Data Preprocessing

Now, we are done with exploratory data analysis of both training and testing datasets. Now, we should get into preprocessing for both the datasets as some of the features are not numerical.

Importing all packages

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import *
from sklearn.linear_model import *
from math import *
from sklearn.ensemble import *
from sklearn.feature_selection import *
from sklearn.feature_extraction import *
from sklearn.naive_bayes import *
from sklearn.discriminant_analysis import *
from sklearn.preprocessing import *
from sklearn.metrics import *
from sklearn.neighbors import *
from sklearn.cluster import *
```

In [2]:
```
df_train = pd.read_csv("train_eda.csv")
df_test = pd.read_csv("test_eda.csv")
```

In [3]:
```
df_train.head()
```

Out[3]:

	id	date	store_nbr	family	onpromotion	city	state	type	cluster	dcoilwti
0	0	2013-01-01	1	AUTOMOTIVE	0	Quito	Pichincha	D	13	93.3(
1	1	2013-01-01	1	BABY CARE	0	Quito	Pichincha	D	13	93.3(
2	2	2013-01-01	1	BEAUTY	0	Quito	Pichincha	D	13	93.3(
3	3	2013-01-01	1	BEVERAGES	0	Quito	Pichincha	D	13	93.3(
4	4	2013-01-01	1	BOOKS	0	Quito	Pichincha	D	13	93.3(

In [4]:
```
df_test.head()
```

Out[4]:

	id	date	store_nbr	family	onpromotion	city	state	type	cluster	d
0	3000888	2017-08-16	1	AUTOMOTIVE	0	Quito	Pichincha	D	13	
1	3000889	2017-08-16	1	BABY CARE	0	Quito	Pichincha	D	13	
2	3000890	2017-08-16	1	BEAUTY	2	Quito	Pichincha	D	13	
3	3000891	2017-08-16	1	BEVERAGES	20	Quito	Pichincha	D	13	
4	3000892	2017-08-16	1	BOOKS	0	Quito	Pichincha	D	13	

In [5]:
```
print("Length of training dataset : ",len(df_train))
print("Length of testing dataset  : ",len(df_test))
```

```
Length of training dataset :  1048575
Length of testing dataset  :  28512
```

Description on both training and testing datasets

In [6]:
```
df_train.describe().round(2)
```

Out[6]:

	id	store_nbr	onpromotion	cluster	dcoilwtico	holiday?	sale
count	1048575.00	1048575.00	1048575.00	1048575.00	1048575.00	1048575.00	1048575.0
mean	524287.00	27.49	0.11	8.48	99.19	0.13	244.5
std	302697.67	15.58	2.38	4.65	4.31	0.34	806.5
min	0.00	1.00	0.00	1.00	86.65	0.00	0.0
25%	262143.50	14.00	0.00	4.00	96.29	0.00	0.0
50%	524287.00	27.00	0.00	9.00	99.19	0.00	1.0
75%	786430.50	41.00	0.00	13.00	101.92	0.00	120.0
max	1048574.00	54.00	196.00	17.00	110.62	1.00	46271.0

In [7]:
```
df_test.describe().round(2)
```

Out[7]:

	id	store_nbr	onpromotion	cluster	dcoilwtico	holiday?
count	28512.00	28512.00	28512.00	28512.00	28512.00	28512.00
mean	3015143.50	27.50	6.97	8.48	47.24	0.06
std	8230.85	15.59	20.68	4.65	0.65	0.24
min	3000888.00	1.00	0.00	1.00	45.96	0.00
25%	3008015.75	14.00	0.00	4.00	47.00	0.00
50%	3015143.50	27.50	0.00	8.50	47.24	0.00
75%	3022271.25	41.00	6.00	13.00	47.46	0.00
max	3029399.00	54.00	646.00	17.00	48.59	1.00

Information about both training and testing datasets

In [8]:
```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1048575 entries, 0 to 1048574
Data columns (total 12 columns):
 #   Column       Non-Null Count    Dtype  
---  ------       --------------    -----  
 0   id           1048575 non-null  int64  
 1   date         1048575 non-null  object 
 2   store_nbr    1048575 non-null  int64  
 3   family       1048575 non-null  object 
 4   onpromotion  1048575 non-null  int64  
 5   city         1048575 non-null  object 
 6   state        1048575 non-null  object 
 7   type         1048575 non-null  object 
 8   cluster      1048575 non-null  int64  
 9   dcoilwtico   1048575 non-null  float64
 10  holiday?     1048575 non-null  float64
 11  sales        1048575 non-null  float64
dtypes: float64(3), int64(4), object(5)
memory usage: 96.0+ MB
```

In [9]:
```
df_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 28512 entries, 0 to 28511
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   id           28512 non-null  int64  
 1   date         28512 non-null  object 
 2   store_nbr    28512 non-null  int64  
 3   family       28512 non-null  object 
 4   onpromotion  28512 non-null  int64  
 5   city         28512 non-null  object 
 6   state        28512 non-null  object 
 7   type         28512 non-null  object 
 8   cluster      28512 non-null  int64  
 9   dcoilwtico   28512 non-null  float64
 10  holiday?     28512 non-null  float64
dtypes: float64(2), int64(4), object(5)
memory usage: 2.4+ MB
```

Conversion of date from object type to datetime[64ns] type

In [10]:
```
df_train["date"] = pd.to_datetime(df_train["date"])
df_test["date"] = pd.to_datetime(df_test["date"])
```

Checking for "store" column

In [11]:
```
store_num_train = df_train["store_nbr"].unique()
store_num_train.sort()
store_num_train
```

Out[11]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
       35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51,
       52, 53, 54])
```

In [12]:
```
store_num_test = df_test["store_nbr"].unique()
store_num_test.sort()
store_num_test
```

Out[12]:
```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16, 17,
       18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34,
       35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51,
       52, 53, 54])
```

Subtracting each element of store numbers for convenience

In [13]:
```
df_train_1 = df_train.copy()
df_test_1 = df_test.copy()
df_train_1["store_nbr"] = df_train_1["store_nbr"] - 1
df_test_1["store_nbr"] = df_test_1["store_nbr"] - 1
```

Binary encoding function

In [14]:
```
def dec_to_bin(num):
    n = num
    st = []
    while n > 0:
        r = int(n % 2)
        n = int(n / 2)
        st.append(r)
    fin = st[::-1]
    return fin
```

In [15]:
```
def bin_to_dec(num):
    n = num[::-1]
    st = 0
    ctr = 0
    for i in n:
        st += i * pow(2,ctr)
        ctr += 1
    return int(st)
```

In [16]:
```
def max_len_bin(x, lt):
    ctr = len(x)
    if ctr < lt:
        u = np.zeros(lt - ctr)
        v = u.astype("int")
```

```
        t = list(v)
        f = t + x
    else:
        f = x
    return f
```

In [17]:
```
def column_names(nam, col_num):
    lis = []
    for i in range(0, col_num):
        t = nam + "_" + str(i)
        lis.append(t)
    return lis
```

In [18]:
```
def binary_encoder(df,col_name):
    bin_convert = lambda x: dec_to_bin(x)
    bin_num = list(map(bin_convert,df[col_name]))
    bin_len = lambda x: len(x)
    max_bin = max(list(map(bin_len,bin_num)))
    equal_len = lambda x: max_len_bin(x,max_bin)
    equal_ele = list(map(equal_len,bin_num))
    cols = column_names(col_name,max_bin)
    df_bin = pd.DataFrame(equal_ele,columns=cols)
    return df_bin
```

In [19]:
```
binary_encoder(df_train_1,"store_nbr").head()
```

Out[19]:

	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0

Performing binary encoding for "store_nbr" column

In [20]:
```
store_num_train = binary_encoder(df_train_1,"store_nbr")
store_num_test = binary_encoder(df_test_1,"store_nbr")
```

In [21]:
```
store_num_train.head()
```

Out[21]:

	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0

In [22]:
```
store_num_test.head()
```

Out[22]:

	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	0	0	0	0	0
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	0

In [23]:
```
print("Number of rows in training set : ",len(store_num_train))
print("Number of rows in testing set  : ",len(store_num_test))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

In [24]:
```
df_train_2 = df_train_1.copy()
df_test_2 = df_test_1.copy()
df_train_2[store_num_train.columns.values] = store_num_train
df_test_2[store_num_test.columns.values] = store_num_test
df_train_2.drop("store_nbr",axis=1,inplace=True)
df_test_2.drop("store_nbr",axis=1,inplace=True)
df_train_2 = df_train_2[["id","date","store_nbr_0","store_nbr_1","store_nbr_
df_test_2 = df_test_2[["id","date","store_nbr_0","store_nbr_1","store_nbr_2"
```

In [25]:
```
df_train_2.head()
```

Out[25]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	2013-01-01	0	0	0	0	0	0
1	1	2013-01-01	0	0	0	0	0	0
2	2	2013-01-01	0	0	0	0	0	0
3	3	2013-01-01	0	0	0	0	0	0
4	4	2013-01-01	0	0	0	0	0	0

In [26]:
```
df_test_2.head()
```

Out[26]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_
0	3000888	2017-08-16	0	0	0	0	0	
1	3000889	2017-08-16	0	0	0	0	0	
2	3000890	2017-08-16	0	0	0	0	0	
3	3000891	2017-08-16	0	0	0	0	0	
4	3000892	2017-08-16	0	0	0	0	0	

In [27]:
```
print("Number of rows in training set : ",len(df_train_2))
print("Number of rows in testing set  : ",len(df_test_2))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Checking for "family" column

In [28]:
```
df_train_2["family"].unique()
```

Out[28]:
```
array(['AUTOMOTIVE', 'BABY CARE', 'BEAUTY', 'BEVERAGES', 'BOOKS',
       'BREAD/BAKERY', 'CELEBRATION', 'CLEANING', 'DAIRY', 'DELI', 'EGGS',
       'FROZEN FOODS', 'GROCERY I', 'GROCERY II', 'HARDWARE',
       'HOME AND KITCHEN I', 'HOME AND KITCHEN II', 'HOME APPLIANCES',
       'HOME CARE', 'LADIESWEAR', 'LAWN AND GARDEN', 'LINGERIE',
       'LIQUOR,WINE,BEER', 'MAGAZINES', 'MEATS', 'PERSONAL CARE',
       'PET SUPPLIES', 'PLAYERS AND ELECTRONICS', 'POULTRY',
       'PREPARED FOODS', 'PRODUCE', 'SCHOOL AND OFFICE SUPPLIES',
       'SEAFOOD'], dtype=object)
```

In [29]:
```
df_test_2["family"].unique()
```

Out[29]:
```
array(['AUTOMOTIVE', 'BABY CARE', 'BEAUTY', 'BEVERAGES', 'BOOKS',
       'BREAD/BAKERY', 'CELEBRATION', 'CLEANING', 'DAIRY', 'DELI', 'EGGS',
       'FROZEN FOODS', 'GROCERY I', 'GROCERY II', 'HARDWARE',
       'HOME AND KITCHEN I', 'HOME AND KITCHEN II', 'HOME APPLIANCES',
       'HOME CARE', 'LADIESWEAR', 'LAWN AND GARDEN', 'LINGERIE',
       'LIQUOR,WINE,BEER', 'MAGAZINES', 'MEATS', 'PERSONAL CARE',
       'PET SUPPLIES', 'PLAYERS AND ELECTRONICS', 'POULTRY',
       'PREPARED FOODS', 'PRODUCE', 'SCHOOL AND OFFICE SUPPLIES',
       'SEAFOOD'], dtype=object)
```

Mapping family elements to numbers

In [30]:
```
ctr = 0
diction = dict()
for i in df_train_2["family"].unique():
    diction[i] = ctr
    ctr += 1
diction
```

Out[30]:
```
{'AUTOMOTIVE': 0,
 'BABY CARE': 1,
 'BEAUTY': 2,
 'BEVERAGES': 3,
 'BOOKS': 4,
 'BREAD/BAKERY': 5,
 'CELEBRATION': 6,
 'CLEANING': 7,
 'DAIRY': 8,
 'DELI': 9,
 'EGGS': 10,
 'FROZEN FOODS': 11,
 'GROCERY I': 12,
 'GROCERY II': 13,
 'HARDWARE': 14,
 'HOME AND KITCHEN I': 15,
 'HOME AND KITCHEN II': 16,
 'HOME APPLIANCES': 17,
 'HOME CARE': 18,
 'LADIESWEAR': 19,
 'LAWN AND GARDEN': 20,
 'LINGERIE': 21,
 'LIQUOR,WINE,BEER': 22,
 'MAGAZINES': 23,
 'MEATS': 24,
 'PERSONAL CARE': 25,
 'PET SUPPLIES': 26,
 'PLAYERS AND ELECTRONICS': 27,
 'POULTRY': 28,
 'PREPARED FOODS': 29,
 'PRODUCE': 30,
 'SCHOOL AND OFFICE SUPPLIES': 31,
 'SEAFOOD': 32}
```

In [31]:
```
fam_train = pd.DataFrame(df_train_2["family"].map(diction))
fam_test = pd.DataFrame(df_test_2["family"].map(diction))
```

In [32]:
```
fam_train.head()
```

Out[32]:

	family
0	0
1	1
2	2
3	3
4	4

In [33]:
```
fam_test.head()
```

Out[33]:

	family
0	0
1	1
2	2
3	3
4	4

In [34]:
```
print("Number of rows in training set : ",len(fam_train))
print("Number of rows in testing set  : ",len(fam_test))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Performing binary encoding on "family" column

In [35]:
```
family_train = binary_encoder(fam_train,"family")
family_test = binary_encoder(fam_test,"family")
```

In [36]:
```
family_train.head()
```

Out[36]:

	family_0	family_1	family_2	family_3	family_4	family_5
0	0	0	0	0	0	0
1	0	0	0	0	0	1
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	0	0	1	0	0

In [37]:
```
family_test.head()
```

Out[37]:

	family_0	family_1	family_2	family_3	family_4	family_5
0	0	0	0	0	0	0
1	0	0	0	0	0	1
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	0	0	1	0	0

In [38]:
```
df_train_3 = df_train_2.copy()
df_test_3 = df_test_2.copy()
df_train_3[family_train.columns.values] = family_train
df_test_3[family_test.columns.values] = family_test
df_train_3.drop("family",axis=1,inplace=True)
df_test_3.drop("family",axis=1,inplace=True)
df_train_3 = df_train_3[["id","date","store_nbr_0","store_nbr_1","store_nbr_
df_test_3 = df_test_3[["id","date","store_nbr_0","store_nbr_1","store_nbr_2"
```

In [39]:
```
df_train_3.head()
```

Out[39]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	2013-01-01	0	0	0	0	0	0
1	1	2013-01-01	0	0	0	0	0	0
2	2	2013-01-01	0	0	0	0	0	0
3	3	2013-01-01	0	0	0	0	0	0
4	4	2013-01-01	0	0	0	0	0	0

5 rows × 22 columns

In [40]:
```
df_test_3.head()
```

Out[40]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_
0	3000888	2017-08-16	0	0	0	0	0	
1	3000889	2017-08-16	0	0	0	0	0	
2	3000890	2017-08-16	0	0	0	0	0	
3	3000891	2017-08-16	0	0	0	0	0	
4	3000892	2017-08-16	0	0	0	0	0	

5 rows × 21 columns

In [41]:
```
print("Number of rows in training set : ",len(df_train_3))
print("Number of rows in testing set  : ",len(df_test_3))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Checking for "city" column

In [42]:
```
df_train_3["city"].unique()
```

Out[42]:
```
array(['Quito', 'Cayambe', 'Latacunga', 'Riobamba', 'Ibarra',
       'Santo Domingo', 'Guaranda', 'Puyo', 'Ambato', 'Guayaquil',
       'Salinas', 'Daule', 'Babahoyo', 'Quevedo', 'Playas', 'Libertad',
       'Cuenca', 'Loja', 'Machala', 'Esmeraldas', 'Manta', 'El Carmen'],
      dtype=object)
```

In [43]:
```
df_test_3["city"].unique()
```

Out[43]:
```
array(['Quito', 'Cayambe', 'Latacunga', 'Riobamba', 'Ibarra',
       'Santo Domingo', 'Guaranda', 'Puyo', 'Ambato', 'Guayaquil',
       'Salinas', 'Daule', 'Babahoyo', 'Quevedo', 'Playas', 'Libertad',
       'Cuenca', 'Loja', 'Machala', 'Esmeraldas', 'Manta', 'El Carmen'],
      dtype=object)
```

In [44]:
```
print("Number of rows in training set : ",len(df_train_3["city"]))
print("Number of rows in testing set  : ",len(df_test_3["city"]))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Mapping city elements to numbers

In [45]:
```
city = dict()
ctr = 0
for i in df_train_3["city"].unique():
    city[i] = ctr
    ctr +=1
city
```

Out[45]:
```
{'Quito': 0,
 'Cayambe': 1,
 'Latacunga': 2,
 'Riobamba': 3,
 'Ibarra': 4,
 'Santo Domingo': 5,
 'Guaranda': 6,
 'Puyo': 7,
 'Ambato': 8,
 'Guayaquil': 9,
 'Salinas': 10,
 'Daule': 11,
 'Babahoyo': 12,
 'Quevedo': 13,
 'Playas': 14,
 'Libertad': 15,
 'Cuenca': 16,
 'Loja': 17,
 'Machala': 18,
 'Esmeraldas': 19,
 'Manta': 20,
 'El Carmen': 21}
```

In [46]:
```
df_train_4 = df_train_3.copy()
df_test_4 = df_test_3.copy()
city_train = pd.DataFrame(df_train_4["city"].map(city))
city_test = pd.DataFrame(df_test_4["city"].map(city))
```

In [47]:
```
print("Number of rows in training set : ",len(city_train))
print("Number of rows in testing set  : ",len(city_test))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Performing binary encoding on "city" column

In [48]:
```
city_bin_train = binary_encoder(city_train,"city")
city_bin_test = binary_encoder(city_test,"city")
```

In [49]:
```
city_bin_train.head()
```

Out[49]:

	city_0	city_1	city_2	city_3	city_4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

In [50]:
```
city_bin_test.head()
```

Out[50]:

	city_0	city_1	city_2	city_3	city_4
0	0	0	0	0	0
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0

In [51]:
```
print("Number of rows in training set : ",len(city_bin_train))
print("Number of rows in testing set  : ",len(city_bin_test))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

In [52]:
```
df_train_5 = df_train_4.copy()
df_test_5 = df_test_4.copy()
df_train_5[city_bin_train.columns.values] = city_bin_train
df_test_5[city_bin_test.columns.values] = city_bin_test
df_train_5.drop("city",axis=1,inplace=True)
df_test_5.drop("city",axis=1,inplace=True)
df_train_5 = df_train_5[["id","date","store_nbr_0","store_nbr_1","store_nbr_
df_test_5 = df_test_5[["id","date","store_nbr_0","store_nbr_1","store_nbr_2"
```

In [53]:
```
df_train_5.head()
```

Out[53]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	2013-01-01	0	0	0	0	0	0
1	1	2013-01-01	0	0	0	0	0	0
2	2	2013-01-01	0	0	0	0	0	0
3	3	2013-01-01	0	0	0	0	0	0
4	4	2013-01-01	0	0	0	0	0	0

5 rows × 26 columns

In [54]:
```
df_test_5.head()
```

Out[54]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_
0	3000888	2017-08-16	0	0	0	0	0	
1	3000889	2017-08-16	0	0	0	0	0	
2	3000890	2017-08-16	0	0	0	0	0	
3	3000891	2017-08-16	0	0	0	0	0	
4	3000892	2017-08-16	0	0	0	0	0	

5 rows × 25 columns

In [55]:
```
print("Number of rows in training set : ",len(df_train_5))
print("Number of rows in testing set  : ",len(df_test_5))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Checking for "state" feature

In [56]:
```
df_train_5["state"].unique()
```

Out[56]:
```
array(['Pichincha', 'Cotopaxi', 'Chimborazo', 'Imbabura',
       'Santo Domingo de los Tsachilas', 'Bolivar', 'Pastaza',
       'Tungurahua', 'Guayas', 'Santa Elena', 'Los Rios', 'Azuay', 'Loja',
       'El Oro', 'Esmeraldas', 'Manabi'], dtype=object)
```

In [57]:
```
df_test_5["state"].unique()
```

Out[57]:
```
array(['Pichincha', 'Cotopaxi', 'Chimborazo', 'Imbabura',
       'Santo Domingo de los Tsachilas', 'Bolivar', 'Pastaza',
       'Tungurahua', 'Guayas', 'Santa Elena', 'Los Rios', 'Azuay', 'Loja',
       'El Oro', 'Esmeraldas', 'Manabi'], dtype=object)
```

Mapping "state" feature with numbers

In [58]:
```
state = dict()
ctr = 0
for i in df_train_5["state"].unique():
    state[i] = ctr
    ctr += 1
state
```

Out[58]:
```
{'Pichincha': 0,
 'Cotopaxi': 1,
 'Chimborazo': 2,
 'Imbabura': 3,
 'Santo Domingo de los Tsachilas': 4,
 'Bolivar': 5,
 'Pastaza': 6,
 'Tungurahua': 7,
 'Guayas': 8,
 'Santa Elena': 9,
 'Los Rios': 10,
 'Azuay': 11,
 'Loja': 12,
 'El Oro': 13,
 'Esmeraldas': 14,
 'Manabi': 15}
```

In [59]:
```
df_train_6 = df_train_5.copy()
df_test_6 = df_test_5.copy()
df_train_6["state"] = df_train_6["state"].map(state)
df_test_6["state"] = df_test_6["state"].map(state)
```

In [60]:
```
state_train = binary_encoder(df_train_6,"state")
state_test = binary_encoder(df_test_6,"state")
```

In [61]:
```
state_train.head()
```

Out[61]:

	state_0	state_1	state_2	state_3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

In [62]:
```
state_test.head()
```

Out[62]:

	state_0	state_1	state_2	state_3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

In [64]:
```
print("Number of rows in training set : ",len(state_train))
print("Number of rows in testing set  : ",len(state_test))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

In [65]:
```
df_train_7 = df_train_6.copy()
df_test_7 = df_test_6.copy()
df_train_7[state_train.columns.values] = state_train
df_test_7[state_test.columns.values] = state_test
df_train_7.drop("state",axis=1,inplace=True)
df_test_7.drop("state",axis=1,inplace=True)
```

```
df_train_7 = df_train_7[["id","date","store_nbr_0","store_nbr_1","store_nbr_
df_test_7 = df_test_7[["id","date","store_nbr_0","store_nbr_1","store_nbr_2"
```

In [66]:
```
df_train_7.head()
```

Out[66]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	2013-01-01	0	0	0	0	0	0
1	1	2013-01-01	0	0	0	0	0	0
2	2	2013-01-01	0	0	0	0	0	0
3	3	2013-01-01	0	0	0	0	0	0
4	4	2013-01-01	0	0	0	0	0	0

5 rows × 29 columns

In [67]:
```
df_test_7.head()
```

Out[67]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_
0	3000888	2017-08-16	0	0	0	0	0	
1	3000889	2017-08-16	0	0	0	0	0	
2	3000890	2017-08-16	0	0	0	0	0	
3	3000891	2017-08-16	0	0	0	0	0	
4	3000892	2017-08-16	0	0	0	0	0	

5 rows × 28 columns

In [68]:
```
print("Number of rows in training set : ",len(df_train_7))
print("Number of rows in testing set  : ",len(df_test_7))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Checking for "cluster" feature

In [74]:
```
clu_train = df_train_7["cluster"].unique()
clu_test = df_test_7["cluster"].unique()

clu_train.sort()
clu_test.sort()
```

```
clu_train = clu_train - 1
clu_test = clu_test - 1
```

In [75]:
```
clu_train
```

Out[75]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16])
```

In [76]:
```
clu_test
```

Out[76]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16])
```

In [100…
```
df_train_8 = df_train_7.copy()
df_test_8 = df_test_7.copy()
df_train_8["cluster"] = df_train_8["cluster"] - 1
df_test_8["cluster"] = df_test_8["cluster"] - 1
```

In [101…
```
cluster_train = binary_encoder(df_train_8,"cluster")
cluster_test = binary_encoder(df_test_8,"cluster")
```

In [102…
```
cluster_train.head()
```

Out[102]:

	cluster_0	cluster_1	cluster_2	cluster_3	cluster_4
0	0	1	1	0	0
1	0	1	1	0	0
2	0	1	1	0	0
3	0	1	1	0	0
4	0	1	1	0	0

In [103…
```
cluster_test.head()
```

Out[103]:

	cluster_0	cluster_1	cluster_2	cluster_3	cluster_4
0	0	1	1	0	0
1	0	1	1	0	0
2	0	1	1	0	0
3	0	1	1	0	0
4	0	1	1	0	0

In [104…
```
print("Number of rows in training set : ",len(cluster_train))
print("Number of rows in testing set  : ",len(cluster_test))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

In [106…
```
df_train_9 = df_train_8.copy()
df_test_9 = df_test_8.copy()
df_train_9[cluster_train.columns.values] = cluster_train
df_test_9[cluster_test.columns.values] = cluster_test
df_train_9.drop("cluster",axis=1,inplace=True)
df_test_9.drop("cluster",axis=1,inplace=True)
df_train_9 = df_train_9[["id","date","store_nbr_0","store_nbr_1","store_nbr_
df_test_9 = df_test_9[["id","date","store_nbr_0","store_nbr_1","store_nbr_2"
```

In [107…
```
df_train_9.head()
```

Out[107]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	2013-01-01	0	0	0	0	0	0
1	1	2013-01-01	0	0	0	0	0	0
2	2	2013-01-01	0	0	0	0	0	0
3	3	2013-01-01	0	0	0	0	0	0
4	4	2013-01-01	0	0	0	0	0	0

5 rows × 33 columns

In [108…
```
df_test_9.head()
```

Out[108]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store
0	3000888	2017-08-16	0	0	0	0	0	
1	3000889	2017-08-16	0	0	0	0	0	
2	3000890	2017-08-16	0	0	0	0	0	
3	3000891	2017-08-16	0	0	0	0	0	
4	3000892	2017-08-16	0	0	0	0	0	

5 rows × 32 columns

In [109…
```
print("Number of rows in training set : ",len(df_train_9))
print("Number of rows in testing set  : ",len(df_test_9))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Checking for "type" column

In [112…
```
type_train = df_train_9["type"].unique()
type_test = df_test_9["type"].unique()
type_train.sort()
type_test.sort()
```

In [113…
```
type_train
```

Out[113]:
```
array(['A', 'B', 'C', 'D', 'E'], dtype=object)
```

In [114…
```
type_test
```

Out[114]:
```
array(['A', 'B', 'C', 'D', 'E'], dtype=object)
```

In [115…]:
```
type = dict()
ctr = 0
for i in type_train:
    type[i] = ctr
    ctr += 1
type
```

Out[115]:
```
{'A': 0, 'B': 1, 'C': 2, 'D': 3, 'E': 4}
```

In [116…]:
```
df_train_10 = df_train_9.copy()
df_test_10 = df_test_9.copy()
```

In [117…]:
```
df_train_10["type"] = df_train_10["type"].map(type)
df_test_10["type"] = df_test_10["type"].map(type)
```

In [118…]:
```
type_tr = binary_encoder(df_train_10,"type")
type_te = binary_encoder(df_test_10,"type")
```

In [119…]:
```
type_tr.head()
```

Out[119]:

	type_0	type_1	type_2
0	0	1	1
1	0	1	1
2	0	1	1
3	0	1	1
4	0	1	1

In [120…]:
```
type_te.head()
```

Out[120]:

	type_0	type_1	type_2
0	0	1	1
1	0	1	1
2	0	1	1
3	0	1	1
4	0	1	1

In [121…]:
```
df_train_11 = df_train_10.copy()
df_test_11 = df_test_10.copy()
df_train_11[type_tr.columns.values] = type_tr
df_test_11[type_te.columns.values] = type_te
df_train_11.drop("type",axis=1,inplace=True)
df_test_11.drop("type",axis=1,inplace=True)
df_train_11 = df_train_11[["id","date","store_nbr_0","store_nbr_1","store_nb
df_test_11 = df_test_11[["id","date","store_nbr_0","store_nbr_1","store_nbr_
```

In [122…]:
```
df_train_11.head()
```

Out[122]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store_nbr_5
0	0	2013-01-01	0	0	0	0	0	0
1	1	2013-01-01	0	0	0	0	0	0
2	2	2013-01-01	0	0	0	0	0	0
3	3	2013-01-01	0	0	0	0	0	0
4	4	2013-01-01	0	0	0	0	0	0

5 rows × 35 columns

In [123…
```
df_test_11.head()
```

Out[123]:

	id	date	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3	store_nbr_4	store
0	3000888	2017-08-16	0	0	0	0	0	
1	3000889	2017-08-16	0	0	0	0	0	
2	3000890	2017-08-16	0	0	0	0	0	
3	3000891	2017-08-16	0	0	0	0	0	
4	3000892	2017-08-16	0	0	0	0	0	

5 rows × 34 columns

In [124…
```
print("Number of rows in training set : ",len(df_train_11))
print("Number of rows in testing set  : ",len(df_test_11))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

Checking for date feature

In [146…
```
df_train_11["date"].dtype
```

Out[146]:
```
dtype('<M8[ns]')
```

In [147…
```
df_test_11["date"].dtype
```

Out[147]:
```
dtype('<M8[ns]')
```

In [176…
```
y_year = lambda x: x.year
map_year = map(y_year,df_train_11["date"])
lis_year = list(map_year)
```

```
y_month = lambda x: x.month
map_month = map(y_month,df_train_11["date"])
lis_month = list(map_month)

y_day = lambda x: x.day
map_day = map(y_day,df_train_11["date"])
lis_day = list(map_day)

df_train_12 = df_train_11.copy()
df_train_12.drop("date",axis=1,inplace=True)

df_train_12["date_year"] = lis_year
df_train_12["date_month"] = lis_month
df_train_12["date_day"] = lis_day

df_train_12 = df_train_12[["id","date_year","date_month","date_day","store_n
```

In [177…

```
y_year = lambda x: x.year
map_year = map(y_year,df_test_11["date"])
lis_year = list(map_year)

y_month = lambda x: x.month
map_month = map(y_month,df_test_11["date"])
lis_month = list(map_month)

y_day = lambda x: x.day
map_day = map(y_day,df_test_11["date"])
lis_day = list(map_day)

df_test_12 = df_test_11.copy()
df_test_12.drop("date",axis=1,inplace=True)

df_test_12["date_year"] = lis_year
df_test_12["date_month"] = lis_month
df_test_12["date_day"] = lis_day

df_test_12 = df_test_12[["id","date_year","date_month","date_day","store_nbr
```

In [180…

```
df_train_12.head()
```

Out[180]:

	id	date_year	date_month	date_day	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_
0	0	2013	1	1	0	0	0	
1	1	2013	1	1	0	0	0	
2	2	2013	1	1	0	0	0	
3	3	2013	1	1	0	0	0	
4	4	2013	1	1	0	0	0	

5 rows × 37 columns

In [181…

```
df_test_12.head()
```

Out[181]:

	id	date_year	date_month	date_day	store_nbr_0	store_nbr_1	store_nbr_2	sto
0	3000888	2017	8	16	0	0	0	
1	3000889	2017	8	16	0	0	0	
2	3000890	2017	8	16	0	0	0	
3	3000891	2017	8	16	0	0	0	
4	3000892	2017	8	16	0	0	0	

5 rows × 36 columns

In [182…
```
print("Number of rows in training set : ",len(df_train_12))
print("Number of rows in testing set  : ",len(df_test_12))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

In [183…
```
retr_df_train = df_train_12.drop("id",axis=1,inplace=False)
retr_df_test = df_test_12.drop("id",axis=1,inplace=False)
```

In [187…
```
retr_df_train.head()
```

Out[187]:

	date_year	date_month	date_day	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3
0	2013	1	1	0	0	0	0
1	2013	1	1	0	0	0	0
2	2013	1	1	0	0	0	0
3	2013	1	1	0	0	0	0
4	2013	1	1	0	0	0	0

5 rows × 36 columns

In [188…
```
retr_df_test.head()
```

Out[188]:

	date_year	date_month	date_day	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3
0	2017	8	16	0	0	0	0
1	2017	8	16	0	0	0	0
2	2017	8	16	0	0	0	0
3	2017	8	16	0	0	0	0
4	2017	8	16	0	0	0	0

5 rows × 35 columns

In [189…
```
print("Number of rows in training set : ",len(retr_df_train))
print("Number of rows in testing set  : ",len(retr_df_test))
```

```
Number of rows in training set :  1048575
Number of rows in testing set  :  28512
```

In [191…
```
X = retr_df_train.drop("sales",axis=1,inplace=False)
y = retr_df_train["sales"]
```

In [192…
```
X.head()
```

Out[192]:

	date_year	date_month	date_day	store_nbr_0	store_nbr_1	store_nbr_2	store_nbr_3
0	2013	1	1	0	0	0	0
1	2013	1	1	0	0	0	0
2	2013	1	1	0	0	0	0
3	2013	1	1	0	0	0	0
4	2013	1	1	0	0	0	0

5 rows × 35 columns

In [193…
```
y.head()
```

Out[193]:
```
0    0.0
1    0.0
2    0.0
3    0.0
4    0.0
Name: sales, dtype: float64
```

In [195…
```
print("Number of rows in X set : ",len(X))
print("Number of rows in y set : ",len(y))
```

```
Number of rows in X set :  1048575
Number of rows in y set :  1048575
```

In [230…
```
f_reg = f_regression(X,y)
p_values1 = f_reg[1]
p_values2 = p_values1.round(2)
p_values = pd.DataFrame(columns=["features","p-values"])
p_values["features"] = X.columns.values
p_values["p-values"] = p_values2
```

In [231…
```
p_values.head()
```

Out[231]:

	features	p-values
0	date_year	0.00
1	date_month	0.05
2	date_day	0.00
3	store_nbr_0	0.00
4	store_nbr_1	0.00

In [232…
```
print("Length of p-values dataframe : ",len(p_values))
```

```
Length of p-values dataframe :  35
```

In [233…
```
p_values.head(5)
```

Out[233]:

	features	p-values
0	date_year	0.00
1	date_month	0.05
2	date_day	0.00
3	store_nbr_0	0.00
4	store_nbr_1	0.00

In [234…
```
p_values[5:11]
```

Out[234]:

	features	p-values
5	store_nbr_2	0.00
6	store_nbr_3	0.46
7	store_nbr_4	0.00
8	store_nbr_5	0.00
9	family_0	0.00
10	family_1	0.00

In [235…
```
p_values[10:15]
```

Out[235]:

	features	p-values
10	family_1	0.0
11	family_2	0.0
12	family_3	0.0
13	family_4	0.0
14	family_5	0.0

In [236…
```
p_values[15:20]
```

Out[236]:

	features	p-values
15	onpromotion	0.0
16	city_0	0.0
17	city_1	0.0
18	city_2	0.0
19	city_3	0.0

In [237…
```
p_values[20:25]
```

Out[237]:

	features	p-values
20	city_4	0.0
21	state_0	0.0
22	state_1	0.0
23	state_2	0.0
24	state_3	0.0

In [238…
```
p_values[25:30]
```

Out[238]:

	features	p-values
25	type_0	0.0
26	type_1	0.0
27	type_2	0.0
28	cluster_0	0.0
29	cluster_1	0.0

In [239…
```
p_values[30:35]
```

Out[239]:

	features	p-values
30	cluster_2	0.0
31	cluster_3	0.0
32	cluster_4	0.0
33	dcoilwtico	0.0
34	holiday?	0.0

In [240…
```
X.to_csv("train_X_preprocessed.csv",index=False)
y.to_csv("train_y_preprocessed.csv",index=False)
retr_df_test.to_csv("test_preprocessed.csv",index=False)
```

In [ ]:
```
```