtype: object
```

In [35]:

```
merged['Year'] = merged.Date.dt.year
merged['Month'] = merged.Date.dt.month
merged['Day'] = merged.Date.dt.day
merged['Week'] = merged.Date.dt.week
merged.head()
```

Out[35]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	SalesPerCustomer	...	CompetitionOpenS
0	1	5	2015-07-31	5263	555	1	1	1	1	9.482883	...	
1	2	5	2015-07-31	6064	625	1	1	1	1	9.702400	...	
2	3	5	2015-07-31	8314	821	1	1	1	1	10.126675	...	
3	4	5	2015-07-31	13995	1498	1	1	1	1	9.342457	...	
4	5	5	2015-07-31	4822	559	1	1	1	1	8.626118	...	

5 rows × 26 columns

In [36]:

```
#Number of months that competition has expired for
merged['MonthsCompetitionOpen'] = 12*(merged['Year']-merged['CompetitionOpenSinceYear'])+(merged['Month']-merged['CompetitionOpenSinceMonth'])
merged.loc[merged['CompetitionOpenSinceYear'] == 0,'MonthsCompetitionOpen']=0
merged.head()
```

Out[36]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	SalesPerCustomer	...	CompetitionOpenS
0	1	5	2015-07-31	5263	555	1	1	1	1	9.482883	...	
1	2	5	2015-07-31	6064	625	1	1	1	1	9.702400	...	
2	3	5	2015-07-31	8314	821	1	1	1	1	10.126675	...	
3	4	5	2015-07-31	13995	1498	1	1	1	1	9.342457	...	
4	5	5	2015-07-31	4822	559	1	1	1	1	8.626118	...	

5 rows × 27 columns

In [37]:

```
#Number of weeks that promotion has existed for

merged['WeekPromoOpen'] = 12*(merged['Year']-merged['Promo2SinceYear'])+(merged['Week']-merged['Promo2SinceWeek'])

merged.loc[merged['Promo2SinceWeek']==0,'WeekPromoOpen']=0
merged.head()
```

Out[37]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	SalesPerCustomer	...	Promo2	Promo2S
0	1	5	2015-07-31	5263	555	1	1	1	1	9.482883	...	0	
1	2	5	2015-07-31	6064	625	1	1	1	1	9.702400	...	1	
2	3	5	2015-										

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday	SalesPerCustomer	...	Promo2	Promo2S
2	3	5	07-31	8314	821	1	1	1	1	10.126675	...	1	
3	4	5	2015-07-31	13995	1498	1	1	1	1	9.342457	...	0	
4	5	5	2015-07-31	4822	559	1	1	1	1	8.626118	...	0	

5 rows × 28 columns

In [38]:

```
merged.dtypes
```

Out[38]:

```
Store                              int64
DayOfWeek                          int64
Date                      datetime64[ns]
Sales                              int64
Customers                          int64
Open                               int64
Promo                              int64
StateHoliday                        int8
SchoolHoliday                      int64
SalesPerCustomer                 float64
AvgSales                         float64
AvgCustomers                     float64
AvgSalesPerCustomer              float64
StoreType                           int8
Assortment                          int8
CompetitionDistance              float64
CompetitionOpenSinceMonth        float64
CompetitionOpenSinceYear         float64
Promo2                             int64
Promo2SinceWeek                  float64
Promo2SinceYear                  float64
PromoInterval                     object
Year                               int64
Month                              int64
Day                                int64
Week                               int64
MonthsCompetitionOpen            float64
WeekPromoOpen                    float64
dtype: object
```

In [39]:

```
toInt = [
         'CompetitionOpenSinceMonth',
          'CompetitionOpenSinceYear',
          'Promo2SinceWeek',
          'Promo2SinceYear',
          'MonthsCompetitionOpen',
          'WeekPromoOpen'

]
merged[toInt] = merged[toInt].astype(int)
```

In [40]:

```
merged.dtypes
```

Out[40]:

```
Store                              int64
DayOfWeek                          int64
Date                      datetime64[ns]
Sales                              int64
Customers                          int64
Open                               int64
Promo                              int64
StateHoliday                        int8
SchoolHoliday                      int64
SalesPerCustomer                 float64
```

```
SalesPerCustomer                      float64
AvgSales                              float64
AvgCustomers                          float64
AvgSalesPerCustomer                   float64
StoreType                                int8
Assortment                               int8
CompetitionDistance                   float64
CompetitionOpenSinceMonth               int32
CompetitionOpenSinceYear                int32
Promo2                                  int64
Promo2SinceWeek                         int32
Promo2SinceYear                         int32
PromoInterval                          object
Year                                    int64
Month                                   int64
Day                                     int64
Week                                    int64
MonthsCompetitionOpen                   int32
WeekPromoOpen                           int32
dtype: object
```

In [41]:

```
merged.dtypes
```

Out[41]:

```
Store                                 int64
DayOfWeek                             int64
Date                         datetime64[ns]
Sales                                 int64
Customers                             int64
Open                                  int64
Promo                                 int64
StateHoliday                           int8
SchoolHoliday                         int64
SalesPerCustomer                    float64
AvgSales                            float64
AvgCustomers                        float64
AvgSalesPerCustomer                 float64
StoreType                              int8
Assortment                             int8
CompetitionDistance                 float64
CompetitionOpenSinceMonth             int32
CompetitionOpenSinceYear              int32
Promo2                                int64
Promo2SinceWeek                       int32
Promo2SinceYear                       int32
PromoInterval                        object
Year                                  int64
Month                                 int64
Day                                   int64
Week                                  int64
MonthsCompetitionOpen                 int32
WeekPromoOpen                         int32
dtype: object
```

In [42]:

```
med_store = train.groupby('Store')[['Sales','Customers','SalesPerCustomer']].median()
med_store.rename(columns=lambda x : 'Med'+x,inplace=True)

store = pd.merge(med_store.reset_index(),store,on='Store')

store.head()
```

Out[42]:

	Store	MedSales	MedCustomers	MedSalesPerCustomer	AvgSales	AvgCustomers	AvgSalesPerCustomer	StoreType	Assortment
0	1	4647.0	550.0	8.362376	4759.096031	564.049936	8.393038	2	0
1	2	4783.0	575.5	8.313092	4953.900510	583.998724	8.408443	0	0
2	3	6619.0	744.0	9.123440	6942.568678	750.077022	9.117599	0	0

	Store	MedSales	MedCustomers	MedSalesPerCustomer	AvgSales	AvgCustomers	AvgSalesPerCustomer	StoreType	Assortment
3	4	9430.5	1301.5	7.245175	9638.404786	1321.752551	7.249827	2	2
4	5	4616.0	564.0	8.584677	4676.274711	537.340180	8.611229	0	0

In [43]:

```
merged = pd.merge(med_store.reset_index(),merged,on = 'Store')
merged.head()
```

Out[43]:

	Store	MedSales	MedCustomers	MedSalesPerCustomer	DayOfWeek	Date	Sales	Customers	Open	Promo	...	Promo2	Promo2
0	1	4647.0	550.0	8.362376	5	2015-07-31	5263	555	1	1	...	0	
1	1	4647.0	550.0	8.362376	4	2015-07-30	5020	546	1	1	...	0	
2	1	4647.0	550.0	8.362376	3	2015-07-29	4782	523	1	1	...	0	
3	1	4647.0	550.0	8.362376	2	2015-07-28	5011	560	1	1	...	0	
4	1	4647.0	550.0	8.362376	1	2015-07-27	6102	612	1	1	...	0	

5 rows × 31 columns

In [44]:

```
merged.columns
```

Out[44]:

```
Index(['Store', 'MedSales', 'MedCustomers', 'MedSalesPerCustomer', 'DayOfWeek',
       'Date', 'Sales', 'Customers', 'Open', 'Promo', 'StateHoliday',
       'SchoolHoliday', 'SalesPerCustomer', 'AvgSales', 'AvgCustomers',
       'AvgSalesPerCustomer', 'StoreType', 'Assortment', 'CompetitionDistance',
       'CompetitionOpenSinceMonth', 'CompetitionOpenSinceYear', 'Promo2',
       'Promo2SinceWeek', 'Promo2SinceYear', 'PromoInterval', 'Year', 'Month',
       'Day', 'Week', 'MonthsCompetitionOpen', 'WeekPromoOpen'],
      dtype='object')
```

In [45]:

```
merged.dtypes
```

Out[45]:

```
Store                              int64
MedSales                         float64
MedCustomers                     float64
MedSalesPerCustomer              float64
DayOfWeek                          int64
Date                      datetime64[ns]
Sales                              int64
Customers                          int64
Open                               int64
Promo                              int64
StateHoliday                        int8
SchoolHoliday                      int64
SalesPerCustomer                 float64
AvgSales                         float64
AvgCustomers                     float64
AvgSalesPerCustomer              float64
StoreType                           int8
Assortment                          int8
CompetitionDistance              float64
CompetitionOpenSinceMonth          int32
CompetitionOpenSinceYear           int32
Promo2                             int64
Promo2SinceWeek                    int32
Promo2SinceYear                    int32
PromoInterval                     object
```

```
PromoInterval                    object
Year                              int64
Month                             int64
Day                               int64
Week                              int64
MonthsCompetitionOpen             int32
WeekPromoOpen                     int32
dtype: object
```

In [46]:

```
merged.hist(figsize=(20,20))
plt.show()
```

In [47]:

```
X = [
    'Store',
    'Customers',
    'CompetitionDistance',

    'Promo',
    'Promo2',

    'CompetitionOpenSinceMonth',
    'CompetitionOpenSinceYear',
    'Promo2SinceWeek',
```

```
    'Promo2SinceYear',


    'StateHoliday',
    'StoreType',
    'Assortment',

    'AvgSales',
    'AvgCustomers',
    'AvgSalesPerCustomer',

    'MedSales',
    'MedCustomers',
    'MedSalesPerCustomer',

    'DayOfWeek',
    'Week',
    'Day',
    'Month',
    'Year',

]
```

In [48]:

```
merged[X].head()
```

Out[48]:

	Store	Customers	CompetitionDistance	Promo	Promo2	CompetitionOpenSinceMonth	CompetitionOpenSinceYear	Promo2SinceWee
0	1	555	1270.0	1	0	9	2008	
1	1	546	1270.0	1	0	9	2008	
2	1	523	1270.0	1	0	9	2008	
3	1	560	1270.0	1	0	9	2008	
4	1	612	1270.0	1	0	9	2008	

5 rows × 23 columns

In [49]:

```
X_data = merged[X]
Y_data = np.log(merged['Sales'])
Y_data
```

Out[49]:

```
0         8.568456
1         8.521185
2         8.472614
3         8.519391
4         8.716372
              ...
844333    8.840001
844334    8.470311
844335    8.420682
844336    8.365672
844337    8.215277
Name: Sales, Length: 844338, dtype: float64
```

In [50]:

```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test = train_test_split(X_data,Y_data,test_size=0.20,random_state=10)
```

In [51]:

```
X_train.shape
```

Out[51]:

(675470, 23)

In [52]:

```
Y_train.shape
```

Out[52]:

(675470,)

In [53]:

```
X_test.shape
```

Out[53]:

(168868, 23)

In [54]:

```
Y_test.shape
```

Out[54]:

(168868,)

In [55]:

```
import xgboost as xgb
from sklearn.model_selection import GridSearchCV

param ={
            'n_estimators': [10,32,45,67],
            'max_depth':[2,4,6,8]
        }

xgboost_tree = xgb.XGBRegressor(
    eta = 0.1,
    min_child_weight = 2,
    subsample = 0.8,
    colsample_bytree = 0.8,
    tree_method = 'exact',
    reg_alpha = 0.05,
    silent = 0,
    random_state = 1023
)

grid = GridSearchCV(estimator=xgboost_tree,param_grid= param,cv=5,verbose=1, n_jobs=-1,scoring='neg
_mean_squared_error')



grid_result = grid.fit(X_train, Y_train)
best_params = grid_result.best_params_

print('Best Params :',best_params)
```

```
Fitting 5 folds for each of 16 candidates, totalling 80 fits
[13:31:47] WARNING: C:/Jenkins/workspace/xgboost-
win64_release_0.90/src/objective/regression_obj.cu:152: reg:linear is now deprecated in favor of
reg:squarederror.
Best Params : {'max_depth': 8, 'n_estimators': 67}
```

We got best parameters max_depth = 8 and n_estimators =67

In [56]:

```
from math import sqrt
```

```
from sklearn.metrics import mean_squared_error
pred = grid_result.predict(X_test)

print("Root Mean Squared error {}".format(sqrt(mean_squared_error(np.exp(Y_test),np.exp(pred)))))
```

```
Root Mean Squared error 437.55307913181764
```

In [57]:

```
import sklearn
sorted(sklearn.metrics.SCORERS.keys())
```

Out[57]:

```
['accuracy',
 'adjusted_mutual_info_score',
 'adjusted_rand_score',
 'average_precision',
 'balanced_accuracy',
 'completeness_score',
 'explained_variance',
 'f1',
 'f1_macro',
 'f1_micro',
 'f1_samples',
 'f1_weighted',
 'fowlkes_mallows_score',
 'homogeneity_score',
 'jaccard',
 'jaccard_macro',
 'jaccard_micro',
 'jaccard_samples',
 'jaccard_weighted',
 'max_error',
 'mutual_info_score',
 'neg_brier_score',
 'neg_log_loss',
 'neg_mean_absolute_error',
 'neg_mean_absolute_percentage_error',
 'neg_mean_gamma_deviance',
 'neg_mean_poisson_deviance',
 'neg_mean_squared_error',
 'neg_mean_squared_log_error',
 'neg_median_absolute_error',
 'neg_root_mean_squared_error',
 'normalized_mutual_info_score',
 'precision',
 'precision_macro',
 'precision_micro',
 'precision_samples',
 'precision_weighted',
 'r2',
 'rand_score',
 'recall',
 'recall_macro',
 'recall_micro',
 'recall_samples',
 'recall_weighted',
 'roc_auc',
 'roc_auc_ovo',
 'roc_auc_ovo_weighted',
 'roc_auc_ovr',
 'roc_auc_ovr_weighted',
 'top_k_accuracy',
 'v_measure_score']
```

In [ ]:

```

```