Description

- Demand Forecast is one of the key tasks in Supply Chain and Retail Domain in general. It is key in effective operation and optimization of retail supply chain. Effectively solving this problem requires knowledge about a wide range of tricks in Data Sciences and good understanding of ensemble techniques.
- Training Data Description: Historic sales at Store-Day level for about two years for a retail giant, for more than 1000 stores. Also, other sale influencers like, whether on a particular day the store was fully open or closed for renovation, holiday and special event details, are also provided.

import libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

import seaborn as sns
sns.set_style("whitegrid")

from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error,mean_squared_error
from sklearn.preprocessing import LabelEncoder
import tensorflow as tf
import math

import warnings
warnings.filterwarnings('ignore')

from statsmodels.tsa.stattools import adfuller,acf,pacf
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.arima_model import ARIMA
from keras.models import Sequential
from keras.layers import Dense, LSTM
from keras import layers
from keras.optimizers import Adam,RMSprop,SGD,Adagrad
from keras.models import load_model
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import Ridge
from sklearn.ensemble import AdaBoostRegressor
```

Project Task: Week 1

Exploratory Data Analysis (EDA) and Linear Regression:

In [2]:
```
train_data = pd.read_csv(r'D:\Simplilearn\project\Artificial-Intelligence-Capstone-Project-Datasets-master\Project 3-Retail-Datasets_data\tr
train_data.Date = pd.to_datetime(train_data.Date)
train_data.head()
```

Out[2]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	2	2015-06-30	5735	568	1	1	0	0
1	2	2	2015-06-30	9863	877	1	1	0	0
2	3	2	2015-06-30	13261	1072	1	1	0	1
3	4	2	2015-06-30	13106	1488	1	1	0	0
4	5	2	2015-06-30	6635	645	1	1	0	0

In [3]:
```
train_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 982644 entries, 0 to 982643
Data columns (total 9 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   Store          982644 non-null  int64
 1   DayOfWeek      982644 non-null  int64
 2   Date           982644 non-null  datetime64[ns]
 3   Sales          982644 non-null  int64
 4   Customers      982644 non-null  int64
 5   Open           982644 non-null  int64
 6   Promo          982644 non-null  int64
 7   StateHoliday   982644 non-null  object
 8   SchoolHoliday  982644 non-null  int64
dtypes: datetime64[ns](1), int64(7), object(1)
memory usage: 67.5+ MB
```

- StateHoliday column is object type

1. Transform the variables by using data manipulation techniques like, One-Hot Encoding

In [4]:
```python
train_data.StateHoliday.unique() ## checking unique values
```
Out[4]:
```
array(['0', 'a', 'b', 'c', 0], dtype=object)
```

In [5]:
```python
train_data.loc[train_data.StateHoliday==0,'StateHoliday'] = '0'
```

- use One-Hot Encoding to convert this column

In [6]:
```python
labelencoder= LabelEncoder()
train_data.StateHoliday = labelencoder.fit_transform(train_data['StateHoliday'])
```

In [7]:
```python
train_data.StateHoliday.unique()
```
Out[7]:
```
array([0, 1, 2, 3])
```

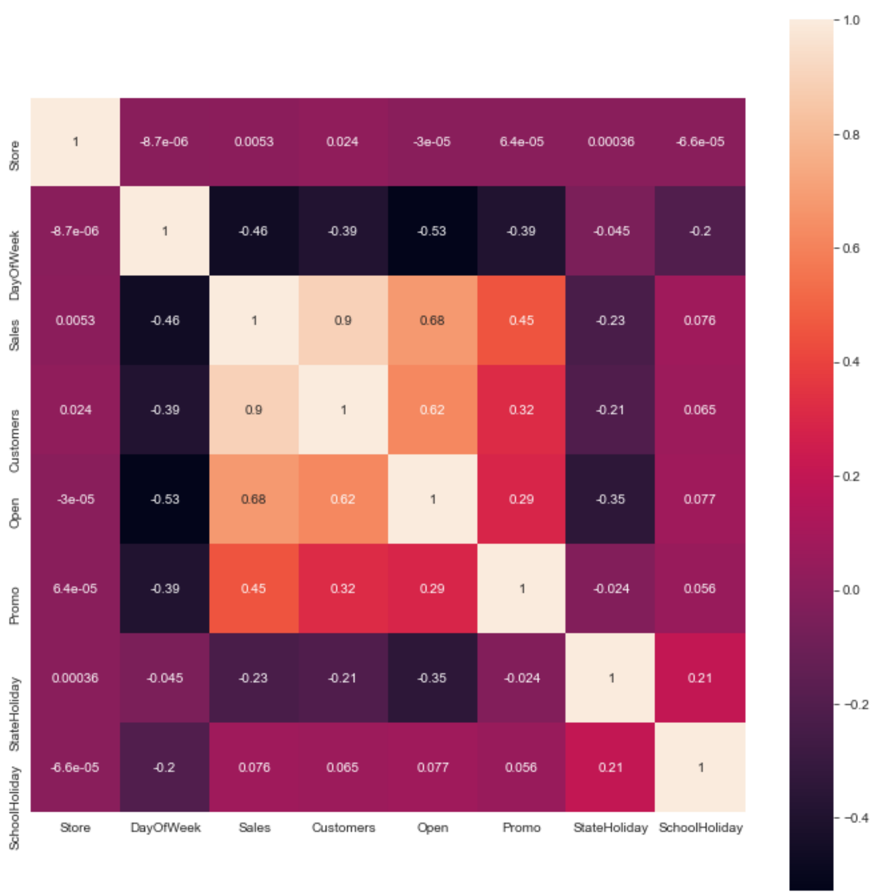
2. Perform an EDA (Exploratory Data Analysis) to see the impact of variables over Sales.

In [8]:
```python
train_data.isna().sum() # checking null values
```
Out[8]:
```
Store            0
DayOfWeek        0
Date             0
Sales            0
Customers        0
Open             0
Promo            0
StateHoliday     0
SchoolHoliday    0
dtype: int64
```

- There is no null value

In [9]:
```python
train_data =train_data[~train_data.isin([np.nan, np.inf, -np.inf]).any(1)]
```

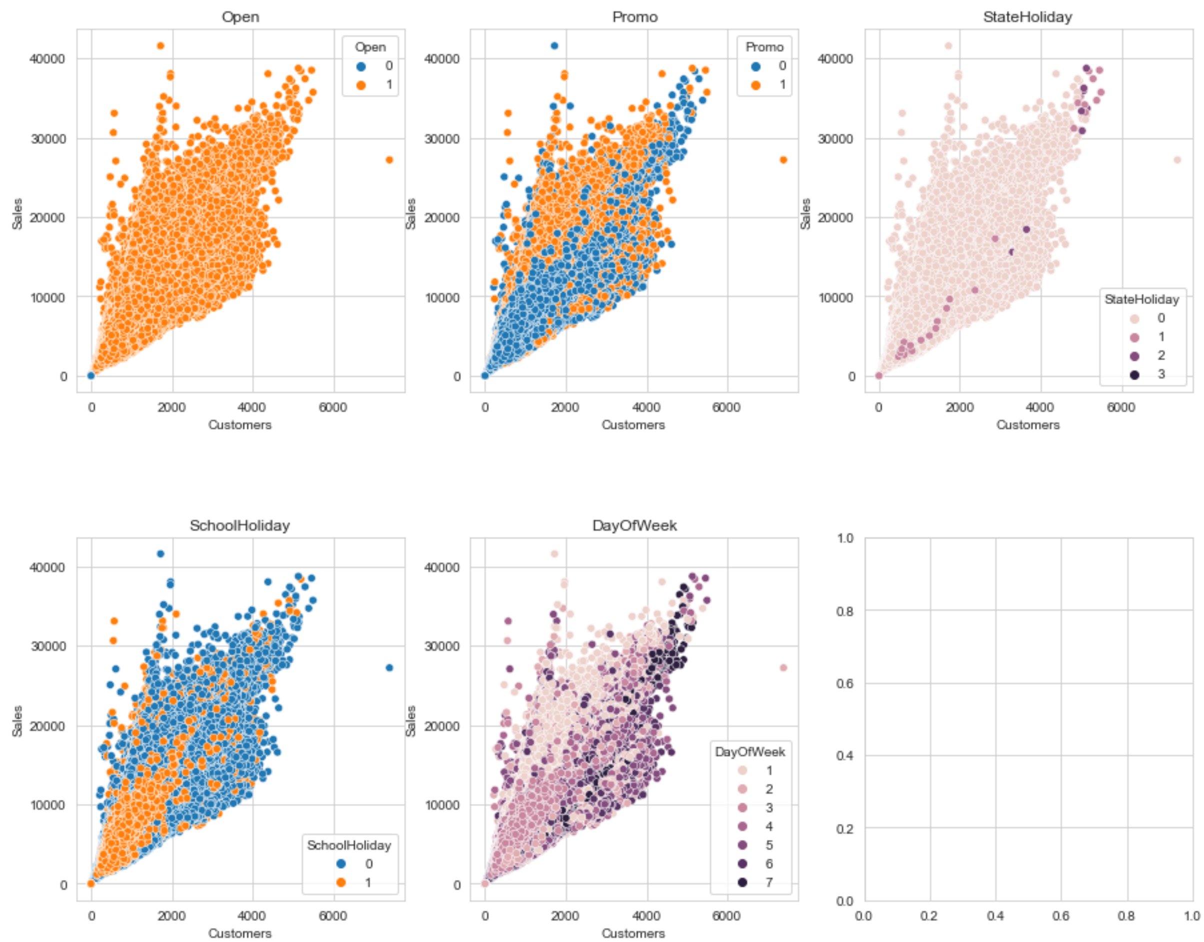
In [10]:
```python
# check the correlation between variables
plt.figure(figsize=(12,12))
sns.heatmap(train_data.corr(),annot=True, square=True)
```
Out[10]:
```
<AxesSubplot:>
```



- The above heatmap shows that there is a very good correlation between 'sales' and 'customer'.

- 'Sales' is also related to 'Open' and 'promo'.
- 'Sales' is negatively correlated with 'StateHoliday' and 'DayOfWeek'

In [11]:
```
fig, axs = plt.subplots(2,3, figsize=(15,12))
fig.subplots_adjust(hspace=0.4)
axs=axs.ravel()
A_list = ['Open', 'Promo','StateHoliday', 'SchoolHoliday','DayOfWeek']
i=0
for col in A_list:
    sns.scatterplot(train_data.Customers,train_data.Sales,hue=train_data[col],ax=axs[i])
    axs[i].set_title(col)
    i+=1

fig.show()
```



From the above EDA analysis we can conclude as follow:

```
* Sales is zero when shop is closed.

* Sales is high when promo codes and discount is available.

* Sales is either very low or very high on State Holidays.

* Sales is high when schools are open.
```

3. Apply Linear Regression to predict the forecast and evaluate different accuracy metrices like RMSE (Root Mean Squared Error) and MAE(Mean Absolute Error) and determine which metric makes more sense. Can there be a better accuracy metric?

a) Train a single model for all stores, using storeId as a feature.

In [12]:
```
## build linear regression model for all stores
```

In [13]:
```
##Total number of stores
n_shops = train_data.Store.nunique()
n_shops
```

Out[13]: 1115

In [14]:
```
original_data = train_data.copy()
train_data.drop('Date', axis=1, inplace=True)
```

In [15]:
```
y=np.array(train_data['Sales'])
x=np.array(train_data.drop('Sales',axis=1))
```

In [16]:
```
lr = LinearRegression(normalize=True)
```

In [17]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,random_state = 100, test_size=0.3)
```

In [18]:
```
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(687850, 7)
(294794, 7)
(687850,)
(294794,)
```

In [19]:
```
lr.fit(x_train,y_train)
```

Out[19]:
```
LinearRegression(normalize=True)
```

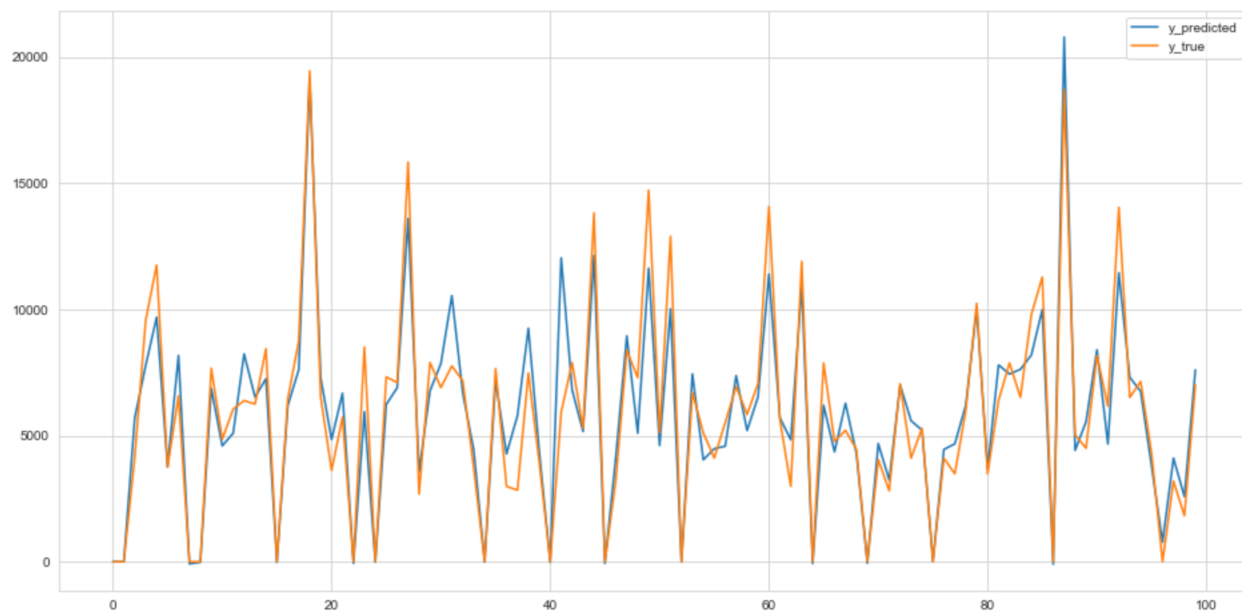
In [20]:
```
y_pred = lr.predict(x_test)
```

In [21]:
```
def error_cal(y_true,y_pred):
    RMSE = math.sqrt(mean_squared_error(y_true,y_pred))
    MAE = mean_absolute_error(y_true,y_pred)
    return RMSE,MAE
```

In [22]:
```
all_store_lr = error_cal(y_test,y_pred)
all_store_lr
```

Out[22]:
```
(1471.475458119501, 978.5123617111672)
```

In [23]:
```
plt.figure(figsize=(16,8))
plt.plot(y_pred[:100],label = 'y_predicted')
plt.plot(y_test[:100], label = 'y_true')
plt.legend()
plt.show()
```



b) Train separate model for each store.

In [24]:
```
## build linear regression model for individual stores
```

In [25]:
```
def model_single_store(x,y):
    lr = LinearRegression(normalize=True)
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=42)
    lr.fit(x_train,y_train)
    y_pred = lr.predict(x_test)
    return y_test,y_pred
```

In [26]:
```
stores = [1,2,3,4,5,6,7,8,9,10,11]
```

In [27]:
```
RMSE_array_lr = []
MAE_array_lr=[]
for store in range(1,12):
    data = train_data[train_data.Store==store]
    data.drop('Store',axis=1,inplace=True)
```

```
    y=np.array(data['Sales'])
    x=np.array(data.drop('Sales',axis=1))
    y_true,y_pred = model_single_store(x,y)
    RMSE_1,MAE_1 = error_cal(y_true,y_pred)
    RMSE_array_lr.append(RMSE_1)
    MAE_array_lr.append(MAE_1)
```

In [28]:

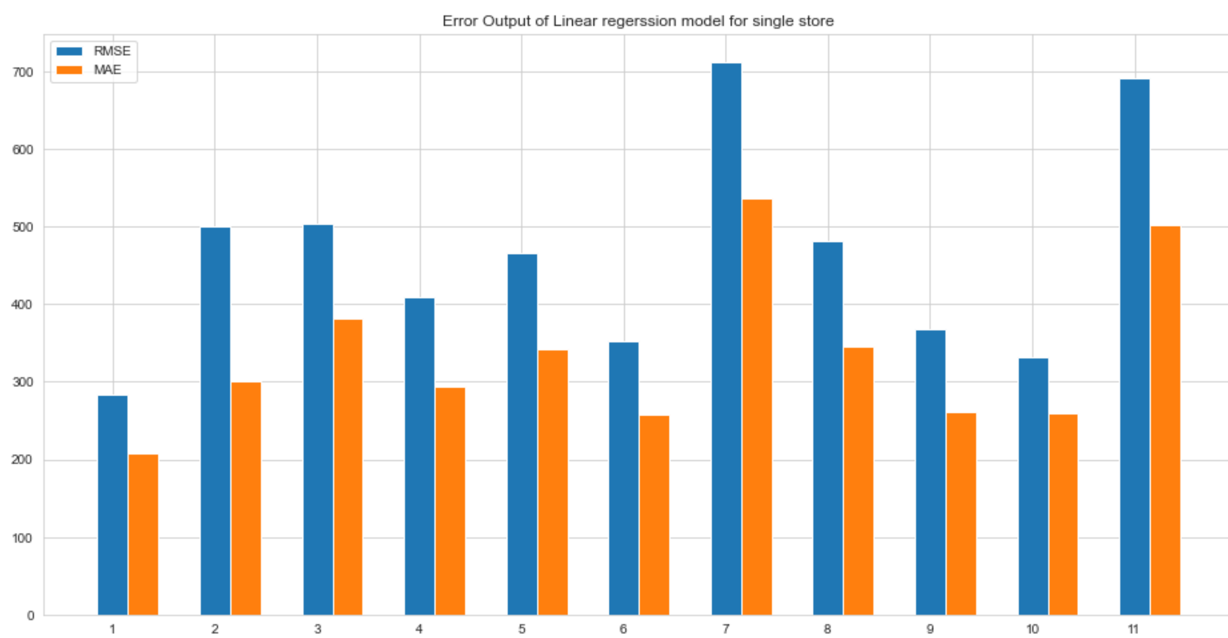
```
error_output_lr = pd.DataFrame()
error_output_lr['Stores'] = stores
error_output_lr['RMSE'] = RMSE_array_lr
error_output_lr['MAE'] = MAE_array_lr
error_output_lr
```

Out[28]:

	Stores	RMSE	MAE
0	1	284.267843	208.057711
1	2	499.547398	300.639640
2	3	503.516363	380.807745
3	4	408.954683	293.918216
4	5	466.091596	342.290667
5	6	352.195690	257.880767
6	7	712.565579	536.974131
7	8	481.808989	345.948447
8	9	368.470541	261.323809
9	10	331.881675	259.922444
10	11	691.636510	501.935023

In [29]:

```
plt.figure(figsize=(16,8))
N = 12
x = np.arange(1,N)
plt.bar(x,height=error_output_lr.RMSE,label = 'RMSE',width = 0.3)
plt.bar(x+0.3,height=error_output_lr.MAE,label = 'MAE',width = 0.3)
plt.xticks(stores)
plt.legend()
plt.title('Error Output of Linear regerssion model for single store')
plt.show()
```



c) Which performs better and Why?

- Above model shows that accuracy increases when we train our model for individual store, because Sales might also depend on geographical or locality of the store, So when we predict for individual store then this factor could be treated as constant and can be neglected.

d) Try Ensemble of b) and c). What are the findings?

In [30]:

```
## ensemble learning
```

In [31]:

```
def adaboost_single_store(x,y):
    lr = AdaBoostRegressor(n_estimators=100, learning_rate=0.1)
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=42)
    lr.fit(x_train,y_train)
```

```python
    y_pred = lr.predict(x_test)
    return y_test,y_pred
```

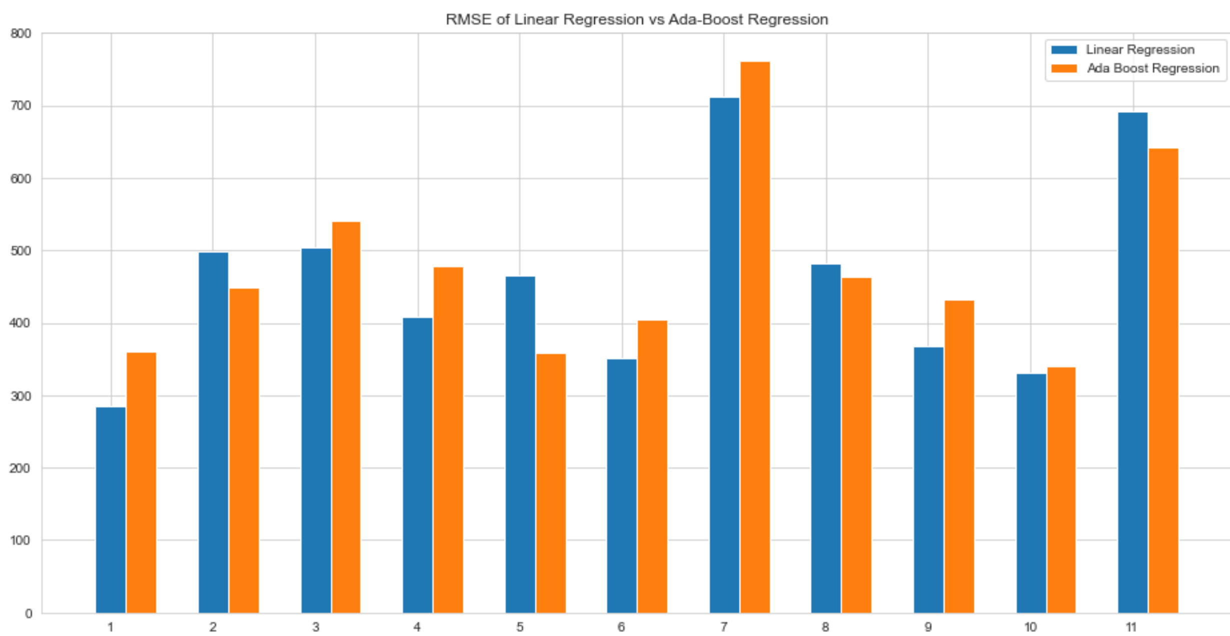
In [32]:
```python
RMSE_array_ada = []
MAE_array_ada =[]
for store in range(1,12):
    data = train_data[train_data.Store==store]
    data.drop('Store',axis=1,inplace=True)
    y=np.array(data['Sales'])
    x=np.array(data.drop('Sales',axis=1))
    y_true,y_pred = adaboost_single_store(x,y)
    RMSE_1,MAE_1 = error_cal(y_true,y_pred)
    RMSE_array_ada.append(RMSE_1)
    MAE_array_ada.append(MAE_1)
```

In [33]:
```python
error_output_ada = pd.DataFrame()
error_output_ada['Stores'] = stores
error_output_ada['RMSE'] = RMSE_array_ada
error_output_ada['MAE'] = MAE_array_ada
error_output_ada
```

Out[33]:

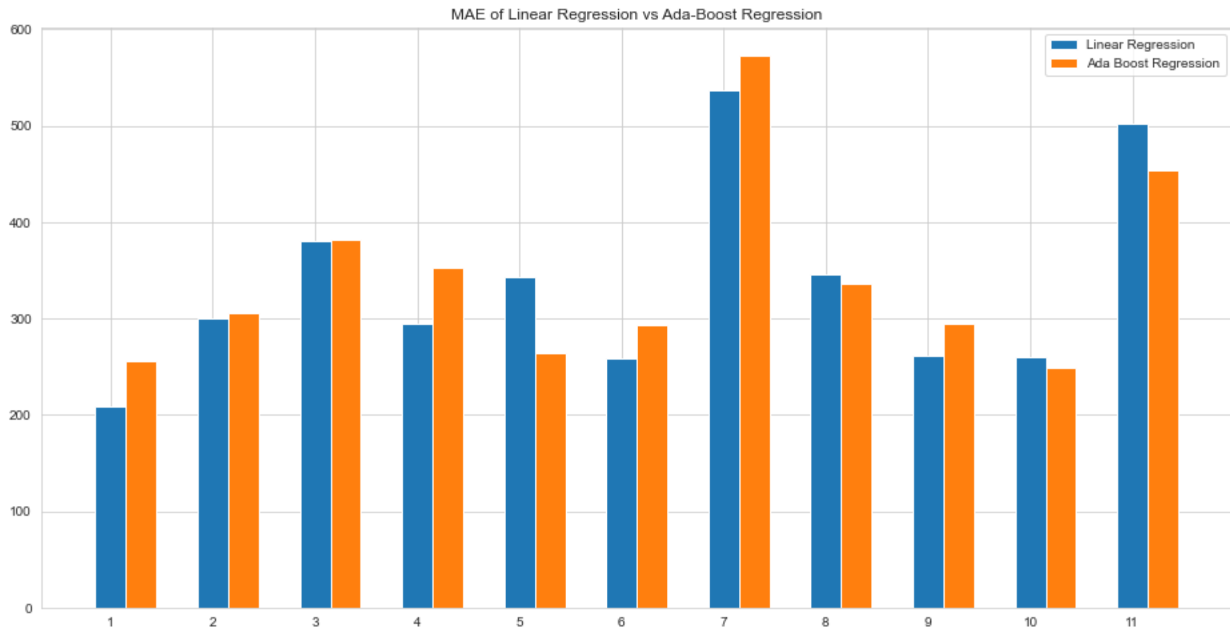
	Stores	RMSE	MAE
0	1	359.790941	255.360556
1	2	448.281241	305.834994
2	3	540.741881	381.984390
3	4	478.518854	353.173600
4	5	358.276584	264.578576
5	6	405.072046	293.649487
6	7	762.623474	573.200371
7	8	462.982574	336.571569
8	9	431.440366	294.909974
9	10	340.029931	248.239667
10	11	641.582876	453.875135

In [34]:
```python
plt.figure(figsize=(16,8))
N=12
x=np.arange(1,N)
plt.bar(x,height=error_output_lr.RMSE,label = 'Linear Regression',width=0.3)
plt.bar(x+0.3,height=error_output_ada.RMSE,label = 'Ada Boost Regression',width=0.3)
plt.xticks(stores)
plt.legend()

plt.title('RMSE of Linear Regression vs Ada-Boost Regression')
plt.show()
```



In [35]:
```python
plt.figure(figsize=(16,8))
N=12
x=np.arange(1,N)
plt.bar(x,height=error_output_lr.MAE,label = 'Linear Regression',width=0.3)
plt.bar(x+0.3,height=error_output_ada.MAE,label = 'Ada Boost Regression',width=0.3)
```

```
plt.xticks(stores)
plt.legend()

plt.title('MAE of Linear Regression vs Ada-Boost Regression')
plt.show()
```



- By comparing RMSE and MAE of ada-boost regression and Linear regression, we can say that there is not much difference between these models.

e) Use Regularized Regression. It should perform better in an unseen test set. Any insights??

In [36]:
```
## regularised regression
```

In [37]:
```
def ridge_single_store(x,y):
    ridge = Ridge(alpha=0.00001,normalize=True)
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=18)
    ridge.fit(x_train,y_train)
    y_pred = ridge.predict(x_test)
    return y_test,y_pred
```

In [38]:
```
RMSE_array_rdg = []
MAE_array_rdg =[]
for store in range(1,12):
    data = train_data[train_data.Store==store]
    data.drop('Store',axis=1,inplace=True)
    y=np.array(data['Sales'])
    x=np.array(data.drop('Sales',axis=1))
    y_true,y_pred = ridge_single_store(x,y)
    RMSE_1,MAE_1 = error_cal(y_true,y_pred)
    RMSE_array_rdg.append(RMSE_1)
    MAE_array_rdg.append(MAE_1)
```

In [39]:
```
error_output_rdg = pd.DataFrame()
error_output_rdg['Stores'] = stores
error_output_rdg['RMSE'] = RMSE_array_rdg
error_output_rdg['MAE'] = MAE_array_rdg
error_output_rdg
```

Out[39]:

	Stores	RMSE	MAE
0	1	253.651635	185.988246
1	2	658.876634	350.142560
2	3	505.016668	359.602319
3	4	440.340415	316.605281
4	5	482.497517	329.292617
5	6	318.540876	227.956072
6	7	717.864403	509.881475
7	8	472.085464	339.311060
8	9	403.765187	269.977126
9	10	335.135667	255.293371
10	11	688.828345	480.443768

f) Open-ended modeling to get possible predictions.

In [40]:
```
## Random forest regression model
```

In [41]:
```
def random_forest(x,y):
    rdm = RandomForestRegressor(n_estimators=60)
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=36)
    rdm.fit(x_train,y_train)
    y_pred = rdm.predict(x_test)
    return y_test,y_pred
```

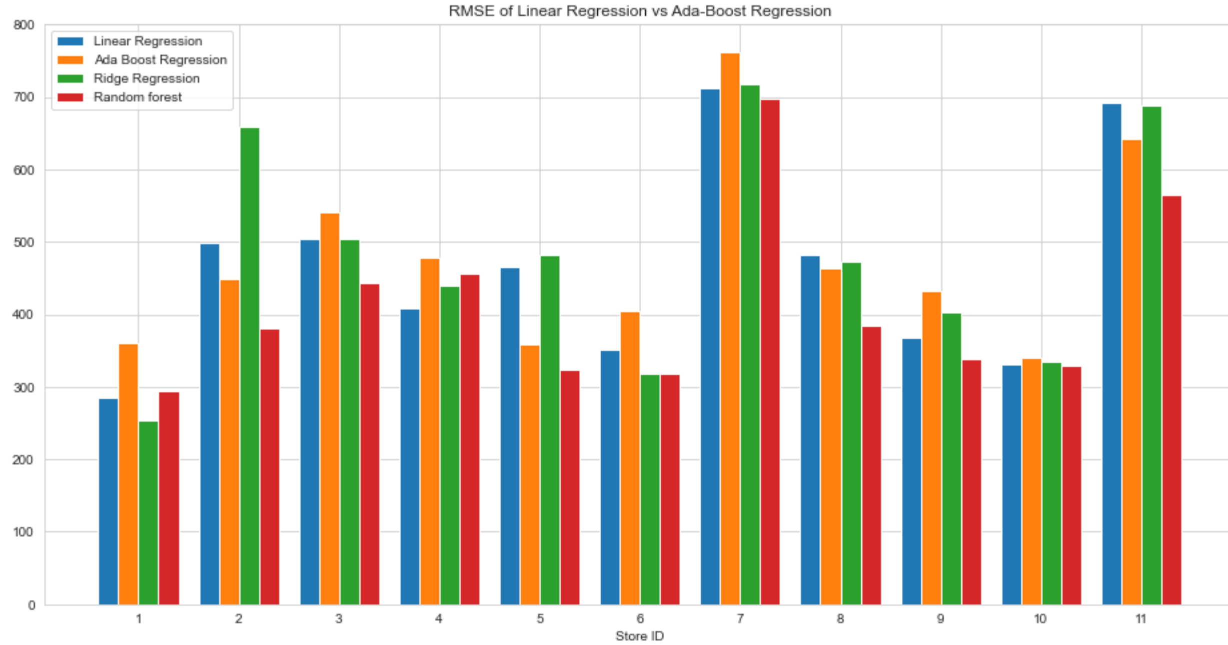
In [42]:
```
RMSE_array_rdm = []
MAE_array_rdm =[]
for store in range(1,12):
    data = train_data[train_data.Store==store]
    data.drop('Store',axis=1,inplace=True)
    y=np.array(data['Sales'])
    x=np.array(data.drop('Sales',axis=1))
    y_true,y_pred = random_forest(x,y)
    RMSE_1,MAE_1 = error_cal(y_true,y_pred)
    RMSE_array_rdm.append(RMSE_1)
    MAE_array_rdm.append(MAE_1)
```

In [43]:
```
error_output_rdm = pd.DataFrame()
error_output_rdm['Stores'] = stores
error_output_rdm['RMSE'] = RMSE_array_rdm
error_output_rdm['MAE'] = MAE_array_rdm
error_output_rdm
```

Out[43]:

	Stores	RMSE	MAE
0	1	293.766041	211.472855
1	2	381.014854	232.517338
2	3	442.631468	314.647135
3	4	456.553940	321.344239
4	5	323.012231	227.066429
5	6	318.773202	224.175162
6	7	697.971325	469.461827
7	8	383.821747	261.705540
8	9	338.337912	238.934426
9	10	329.585731	237.541445
10	11	564.635191	384.195118

In [44]:
```
plt.figure(figsize=(16,8))
N=12
x=np.arange(1,N)
plt.bar(x,height=error_output_lr.RMSE,label = 'Linear Regression',width=0.2)
plt.bar(x+0.2,height=error_output_ada.RMSE,label = 'Ada Boost Regression',width=0.2)
plt.bar(x+0.4,height=error_output_rdg.RMSE,label = 'Ridge Regression',width=0.2)
plt.bar(x+0.6,height=error_output_rdm.RMSE,label = 'Random forest',width=0.2)
plt.xticks((2*x+0.6)/2,stores)
plt.xlabel('Store ID')
plt.legend()

plt.title('RMSE of Linear Regression vs Ada-Boost Regression')
plt.show()
```



- From the above graph we cannot justify that which model is best. so calculate average error.

In [45]:
```
# print('Average RMSE Linear regression Error: {}'.format(error_output_lr.RMSE.mean()))
print('Average RMSE Ada-boost regression Error: {}'.format(error_output_ada.RMSE.mean()))
print('Average RMSE Ridge regression Error: {}'.format(error_output_rdg.RMSE.mean()))
print('Average RMSE Random Forest regression Error: {}'.format(error_output_rdm.RMSE.mean()))
```

```
Average RMSE Ada-boost regression Error: 475.39461525677564
Average RMSE Ridge regression Error: 479.6911645282402
Average RMSE Random Forest regression Error: 411.82760368071763
```

- From Here we can conclude that upto this point random forest performs best.

Project Task: Week 2

Other Regression Techniques:

1. When store is closed, sales = 0. Can this insight be used for Data Cleaning? Perform this and retrain the model. Any benefits of this step?

- When store is closed then there will be no sale. Hence remove that rows.

In [46]:
```
open_store_data = train_data[train_data.Open == 1]
open_store_data.drop('Open',axis=1,inplace=True)
open_store_data.head()
```

Out[46]:

	Store	DayOfWeek	Sales	Customers	Promo	StateHoliday	SchoolHoliday
0	1	2	5735	568	1	0	0
1	2	2	9863	877	1	0	0
2	3	2	13261	1072	1	0	1
3	4	2	13106	1488	1	0	0
4	5	2	6635	645	1	0	0

In [47]:
```
RMSE_array_lrc = []
MAE_array_lrc=[]
for store in range(1,12):
    data = open_store_data[open_store_data.Store==store]
    data.drop('Store',axis=1,inplace=True)
    y=np.array(data['Sales'])
    x=np.array(data.drop('Sales',axis=1))
    y_true,y_pred = model_single_store(x,y)
    RMSE_1,MAE_1 = error_cal(y_true,y_pred)
    RMSE_array_lrc.append(RMSE_1)
    MAE_array_lrc.append(MAE_1)
```

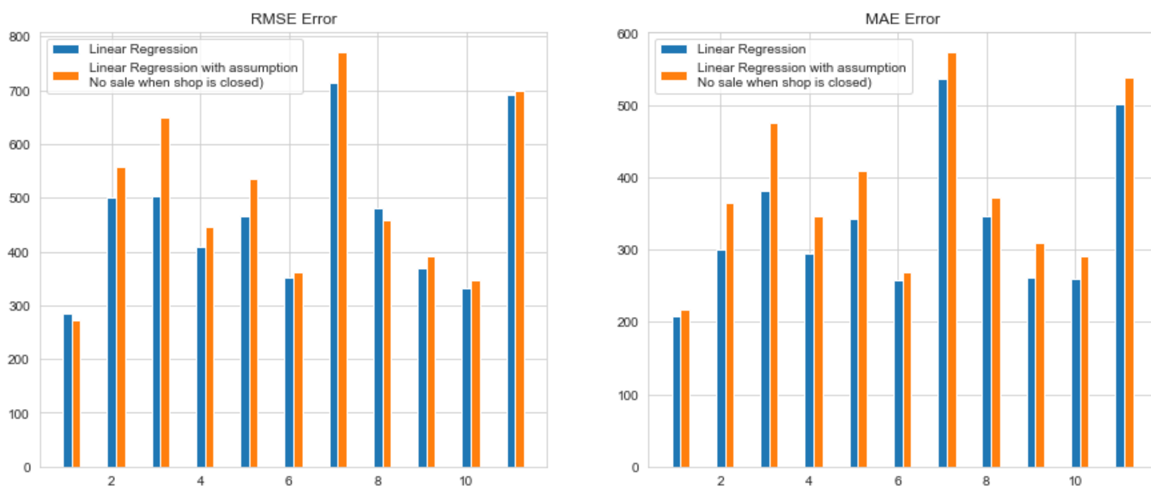
In [48]:
```
error_output_lrc = pd.DataFrame()
error_output_lrc['Stores'] = stores
error_output_lrc['RMSE'] = RMSE_array_lrc
error_output_lrc['MAE'] = MAE_array_lrc
error_output_lrc
```

Out[48]:

	Stores	RMSE	MAE
0	1	271.285014	216.794418
1	2	557.155161	364.294090
2	3	649.968842	476.379140
3	4	444.729628	346.986200
4	5	534.652008	409.715061
5	6	361.446231	269.610691
6	7	770.399228	572.891939
7	8	458.349199	371.704818
8	9	391.073084	309.433183
9	10	347.720527	290.885499
10	11	698.226695	537.427042

In [49]:

```
fig, axs = plt.subplots(1,2, figsize=(15,6))
fig.subplots_adjust(hspace=0.4)
axs=axs.ravel()
N=12
x=np.arange(1,N)
i=0
for col in ['RMSE','MAE']:
    axs[i].bar(x,height=error_output_lr[col],label = 'Linear Regression',width=0.2)
    axs[i].bar(x+0.2,height=error_output_lrc[col],label = 'Linear Regression with assumption\nNo sale when shop is closed)',width=0.2)
    axs[i].legend()
    axs[i].set_title(col+' Error')
    i+=1
```



- The above graph shoes that both types of error get increased when we removed the rows when store are closed.
- I think the main reason for this increased error rate is that, our previous model was predicting accurately when store was closed, so while taking mean of that portion the error output got reduced, but in updated model as we removed that rows, so while taking mean it get increased error output.
- So we do not get any benefit of removing those rows.

2. Use Non-Linear Regressors like Random Forest or other Tree-based Regressors.

a) Train a single model for all stores, where storeId can be a feature.

In [50]:

```
## random forest model for updated cleaned data
```

In [51]:

```
open_store_data.head()
```

Out[51]:

	Store	DayOfWeek	Sales	Customers	Promo	StateHoliday	SchoolHoliday
0	1	2	5735	568	1	0	0
1	2	2	9863	877	1	0	0
2	3	2	13261	1072	1	0	1
3	4	2	13106	1488	1	0	0
4	5	2	6635	645	1	0	0

In [52]:

```
y=np.array(data['Sales'])
x=np.array(data.drop('Sales',axis=1))
y_true,y_pred = random_forest(x,y)
```

In [53]:
```
RMSE_rdm,MAE_rdm = error_cal(y_true,y_pred)
```

In [54]:
```
print('Root mean squared error: ',RMSE_rdm)
print('Mean absolute error: ',MAE_rdm)
```

```
Root mean squared error:  734.5806667844583
Mean absolute error:  498.21204030910604
```

In [55]:
```
plt.figure(figsize=(16,8))
plt.plot(y_pred[:100],label = 'Sales forecast')
plt.plot(y_true[:100],label = 'Actual Sales')
plt.legend()
plt.title('Random Forest Regression taking all stores')
plt.show()
```

b) Train separate models for each store.

Note:

Dimensional Reduction techniques like, PCA and Tree's Hyperparameter Tuning will be required. Cross-validate to find the best parameters. Infer the performance of both the models.

In [56]:
```
open_store_data.reset_index(drop=True,inplace=True)
```

In [57]:
```
x = open_store_data.drop(['Sales','Store'],axis=1)
x = StandardScaler().fit_transform(x)
```

In [58]:
```
pca = PCA(n_components=3)
principalComponents = pca.fit_transform(x)
```

In [59]:
```
principalDf = pd.DataFrame(data = principalComponents
             , columns = ['PC_1', 'pc_2','PC_3'])
```

In [60]:
```
finaldf = pd.concat([open_store_data[['Store','Sales']],principalDf],axis=1)
```

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

Out[61]:

	Store	Sales	PC_1	pc_2	PC_3
0	1	5735	0.891968	-0.602281	0.120771
1	2	9863	1.221942	-0.516040	0.489858
2	3	13261	2.075741	0.506591	-1.261974
3	4	13106	1.874415	-0.345512	1.219671
4	5	6635	0.974194	-0.580790	0.212744

In [62]:
```
finaldf.reset_index(drop=True,inplace=True)
```

In [63]:
```
## k-fold
```

In [64]:
```
def get_stats(model,x_train,y_train,x_test,y_test):
    model.fit(x_train,y_train)
    y_pred = model.predict(x_test)
```

```python
    RMSE,MAE = error_cal(y_test,y_pred)
    return [RMSE,MAE]
```

In [65]:
```python
from sklearn.model_selection import StratifiedKFold
from sklearn.tree import DecisionTreeRegressor
dt = DecisionTreeRegressor()
```

In [66]:
```python
y = np.array(finaldf['Sales'])
x = np.array(finaldf.drop('Sales',axis=1))
```

In [67]:
```python
score_rdm = []
score_dt = []
kf = StratifiedKFold(n_splits=5,random_state=100)
for train_index,test_index in kf.split(x,y):
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.3,random_state=53)
    score_dt.append(get_stats(DecisionTreeRegressor(),x_train,y_train,x_test,y_test))
    score_rdm.append(get_stats(RandomForestRegressor(n_estimators=10),x_train,y_train,x_test,y_test))
```

In [68]:
```python
k_fold_df = pd.DataFrame()
k_fold_df['decision_tree']=pd.DataFrame(score_dt,columns=['RMSE','MAE']).mean(axis=0)
k_fold_df['Random_forest']=pd.DataFrame(score_rdm,columns=['RMSE','MAE']).mean(axis=0)
```

In [69]:
```python
k_fold_df
```

Out[69]:

	decision_tree	Random_forest
RMSE	1111.161370	934.724327
MAE	667.705973	604.585739

In [70]:
```python
## for individual stores
```

In [71]:
```python
## random forest model
RMSE_array_rdm_pca = []
MAE_array_rdm_pca = []
for store in range(1,12):
    data = finaldf[finaldf.Store==store]
    data.drop('Store',axis=1,inplace=True)
    y=np.array(data['Sales'])
    x=np.array(data.drop('Sales',axis=1))
    y_true,y_pred = random_forest(x,y)
    RMSE_1,MAE_1 = error_cal(y_true,y_pred)
    RMSE_array_rdm_pca.append(RMSE_1)
    MAE_array_rdm_pca.append(MAE_1)
```

In [72]:
```python
error_output_rdm_pca = pd.DataFrame()
error_output_rdm_pca['Stores'] = stores
error_output_rdm_pca['RMSE'] = RMSE_array_rdm_pca
error_output_rdm_pca['MAE'] = MAE_array_rdm_pca
error_output_rdm_pca
```

Out[72]:

	Stores	RMSE	MAE
0	1	359.741644	274.435304
1	2	473.322539	305.738169
2	3	506.315693	376.542205
3	4	682.783634	505.468094
4	5	531.106836	339.087485
5	6	484.833955	342.828062
6	7	833.074592	613.688458
7	8	428.007893	310.298831
8	9	505.587707	375.327604
9	10	389.369208	305.832056
10	11	739.210057	531.678240

In [73]:
```python
def decision_tree(x,y):
    dt = DecisionTreeRegressor()
    x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=36)
    dt.fit(x_train,y_train)
    y_pred = dt.predict(x_test)
    return y_test,y_pred
```

In [74]:
```python
## decision tree model
RMSE_array_dt_pca = []
MAE_array_dt_pca = []
for store in range(1,12):
    data = finaldf[finaldf.Store==store]
    data.drop('Store',axis=1,inplace=True)
    y=np.array(data['Sales'])
```

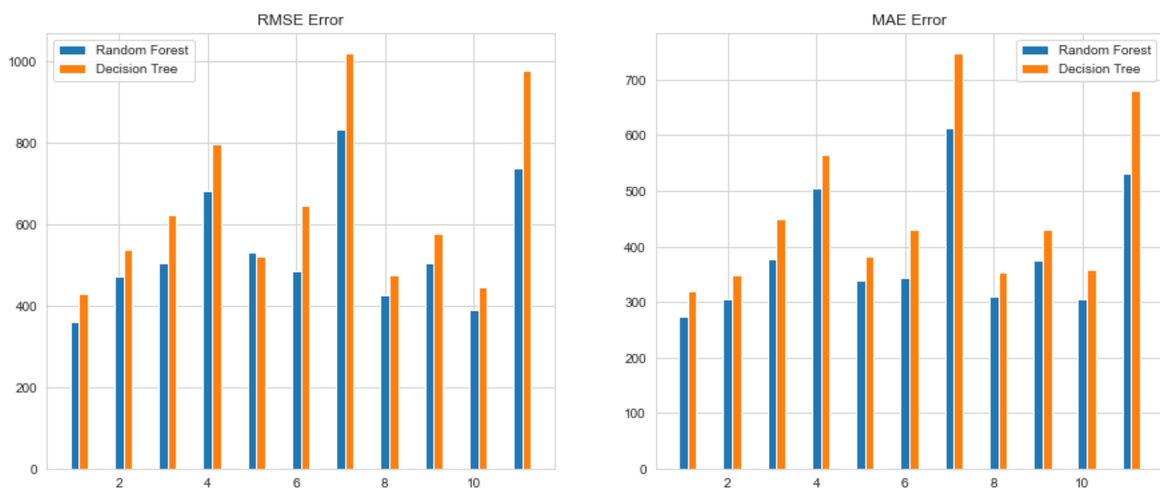
```
    x=np.array(data.drop('Sales',axis=1))
    y_true,y_pred = decision_tree(x,y)
    RMSE_1,MAE_1 = error_cal(y_true,y_pred)
    RMSE_array_dt_pca.append(RMSE_1)
    MAE_array_dt_pca.append(MAE_1)
```

In [75]:
```
error_output_dt_pca = pd.DataFrame()
error_output_dt_pca['Stores'] = stores
error_output_dt_pca['RMSE'] = RMSE_array_dt_pca
error_output_dt_pca['MAE'] = MAE_array_dt_pca
error_output_dt_pca
```

Out[75]:

	Stores	RMSE	MAE
0	1	429.814710	318.719577
1	2	539.282636	347.978070
2	3	623.503335	450.341312
3	4	798.853400	566.110526
4	5	522.576826	382.338652
5	6	645.673189	429.274250
6	7	1020.463707	747.373684
7	8	474.543395	353.057895
8	9	578.754419	430.102837
9	10	445.050558	358.442982
10	11	976.951531	679.260526

In [76]:
```
fig, axs = plt.subplots(1,2, figsize=(15,6))
fig.subplots_adjust(hspace=0.4)
axs=axs.ravel()
N=12
x=np.arange(1,N)
i=0
for col in ['RMSE','MAE']:
    axs[i].bar(x,height=error_output_rdm_pca[col],label = 'Random Forest',width=0.2)
    axs[i].bar(x+0.2,height=error_output_dt_pca[col],label = 'Decision Tree',width=0.2)
    axs[i].legend()
    axs[i].set_title(col+' Error')
    i+=1
```



In [77]:
```
print('Average RMSE Decision Tree Error: {}'.format(error_output_dt_pca.RMSE.mean()))
print('Average RMSE Random Forest Error: {}'.format(error_output_rdm_pca.RMSE.mean()))
```

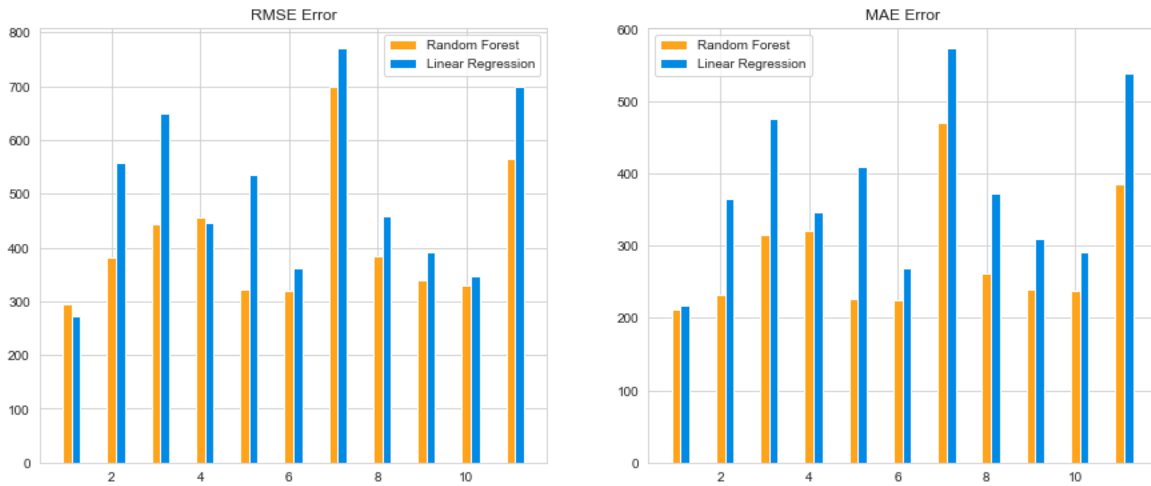
```
Average RMSE Decision Tree Error: 641.4061551719874
Average RMSE Random Forest Error: 539.3957964010076
```

3. Compare the performance of Linear Model and Non-Linear Model from the previous observations. Which performs better and why?

In [78]:
```
## compare the performance of linear and non linear model
```

In [83]:
```
fig, axs = plt.subplots(1,2, figsize=(15,6))
fig.subplots_adjust(hspace=0.4)
axs=axs.ravel()
N=12
x=np.arange(1,N)
i=0
for col in ['RMSE','MAE']:
    axs[i].bar(x,height=error_output_rdm[col],label = 'Random Forest',width=0.2,color = '#ffa31a')
```

```
    axs[i].bar(x+0.2,height=error_output_lrc[col],label = 'Linear Regression',width=0.2,color = '#008ae6')
    axs[i].legend()
    axs[i].set_title(col+' Error')
    i+=1
```



In [84]:
```
print('Average RMSE Random Forest Error: {}'.format(error_output_rdm.RMSE.mean()))
print('Average RMSE Linear Regression Error: {}'.format(error_output_lrc.RMSE.mean()))
```

```
Average RMSE Random Forest Error: 411.82760368071763
Average RMSE Linear Regression Error: 498.6368743223171
```

- From the above graph, it is clear that non-linear model i.e. random forest performs better than Linear Regression model.
- So we can say that our dataset is not liearly separable.

4. Train a Time-series model on the data taking time as the only feature. This will be a store-level training.

a) Identify yearly trends and seasonal months

In [85]:
```
## Time series analysis
```

In [86]:
```
def test_stationarity(timeseries):
    rolmean = timeseries.rolling(window=52,center = False).mean()
    rolstd = timeseries.rolling(window = 52,center = False).std()
    plt.figure(figsize=(16,8))
    orig = plt.plot(timeseries,color = '#3399ff',label = 'Original')
    mean = plt.plot(rolmean,color = 'red',label = 'Rolling Mean')
    std = plt.plot(rolstd,color = 'green',label = 'Rolling Std')
    plt.title('Rolling mean and Standard deviation')
    plt.legend(loc='best')
    plt.show(block=False)

    print('Result of Dickey-Fuller Test: ')
    dftest = adfuller(timeseries,autolag='AIC')
    dfoutput = pd.Series(dftest[0:4],index=['Test Statistic','p-value','Number of lag used','Number of observation used'])

    for key,value in dftest[4].items():
        dfoutput['Critical Value (%s)'%key] = value
    print(dfoutput)
```

In [87]:
```
original_data.head()
```

Out[87]:

	Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	2	2015-06-30	5735	568	1	1	0	0
1	2	2	2015-06-30	9863	877	1	1	0	0
2	3	2	2015-06-30	13261	1072	1	1	0	1
3	4	2	2015-06-30	13106	1488	1	1	0	0
4	5	2	2015-06-30	6635	645	1	1	0	0

In [88]:
```
original_data.sort_values('Date',inplace=True)
```

In [89]:
```
original_data = original_data[original_data.Open==1]
original_data.reset_index(drop = True,inplace=True)
```

In [90]:
```
## lets see the sales graph as per time.
```

In [91]:
```
datax = original_data[original_data.Store==1][['Date','Sales']]
datax.set_index('Date',inplace=True)
```

In [92]:
```
datax
```

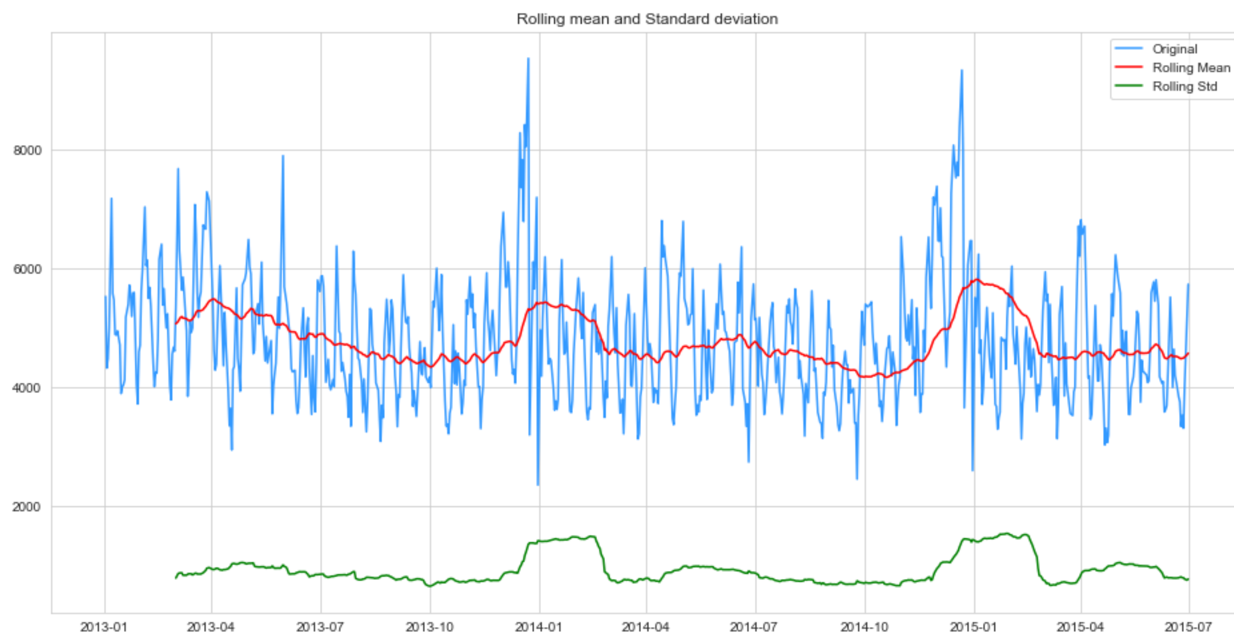
Out[92]:

Date	Sales
2013-01-02	5530
2013-01-03	4327
2013-01-04	4486
2013-01-05	4997
2013-01-07	7176
...	...
2015-06-25	3533
2015-06-26	3317
2015-06-27	4019
2015-06-29	5197
2015-06-30	5735

754 rows × 1 columns

In [93]:

```
test_stationarity(datax)
```

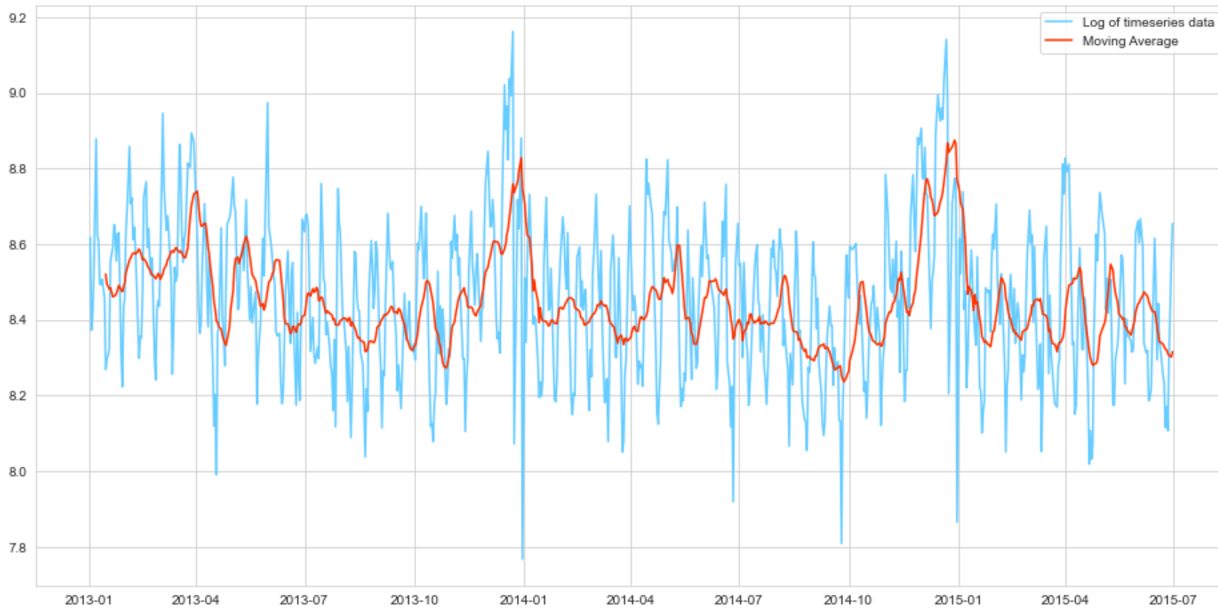



```
Result of Dickey-Fuller Test:
Test Statistic                 -5.336702
p-value                         0.000005
Number of lag used             13.000000
Number of observation used    740.000000
Critical Value (1%)            -3.439218
Critical Value (5%)            -2.865454
Critical Value (10%)           -2.568854
dtype: float64
```

- p-value is very close to zero so we will reject the null hypothesis, that data does not have a unit root and is stationary.
- However, data shows some seasonal effects.

In [94]:

```
ts_log = np.log(datax)
movingavg = ts_log.rolling(window = 12).mean()
plt.figure(figsize=(16,8))
plt.plot(ts_log,color='#66ccff',label = 'Log of timeseries data')
plt.plot(movingavg,color='#ff3300',label = 'Moving Average')
plt.legend()
plt.show()
```



- From the above graph we can see seasonal effect in the dataset.
- In dec to jan month sale is high in comaprison to other month

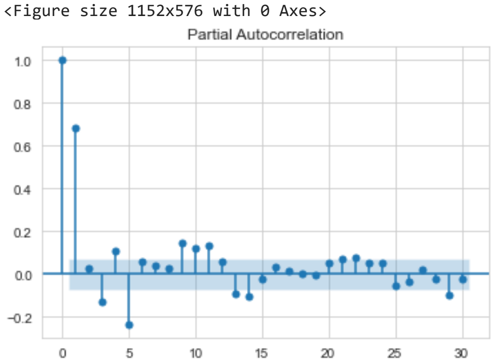
In [95]:
```
## time series model
```

In [96]:
```
ts_log_mv_diff = ts_log - movingavg
ts_log_mv_diff.dropna(inplace=True)
```

- Since p-value is less than 0.05, so we can say that data is stationary.
- hence differencing is not required, therefore d = 0.

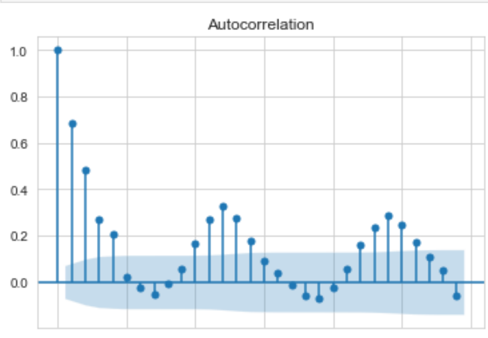
In [97]:
```
plt.figure(figsize=(16,8))
plot_pacf(datax.dropna(), lags=30)
plt.show()
```

<Figure size 1152x576 with 0 Axes>



- The first lag is the only one vastly above the signicance level and so p = 1.

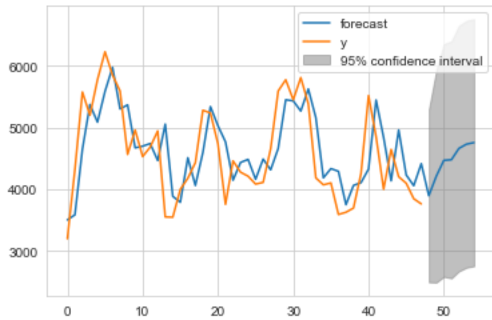
In [98]:
```
plot_acf(datax.dropna())
plt.show()
```



- Four lag can be found above the significance level and thus q = 4.

In [99]:
```
model = ARIMA(np.array(datax[:-6]), order=(1, 0, 4))
results = model.fit()
```

In [162…
```
results.plot_predict(700,754)
plt.show()
```



In [101…
```
results.summary()
```

Out[101…

ARMA Model Results

Dep. Variable:	y	**No. Observations:**	748
Model:	ARMA(1, 4)	**Log Likelihood**	-5977.693
Method:	css-mle	**S.D. of innovations**	714.770
Date:	Sat, 17 Oct 2020	**AIC**	11969.386
Time:	17:55:22	**BIC**	12001.707
Sample:	0	**HQIC**	11981.841

	coef	std err	z	P>|z|	[0.025	0.975]
const	4771.0302	81.286	58.694	0.000	4611.713	4930.348
ar.L1.y	0.3738	0.127	2.949	0.003	0.125	0.622
ma.L1.y	0.3420	0.127	2.689	0.007	0.093	0.591
ma.L2.y	0.2795	0.092	3.030	0.002	0.099	0.460
ma.L3.y	0.0544	0.062	0.878	0.380	-0.067	0.176
ma.L4.y	0.2763	0.035	8.002	0.000	0.209	0.344

Roots

	Real	Imaginary	Modulus	Frequency
AR.1	2.6754	+0.0000j	2.6754	0.0000
MA.1	-0.8914	-0.9326j	1.2901	-0.3714
MA.2	-0.8914	+0.9326j	1.2901	0.3714
MA.3	0.7930	-1.2434j	1.4748	-0.1596
MA.4	0.7930	+1.2434j	1.4748	0.1596

In [102…
```
RMSE_ARIMA = math.sqrt(mean_squared_error(np.array(datax[700:]) , results.predict(700,753)))
RMSE_ARIMA
```

Out[102… 587.1520321281363

In [103…
```
MAE_ARIMA = mean_absolute_error(np.array(datax[700:]) , results.predict(700,753))
MAE_ARIMA
```

Out[103… 482.5240578776619

- Similarly we can predict for other stores

Project Task: Week 3

Implementing Neural Networks:

1. Train a LSTM on the same set of features and compare the result with traditional time-series model.

In [104…
```
std = datax.std()
mean = datax.mean()
timeseries = np.array((datax-mean)/std)
```

In [105…
```
training_size = int(len(timeseries)*0.65)
test_size = len(timeseries)-training_size
```

```
train_size,test_size = timeseries[:training_size,:],timeseries[training_size:len(timeseries),:1]
```

In [106…

```
def create_dataset(dataset,time_step = 1):
    dataX,dataY = [],[]
    for i in range(len(dataset)-time_step-1):
        a = dataset[i:(i+time_step),0]
        dataX.append(a)
        dataY.append(dataset[i+time_step,0])
    return np.array(dataX),np.array(dataY)
```

In [107…

```
time_step =100
x_train,y_train = create_dataset(train_size,time_step)
x_test,y_test = create_dataset(test_size,time_step)
```

In [108…

```
x_train = x_train.reshape(x_train.shape[0],x_train.shape[1],1)
x_test = x_test.reshape(x_test.shape[0],x_test.shape[1],1)
```

In [109…

```
model = Sequential()
model.add(LSTM(50,return_sequences = True,input_shape = (100,1)))
model.add(LSTM(50,return_sequences = True))
model.add(LSTM(50))
model.add(Dense(1))
model.compile(loss='mean_squared_error',optimizer = 'adam')
```

In [110…

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 100, 50)           10400
_________________________________________________________________
lstm_1 (LSTM)                (None, 100, 50)           20200
_________________________________________________________________
lstm_2 (LSTM)                (None, 50)                20200
_________________________________________________________________
dense (Dense)                (None, 1)                 51
=================================================================
Total params: 50,851
Trainable params: 50,851
Non-trainable params: 0
_________________________________________________________________
```

In [111…

```
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
checkpoint = ModelCheckpoint('Sales.h5',
                            monitor='loss',
                            mode=min,
                            save_best_only=True,
                            verbose=1)

early_stopping = EarlyStopping(monitor='loss',
                              patience=9,
                              min_delta=0,
                              restore_best_weights=True,
                              verbose=1)

Reduce_ler_rate = ReduceLROnPlateau(monitor='loss',
                                   factor=0.2,
                                   patience=3,
                                   verbose=1,
                                   min_delta=0.001)

callback = [checkpoint,early_stopping,Reduce_ler_rate]
```

```
WARNING:tensorflow:ModelCheckpoint mode <built-in function min> is unknown, fallback to auto mode.
```

In [112…

```
history = model.fit(x_train,y_train,validation_data=(x_test,y_test),epochs=200,batch_size=64,verbose=1,callbacks=callback)
```

```
Epoch 1/200
7/7 [==============================] - ETA: 0s - loss: 0.8899
Epoch 00001: loss improved from inf to 0.88987, saving model to Sales.h5
7/7 [==============================] - 2s 265ms/step - loss: 0.8899 - val_loss: 1.1912
Epoch 2/200
7/7 [==============================] - ETA: 0s - loss: 0.8719
Epoch 00002: loss improved from 0.88987 to 0.87194, saving model to Sales.h5
7/7 [==============================] - 1s 110ms/step - loss: 0.8719 - val_loss: 1.2051
Epoch 3/200
7/7 [==============================] - ETA: 0s - loss: 0.8729
Epoch 00003: loss did not improve from 0.87194
7/7 [==============================] - 1s 86ms/step - loss: 0.8729 - val_loss: 1.2066
Epoch 4/200
7/7 [==============================] - ETA: 0s - loss: 0.8680
Epoch 00004: loss improved from 0.87194 to 0.86797, saving model to Sales.h5
7/7 [==============================] - 1s 113ms/step - loss: 0.8680 - val_loss: 1.1481
Epoch 5/200
7/7 [==============================] - ETA: 0s - loss: 0.8658
Epoch 00005: loss improved from 0.86797 to 0.86578, saving model to Sales.h5
7/7 [==============================] - 1s 103ms/step - loss: 0.8658 - val_loss: 1.1260
Epoch 6/200
7/7 [==============================] - ETA: 0s - loss: 0.8594
Epoch 00006: loss improved from 0.86578 to 0.85941, saving model to Sales.h5
7/7 [==============================] - 1s 103ms/step - loss: 0.8594 - val_loss: 1.1050
Epoch 7/200
```

```
7/7 [==============================] - ETA: 0s - loss: 0.8563
Epoch 00007: loss improved from 0.85941 to 0.85630, saving model to Sales.h5
7/7 [==============================] - 1s 104ms/step - loss: 0.8563 - val_loss: 1.0524
Epoch 8/200
7/7 [==============================] - ETA: 0s - loss: 0.8442
Epoch 00008: loss improved from 0.85630 to 0.84421, saving model to Sales.h5
7/7 [==============================] - 1s 102ms/step - loss: 0.8442 - val_loss: 1.0099
Epoch 9/200
7/7 [==============================] - ETA: 0s - loss: 0.8226
Epoch 00009: loss improved from 0.84421 to 0.82262, saving model to Sales.h5
7/7 [==============================] - 1s 105ms/step - loss: 0.8226 - val_loss: 1.0283
Epoch 10/200
7/7 [==============================] - ETA: 0s - loss: 0.8110
Epoch 00010: loss improved from 0.82262 to 0.81099, saving model to Sales.h5
7/7 [==============================] - 1s 112ms/step - loss: 0.8110 - val_loss: 1.0177
Epoch 11/200
7/7 [==============================] - ETA: 0s - loss: 0.7889
Epoch 00011: loss improved from 0.81099 to 0.78894, saving model to Sales.h5
7/7 [==============================] - 1s 103ms/step - loss: 0.7889 - val_loss: 1.3343
Epoch 12/200
7/7 [==============================] - ETA: 0s - loss: 0.8475
Epoch 00012: loss did not improve from 0.78894
7/7 [==============================] - 1s 89ms/step - loss: 0.8475 - val_loss: 1.2738
Epoch 13/200
7/7 [==============================] - ETA: 0s - loss: 0.8329
Epoch 00013: loss did not improve from 0.78894
7/7 [==============================] - 1s 88ms/step - loss: 0.8329 - val_loss: 1.1761
Epoch 14/200
7/7 [==============================] - ETA: 0s - loss: 0.8119
Epoch 00014: loss did not improve from 0.78894

Epoch 00014: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
7/7 [==============================] - 1s 91ms/step - loss: 0.8119 - val_loss: 1.1040
Epoch 15/200
7/7 [==============================] - ETA: 0s - loss: 0.7954
Epoch 00015: loss did not improve from 0.78894
7/7 [==============================] - 1s 88ms/step - loss: 0.7954 - val_loss: 1.0993
Epoch 16/200
7/7 [==============================] - ETA: 0s - loss: 0.7904
Epoch 00016: loss did not improve from 0.78894
7/7 [==============================] - 1s 92ms/step - loss: 0.7904 - val_loss: 1.0888
Epoch 17/200
7/7 [==============================] - ETA: 0s - loss: 0.7849
Epoch 00017: loss improved from 0.78894 to 0.78492, saving model to Sales.h5
7/7 [==============================] - 1s 106ms/step - loss: 0.7849 - val_loss: 1.0715
Epoch 18/200
7/7 [==============================] - ETA: 0s - loss: 0.7776
Epoch 00018: loss improved from 0.78492 to 0.77764, saving model to Sales.h5
7/7 [==============================] - 1s 106ms/step - loss: 0.7776 - val_loss: 1.0556
Epoch 19/200
7/7 [==============================] - ETA: 0s - loss: 0.7718
Epoch 00019: loss improved from 0.77764 to 0.77184, saving model to Sales.h5
7/7 [==============================] - 1s 109ms/step - loss: 0.7718 - val_loss: 1.0371
Epoch 20/200
7/7 [==============================] - ETA: 0s - loss: 0.7637
Epoch 00020: loss improved from 0.77184 to 0.76366, saving model to Sales.h5
7/7 [==============================] - 1s 101ms/step - loss: 0.7637 - val_loss: 0.9946
Epoch 21/200
7/7 [==============================] - ETA: 0s - loss: 0.7566
Epoch 00021: loss improved from 0.76366 to 0.75663, saving model to Sales.h5
7/7 [==============================] - 1s 103ms/step - loss: 0.7566 - val_loss: 0.9788
Epoch 22/200
7/7 [==============================] - ETA: 0s - loss: 0.7524
Epoch 00022: loss improved from 0.75663 to 0.75244, saving model to Sales.h5
7/7 [==============================] - 1s 102ms/step - loss: 0.7524 - val_loss: 0.9871
Epoch 23/200
7/7 [==============================] - ETA: 0s - loss: 0.7502
Epoch 00023: loss improved from 0.75244 to 0.75019, saving model to Sales.h5
7/7 [==============================] - 1s 104ms/step - loss: 0.7502 - val_loss: 0.9835
Epoch 24/200
7/7 [==============================] - ETA: 0s - loss: 0.7444
Epoch 00024: loss improved from 0.75019 to 0.74441, saving model to Sales.h5
7/7 [==============================] - 1s 102ms/step - loss: 0.7444 - val_loss: 0.9783
Epoch 25/200
7/7 [==============================] - ETA: 0s - loss: 0.7393
Epoch 00025: loss improved from 0.74441 to 0.73926, saving model to Sales.h5
7/7 [==============================] - 1s 101ms/step - loss: 0.7393 - val_loss: 0.9655
Epoch 26/200
7/7 [==============================] - ETA: 0s - loss: 0.7325
Epoch 00026: loss improved from 0.73926 to 0.73251, saving model to Sales.h5
7/7 [==============================] - 1s 110ms/step - loss: 0.7325 - val_loss: 0.9635
Epoch 27/200
7/7 [==============================] - ETA: 0s - loss: 0.7310
Epoch 00027: loss improved from 0.73251 to 0.73097, saving model to Sales.h5
7/7 [==============================] - 1s 112ms/step - loss: 0.7310 - val_loss: 0.9618
Epoch 28/200
7/7 [==============================] - ETA: 0s - loss: 0.7286
Epoch 00028: loss improved from 0.73097 to 0.72858, saving model to Sales.h5
7/7 [==============================] - 1s 121ms/step - loss: 0.7286 - val_loss: 0.9601
Epoch 29/200
7/7 [==============================] - ETA: 0s - loss: 0.7261
Epoch 00029: loss improved from 0.72858 to 0.72613, saving model to Sales.h5
7/7 [==============================] - 1s 119ms/step - loss: 0.7261 - val_loss: 0.9589
Epoch 30/200
7/7 [==============================] - ETA: 0s - loss: 0.7223
Epoch 00030: loss improved from 0.72613 to 0.72234, saving model to Sales.h5
7/7 [==============================] - 1s 103ms/step - loss: 0.7223 - val_loss: 0.9560
Epoch 31/200
```

```
7/7 [==============================] - ETA: 0s - loss: 0.7183
Epoch 00031: loss improved from 0.72234 to 0.71832, saving model to Sales.h5
7/7 [==============================] - 1s 118ms/step - loss: 0.7183 - val_loss: 0.9447
Epoch 32/200
7/7 [==============================] - ETA: 0s - loss: 0.7160
Epoch 00032: loss improved from 0.71832 to 0.71596, saving model to Sales.h5
7/7 [==============================] - 1s 123ms/step - loss: 0.7160 - val_loss: 0.9133
Epoch 33/200
7/7 [==============================] - ETA: 0s - loss: 0.7133
Epoch 00033: loss improved from 0.71596 to 0.71327, saving model to Sales.h5
7/7 [==============================] - 1s 125ms/step - loss: 0.7133 - val_loss: 0.8909
Epoch 34/200
7/7 [==============================] - ETA: 0s - loss: 0.7120
Epoch 00034: loss improved from 0.71327 to 0.71196, saving model to Sales.h5
7/7 [==============================] - 1s 111ms/step - loss: 0.7120 - val_loss: 0.8364
Epoch 35/200
7/7 [==============================] - ETA: 0s - loss: 0.7088
Epoch 00035: loss improved from 0.71196 to 0.70879, saving model to Sales.h5
7/7 [==============================] - 1s 104ms/step - loss: 0.7088 - val_loss: 0.8224
Epoch 36/200
7/7 [==============================] - ETA: 0s - loss: 0.7048
Epoch 00036: loss improved from 0.70879 to 0.70479, saving model to Sales.h5
7/7 [==============================] - 1s 109ms/step - loss: 0.7048 - val_loss: 0.8288
Epoch 37/200
7/7 [==============================] - ETA: 0s - loss: 0.7008
Epoch 00037: loss improved from 0.70479 to 0.70083, saving model to Sales.h5
7/7 [==============================] - 1s 143ms/step - loss: 0.7008 - val_loss: 0.8513
Epoch 38/200
7/7 [==============================] - ETA: 0s - loss: 0.6965
Epoch 00038: loss improved from 0.70083 to 0.69650, saving model to Sales.h5
7/7 [==============================] - 1s 138ms/step - loss: 0.6965 - val_loss: 0.8539
Epoch 39/200
7/7 [==============================] - ETA: 0s - loss: 0.6965
Epoch 00039: loss improved from 0.69650 to 0.69650, saving model to Sales.h5
7/7 [==============================] - 1s 127ms/step - loss: 0.6965 - val_loss: 0.8364
Epoch 40/200
7/7 [==============================] - ETA: 0s - loss: 0.6927
Epoch 00040: loss improved from 0.69650 to 0.69274, saving model to Sales.h5
7/7 [==============================] - 1s 114ms/step - loss: 0.6927 - val_loss: 0.8512
Epoch 41/200
7/7 [==============================] - ETA: 0s - loss: 0.6897
Epoch 00041: loss improved from 0.69274 to 0.68968, saving model to Sales.h5
7/7 [==============================] - 1s 113ms/step - loss: 0.6897 - val_loss: 0.8939
Epoch 42/200
7/7 [==============================] - ETA: 0s - loss: 0.6882
Epoch 00042: loss improved from 0.68968 to 0.68818, saving model to Sales.h5
7/7 [==============================] - 1s 107ms/step - loss: 0.6882 - val_loss: 0.9209
Epoch 43/200
7/7 [==============================] - ETA: 0s - loss: 0.6829
Epoch 00043: loss improved from 0.68818 to 0.68292, saving model to Sales.h5
7/7 [==============================] - 1s 125ms/step - loss: 0.6829 - val_loss: 0.9187
Epoch 44/200
7/7 [==============================] - ETA: 0s - loss: 0.6799
Epoch 00044: loss improved from 0.68292 to 0.67988, saving model to Sales.h5
7/7 [==============================] - 1s 125ms/step - loss: 0.6799 - val_loss: 0.9117
Epoch 45/200
7/7 [==============================] - ETA: 0s - loss: 0.6784
Epoch 00045: loss improved from 0.67988 to 0.67841, saving model to Sales.h5
7/7 [==============================] - 1s 105ms/step - loss: 0.6784 - val_loss: 0.8836
Epoch 46/200
7/7 [==============================] - ETA: 0s - loss: 0.6741
Epoch 00046: loss improved from 0.67841 to 0.67413, saving model to Sales.h5
7/7 [==============================] - 1s 105ms/step - loss: 0.6741 - val_loss: 0.8530
Epoch 47/200
7/7 [==============================] - ETA: 0s - loss: 0.6719
Epoch 00047: loss improved from 0.67413 to 0.67191, saving model to Sales.h5
7/7 [==============================] - 1s 123ms/step - loss: 0.6719 - val_loss: 0.8480
Epoch 48/200
7/7 [==============================] - ETA: 0s - loss: 0.6699
Epoch 00048: loss improved from 0.67191 to 0.66991, saving model to Sales.h5
7/7 [==============================] - 1s 116ms/step - loss: 0.6699 - val_loss: 0.8352
Epoch 49/200
7/7 [==============================] - ETA: 0s - loss: 0.6657
Epoch 00049: loss improved from 0.66991 to 0.66570, saving model to Sales.h5
7/7 [==============================] - 1s 125ms/step - loss: 0.6657 - val_loss: 0.8282
Epoch 50/200
7/7 [==============================] - ETA: 0s - loss: 0.6607
Epoch 00050: loss improved from 0.66570 to 0.66074, saving model to Sales.h5
7/7 [==============================] - 1s 104ms/step - loss: 0.6607 - val_loss: 0.8236
Epoch 51/200
7/7 [==============================] - ETA: 0s - loss: 0.6565
Epoch 00051: loss improved from 0.66074 to 0.65650, saving model to Sales.h5
7/7 [==============================] - 1s 112ms/step - loss: 0.6565 - val_loss: 0.8395
Epoch 52/200
7/7 [==============================] - ETA: 0s - loss: 0.6587
Epoch 00052: loss did not improve from 0.65650
7/7 [==============================] - 1s 93ms/step - loss: 0.6587 - val_loss: 0.8278
Epoch 53/200
7/7 [==============================] - ETA: 0s - loss: 0.6537
Epoch 00053: loss improved from 0.65650 to 0.65370, saving model to Sales.h5
7/7 [==============================] - 1s 109ms/step - loss: 0.6537 - val_loss: 0.8630
Epoch 54/200
7/7 [==============================] - ETA: 0s - loss: 0.6506
Epoch 00054: loss improved from 0.65370 to 0.65065, saving model to Sales.h5
7/7 [==============================] - 1s 98ms/step - loss: 0.6506 - val_loss: 0.8558
Epoch 55/200
7/7 [==============================] - ETA: 0s - loss: 0.6402
Epoch 00055: loss improved from 0.65065 to 0.64018, saving model to Sales.h5
```

```
7/7 [==============================] - 1s 123ms/step - loss: 0.6402 - val_loss: 0.8252
Epoch 56/200
7/7 [==============================] - ETA: 0s - loss: 0.6480
Epoch 00056: loss did not improve from 0.64018
7/7 [==============================] - 1s 105ms/step - loss: 0.6480 - val_loss: 0.8421
Epoch 57/200
7/7 [==============================] - ETA: 0s - loss: 0.6384
Epoch 00057: loss improved from 0.64018 to 0.63840, saving model to Sales.h5
7/7 [==============================] - 1s 106ms/step - loss: 0.6384 - val_loss: 0.8539
Epoch 58/200
6/7 [========================>.....] - ETA: 0s - loss: 0.6309
Epoch 00058: loss improved from 0.63840 to 0.63305, saving model to Sales.h5
7/7 [==============================] - 1s 109ms/step - loss: 0.6331 - val_loss: 0.8599
Epoch 59/200
7/7 [==============================] - ETA: 0s - loss: 0.6297
Epoch 00059: loss improved from 0.63305 to 0.62968, saving model to Sales.h5
7/7 [==============================] - 1s 108ms/step - loss: 0.6297 - val_loss: 0.9302
Epoch 60/200
7/7 [==============================] - ETA: 0s - loss: 0.6379
Epoch 00060: loss did not improve from 0.62968
7/7 [==============================] - 1s 88ms/step - loss: 0.6379 - val_loss: 0.9653
Epoch 61/200
7/7 [==============================] - ETA: 0s - loss: 0.6312
Epoch 00061: loss did not improve from 0.62968
7/7 [==============================] - 1s 91ms/step - loss: 0.6312 - val_loss: 0.9333
Epoch 62/200
7/7 [==============================] - ETA: 0s - loss: 0.6245
Epoch 00062: loss improved from 0.62968 to 0.62455, saving model to Sales.h5
7/7 [==============================] - 1s 110ms/step - loss: 0.6245 - val_loss: 0.8964
Epoch 63/200
7/7 [==============================] - ETA: 0s - loss: 0.6266
Epoch 00063: loss did not improve from 0.62455
7/7 [==============================] - 1s 93ms/step - loss: 0.6266 - val_loss: 0.9084
Epoch 64/200
7/7 [==============================] - ETA: 0s - loss: 0.6236
Epoch 00064: loss improved from 0.62455 to 0.62357, saving model to Sales.h5
7/7 [==============================] - 1s 113ms/step - loss: 0.6236 - val_loss: 0.9309
Epoch 65/200
7/7 [==============================] - ETA: 0s - loss: 0.6168
Epoch 00065: loss improved from 0.62357 to 0.61681, saving model to Sales.h5
7/7 [==============================] - 1s 104ms/step - loss: 0.6168 - val_loss: 0.9249
Epoch 66/200
7/7 [==============================] - ETA: 0s - loss: 0.6167
Epoch 00066: loss improved from 0.61681 to 0.61667, saving model to Sales.h5
7/7 [==============================] - 1s 102ms/step - loss: 0.6167 - val_loss: 0.9275
Epoch 67/200
7/7 [==============================] - ETA: 0s - loss: 0.6120
Epoch 00067: loss improved from 0.61667 to 0.61195, saving model to Sales.h5
7/7 [==============================] - 1s 102ms/step - loss: 0.6120 - val_loss: 0.9249
Epoch 68/200
7/7 [==============================] - ETA: 0s - loss: 0.6096
Epoch 00068: loss improved from 0.61195 to 0.60962, saving model to Sales.h5
7/7 [==============================] - 1s 106ms/step - loss: 0.6096 - val_loss: 0.9306
Epoch 69/200
7/7 [==============================] - ETA: 0s - loss: 0.6113
Epoch 00069: loss did not improve from 0.60962
7/7 [==============================] - 1s 94ms/step - loss: 0.6113 - val_loss: 0.9415
Epoch 70/200
7/7 [==============================] - ETA: 0s - loss: 0.6038
Epoch 00070: loss improved from 0.60962 to 0.60376, saving model to Sales.h5
7/7 [==============================] - 1s 107ms/step - loss: 0.6038 - val_loss: 0.8950
Epoch 71/200
7/7 [==============================] - ETA: 0s - loss: 0.5995
Epoch 00071: loss improved from 0.60376 to 0.59955, saving model to Sales.h5
7/7 [==============================] - 1s 104ms/step - loss: 0.5995 - val_loss: 0.8980
Epoch 72/200
7/7 [==============================] - ETA: 0s - loss: 0.5963
Epoch 00072: loss improved from 0.59955 to 0.59629, saving model to Sales.h5
7/7 [==============================] - 1s 120ms/step - loss: 0.5963 - val_loss: 0.9016
Epoch 73/200
7/7 [==============================] - ETA: 0s - loss: 0.5995
Epoch 00073: loss did not improve from 0.59629
7/7 [==============================] - 1s 91ms/step - loss: 0.5995 - val_loss: 0.9510
Epoch 74/200
7/7 [==============================] - ETA: 0s - loss: 0.6031
Epoch 00074: loss did not improve from 0.59629
7/7 [==============================] - 1s 116ms/step - loss: 0.6031 - val_loss: 0.9183
Epoch 75/200
7/7 [==============================] - ETA: 0s - loss: 0.5924
Epoch 00075: loss improved from 0.59629 to 0.59237, saving model to Sales.h5
7/7 [==============================] - 1s 120ms/step - loss: 0.5924 - val_loss: 0.8873
Epoch 76/200
7/7 [==============================] - ETA: 0s - loss: 0.5891
Epoch 00076: loss improved from 0.59237 to 0.58912, saving model to Sales.h5
7/7 [==============================] - 1s 101ms/step - loss: 0.5891 - val_loss: 0.8684
Epoch 77/200
7/7 [==============================] - ETA: 0s - loss: 0.5815
Epoch 00077: loss improved from 0.58912 to 0.58149, saving model to Sales.h5
7/7 [==============================] - 1s 107ms/step - loss: 0.5815 - val_loss: 0.8680
Epoch 78/200
7/7 [==============================] - ETA: 0s - loss: 0.5752
Epoch 00078: loss improved from 0.58149 to 0.57521, saving model to Sales.h5
7/7 [==============================] - 1s 96ms/step - loss: 0.5752 - val_loss: 0.8882
Epoch 79/200
7/7 [==============================] - ETA: 0s - loss: 0.5760
Epoch 00079: loss did not improve from 0.57521
7/7 [==============================] - 1s 92ms/step - loss: 0.5760 - val_loss: 0.9268
Epoch 80/200
```

```
7/7 [==============================] - ETA: 0s - loss: 0.5733
Epoch 00080: loss improved from 0.57521 to 0.57334, saving model to Sales.h5
7/7 [==============================] - 1s 122ms/step - loss: 0.5733 - val_loss: 0.9187
Epoch 81/200
7/7 [==============================] - ETA: 0s - loss: 0.5730
Epoch 00081: loss improved from 0.57334 to 0.57297, saving model to Sales.h5
7/7 [==============================] - 1s 119ms/step - loss: 0.5730 - val_loss: 0.9322
Epoch 82/200
7/7 [==============================] - ETA: 0s - loss: 0.5926
Epoch 00082: loss did not improve from 0.57297
7/7 [==============================] - 1s 93ms/step - loss: 0.5926 - val_loss: 0.9129
Epoch 83/200
7/7 [==============================] - ETA: 0s - loss: 0.5940
Epoch 00083: loss did not improve from 0.57297

Epoch 00083: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.
7/7 [==============================] - 1s 98ms/step - loss: 0.5940 - val_loss: 0.9648
Epoch 84/200
7/7 [==============================] - ETA: 0s - loss: 0.5785
Epoch 00084: loss did not improve from 0.57297
7/7 [==============================] - 1s 101ms/step - loss: 0.5785 - val_loss: 0.9748
Epoch 85/200
7/7 [==============================] - ETA: 0s - loss: 0.5761
Epoch 00085: loss did not improve from 0.57297
7/7 [==============================] - 1s 97ms/step - loss: 0.5761 - val_loss: 0.9933
Epoch 86/200
7/7 [==============================] - ETA: 0s - loss: 0.5705
Epoch 00086: loss improved from 0.57297 to 0.57055, saving model to Sales.h5
7/7 [==============================] - 1s 118ms/step - loss: 0.5705 - val_loss: 1.0017
Epoch 87/200
7/7 [==============================] - ETA: 0s - loss: 0.5679
Epoch 00087: loss improved from 0.57055 to 0.56791, saving model to Sales.h5
7/7 [==============================] - 1s 97ms/step - loss: 0.5679 - val_loss: 1.0068
Epoch 88/200
7/7 [==============================] - ETA: 0s - loss: 0.5669
Epoch 00088: loss improved from 0.56791 to 0.56690, saving model to Sales.h5
7/7 [==============================] - 1s 102ms/step - loss: 0.5669 - val_loss: 1.0236
Epoch 89/200
7/7 [==============================] - ETA: 0s - loss: 0.5678
Epoch 00089: loss did not improve from 0.56690
7/7 [==============================] - 1s 101ms/step - loss: 0.5678 - val_loss: 1.0296
Epoch 90/200
7/7 [==============================] - ETA: 0s - loss: 0.5662
Epoch 00090: loss improved from 0.56690 to 0.56621, saving model to Sales.h5
7/7 [==============================] - 1s 105ms/step - loss: 0.5662 - val_loss: 1.0267
Epoch 91/200
7/7 [==============================] - ETA: 0s - loss: 0.5644
Epoch 00091: loss improved from 0.56621 to 0.56444, saving model to Sales.h5
7/7 [==============================] - 1s 105ms/step - loss: 0.5644 - val_loss: 1.0243
Epoch 92/200
7/7 [==============================] - ETA: 0s - loss: 0.5628
Epoch 00092: loss improved from 0.56444 to 0.56278, saving model to Sales.h5
7/7 [==============================] - 1s 108ms/step - loss: 0.5628 - val_loss: 1.0227
Epoch 93/200
7/7 [==============================] - ETA: 0s - loss: 0.5615
Epoch 00093: loss improved from 0.56278 to 0.56155, saving model to Sales.h5
7/7 [==============================] - 1s 112ms/step - loss: 0.5615 - val_loss: 1.0234
Epoch 94/200
7/7 [==============================] - ETA: 0s - loss: 0.5604
Epoch 00094: loss improved from 0.56155 to 0.56037, saving model to Sales.h5
7/7 [==============================] - 1s 116ms/step - loss: 0.5604 - val_loss: 1.0236
Epoch 95/200
7/7 [==============================] - ETA: 0s - loss: 0.5605
Epoch 00095: loss did not improve from 0.56037
7/7 [==============================] - 1s 93ms/step - loss: 0.5605 - val_loss: 1.0218
Epoch 96/200
7/7 [==============================] - ETA: 0s - loss: 0.5589
Epoch 00096: loss improved from 0.56037 to 0.55895, saving model to Sales.h5
7/7 [==============================] - 1s 122ms/step - loss: 0.5589 - val_loss: 1.0247
Epoch 97/200
7/7 [==============================] - ETA: 0s - loss: 0.5579
Epoch 00097: loss improved from 0.55895 to 0.55794, saving model to Sales.h5
7/7 [==============================] - 1s 107ms/step - loss: 0.5579 - val_loss: 1.0288
Epoch 98/200
7/7 [==============================] - ETA: 0s - loss: 0.5569
Epoch 00098: loss improved from 0.55794 to 0.55692, saving model to Sales.h5
7/7 [==============================] - 1s 95ms/step - loss: 0.5569 - val_loss: 1.0356
Epoch 99/200
7/7 [==============================] - ETA: 0s - loss: 0.5567
Epoch 00099: loss improved from 0.55692 to 0.55669, saving model to Sales.h5
7/7 [==============================] - 1s 92ms/step - loss: 0.5567 - val_loss: 1.0351
Epoch 100/200
7/7 [==============================] - ETA: 0s - loss: 0.5564
Epoch 00100: loss improved from 0.55669 to 0.55637, saving model to Sales.h5
7/7 [==============================] - 1s 102ms/step - loss: 0.5564 - val_loss: 1.0434
Epoch 101/200
7/7 [==============================] - ETA: 0s - loss: 0.5560
Epoch 00101: loss improved from 0.55637 to 0.55596, saving model to Sales.h5

Epoch 00101: ReduceLROnPlateau reducing learning rate to 8.000000525498762e-06.
7/7 [==============================] - 1s 107ms/step - loss: 0.5560 - val_loss: 1.0474
Epoch 102/200
7/7 [==============================] - ETA: 0s - loss: 0.5550
Epoch 00102: loss improved from 0.55596 to 0.55504, saving model to Sales.h5
7/7 [==============================] - 1s 104ms/step - loss: 0.5550 - val_loss: 1.0464
Epoch 103/200
7/7 [==============================] - ETA: 0s - loss: 0.5547
Epoch 00103: loss improved from 0.55504 to 0.55468, saving model to Sales.h5
```

```
7/7 [==============================] - 1s 111ms/step - loss: 0.5547 - val_loss: 1.0433
Epoch 104/200
7/7 [==============================] - ETA: 0s - loss: 0.5542
Epoch 00104: loss improved from 0.55468 to 0.55424, saving model to Sales.h5
7/7 [==============================] - 1s 106ms/step - loss: 0.5542 - val_loss: 1.0402
Epoch 105/200
7/7 [==============================] - ETA: 0s - loss: 0.5538
Epoch 00105: loss improved from 0.55424 to 0.55383, saving model to Sales.h5
7/7 [==============================] - 1s 107ms/step - loss: 0.5538 - val_loss: 1.0382
Epoch 106/200
7/7 [==============================] - ETA: 0s - loss: 0.5535
Epoch 00106: loss improved from 0.55383 to 0.55349, saving model to Sales.h5
7/7 [==============================] - 1s 107ms/step - loss: 0.5535 - val_loss: 1.0359
Epoch 107/200
7/7 [==============================] - ETA: 0s - loss: 0.5531
Epoch 00107: loss improved from 0.55349 to 0.55310, saving model to Sales.h5
7/7 [==============================] - 1s 169ms/step - loss: 0.5531 - val_loss: 1.0347
Epoch 108/200
7/7 [==============================] - ETA: 0s - loss: 0.5530
Epoch 00108: loss improved from 0.55310 to 0.55299, saving model to Sales.h5

Epoch 00108: ReduceLROnPlateau reducing learning rate to 1.6000001778593287e-06.
7/7 [==============================] - 1s 112ms/step - loss: 0.5530 - val_loss: 1.0337
Epoch 109/200
7/7 [==============================] - ETA: 0s - loss: 0.5526
Epoch 00109: loss improved from 0.55299 to 0.55258, saving model to Sales.h5
7/7 [==============================] - 1s 114ms/step - loss: 0.5526 - val_loss: 1.0336
Epoch 110/200
7/7 [==============================] - ETA: 0s - loss: 0.5526
Epoch 00110: loss improved from 0.55258 to 0.55257, saving model to Sales.h5
7/7 [==============================] - 1s 96ms/step - loss: 0.5526 - val_loss: 1.0338
Epoch 111/200
7/7 [==============================] - ETA: 0s - loss: 0.5524
Epoch 00111: loss improved from 0.55257 to 0.55243, saving model to Sales.h5
7/7 [==============================] - 1s 106ms/step - loss: 0.5524 - val_loss: 1.0350
Epoch 112/200
7/7 [==============================] - ETA: 0s - loss: 0.5526
Epoch 00112: loss did not improve from 0.55243

Epoch 00112: ReduceLROnPlateau reducing learning rate to 3.200000264769187e-07.
7/7 [==============================] - 1s 90ms/step - loss: 0.5526 - val_loss: 1.0356
Epoch 113/200
7/7 [==============================] - ETA: 0s - loss: 0.5525
Epoch 00113: loss did not improve from 0.55243
7/7 [==============================] - 1s 92ms/step - loss: 0.5525 - val_loss: 1.0356
Epoch 114/200
7/7 [==============================] - ETA: 0s - loss: 0.5525
Epoch 00114: loss did not improve from 0.55243
7/7 [==============================] - 1s 84ms/step - loss: 0.5525 - val_loss: 1.0355
Epoch 115/200
7/7 [==============================] - ETA: 0s - loss: 0.5525
Epoch 00115: loss did not improve from 0.55243

Epoch 00115: ReduceLROnPlateau reducing learning rate to 6.400000529538374e-08.
7/7 [==============================] - 1s 92ms/step - loss: 0.5525 - val_loss: 1.0354
Epoch 116/200
7/7 [==============================] - ETA: 0s - loss: 0.5525
Epoch 00116: loss did not improve from 0.55243
7/7 [==============================] - 1s 89ms/step - loss: 0.5525 - val_loss: 1.0354
Epoch 117/200
7/7 [==============================] - ETA: 0s - loss: 0.5525
Epoch 00117: loss did not improve from 0.55243
7/7 [==============================] - 1s 93ms/step - loss: 0.5525 - val_loss: 1.0354
Epoch 118/200
7/7 [==============================] - ETA: 0s - loss: 0.5525
Epoch 00118: loss did not improve from 0.55243

Epoch 00118: ReduceLROnPlateau reducing learning rate to 1.2800001059076749e-08.
7/7 [==============================] - 1s 91ms/step - loss: 0.5525 - val_loss: 1.0354
Epoch 119/200
7/7 [==============================] - ETA: 0s - loss: 0.5525
Epoch 00119: loss did not improve from 0.55243
7/7 [==============================] - 1s 102ms/step - loss: 0.5525 - val_loss: 1.0354
Epoch 120/200
7/7 [==============================] - ETA: 0s - loss: 0.5525
Epoch 00120: loss did not improve from 0.55243
Restoring model weights from the end of the best epoch.
7/7 [==============================] - 1s 103ms/step - loss: 0.5525 - val_loss: 1.0353
Epoch 00120: early stopping
```

In [113…

```
model = load_model('sales.h5')
```

In [114…

```
train_predict = model.predict(x_train)
test_predict = model.predict(x_test)
```

In [115…

```
#train_predict = std.inverse_transform(train_predict)
train_predict = train_predict.reshape(len(train_predict))
#test_predict = std.inverse_transform(test_predict)
test_predict = test_predict.reshape(len(test_predict))
```

In [116…

```
# inversion of normalisation
train_predict = train_predict*std.values + mean.values
test_predict = test_predict*std.values + mean.values
y_train = y_train*std.values + mean.values
y_test = y_test*std.values + mean.values
```

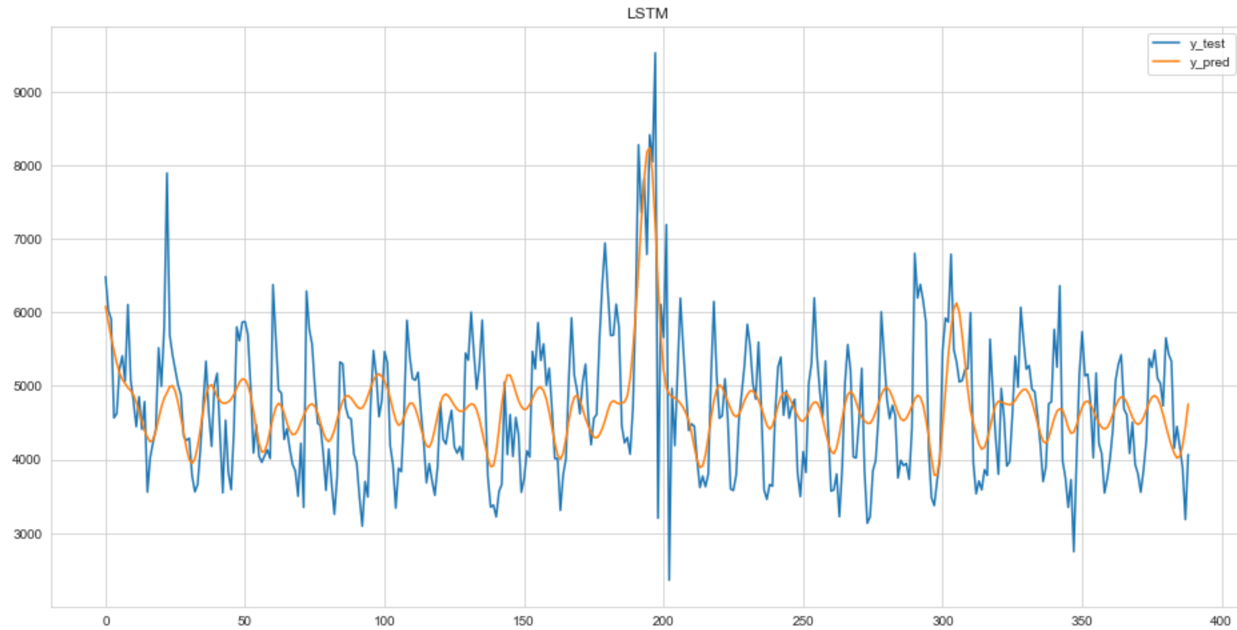
In [117…
```
RMSE_LSTM = math.sqrt(mean_squared_error(y_train,train_predict))
RMSE_LSTM
```

Out[117… 758.7049250457853

In [118…
```
RMSE_LSTM = math.sqrt(mean_squared_error(y_test,test_predict))
RMSE_LSTM
```

Out[118… 1038.4188839819349

In [119…
```
plt.figure(figsize = (16,8))
plt.plot(y_train, label = 'y_test')
plt.plot(train_predict,label = 'y_pred')
plt.title('LSTM')
plt.legend()
plt.show()
```



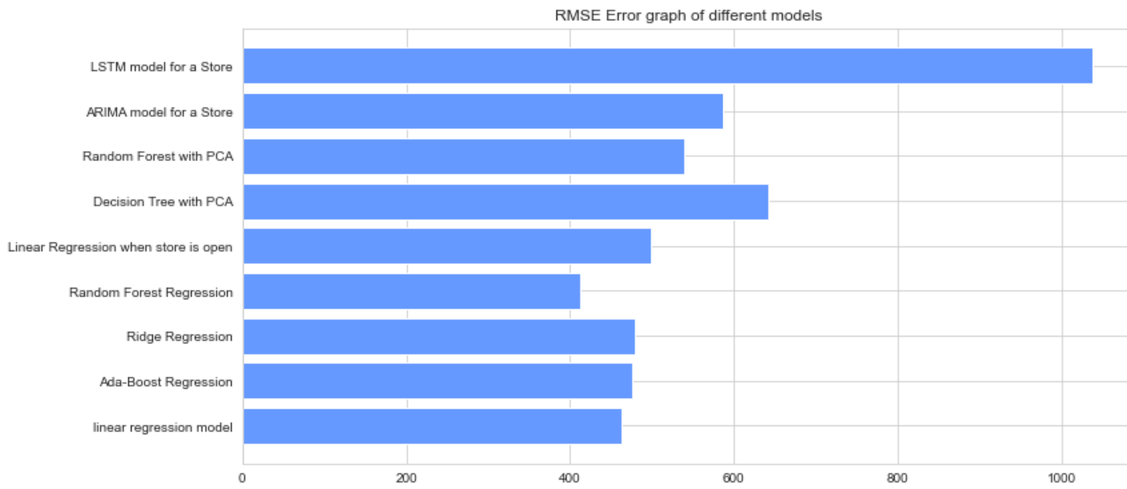
- Here, Tradional Time-Series models performs better than LSTM model.

2. Comment on the behavior of all the models you have built so far

In [121…
```
models_error = [[error_output_lr.RMSE.mean(), 'linear regression model']] # Linear regression model
models_error.append([error_output_ada.RMSE.mean(),'Ada-Boost Regression']) # Ada-Boost Regression
models_error.append([error_output_rdg.RMSE.mean(),'Ridge Regression']) # Ridge Regression
models_error.append([error_output_rdm.RMSE.mean(),'Random Forest Regression']) # Random Forest Regression
models_error.append([error_output_lrc.RMSE.mean(),'Linear Regression when store is open']) # Linear Regression when store is open
models_error.append([error_output_dt_pca.RMSE.mean(),'Decision Tree with PCA'])
models_error.append([error_output_rdm_pca.RMSE.mean(),'Random Forest with PCA'])
models_error.append([RMSE_ARIMA,'ARIMA model for a Store'])
models_error.append([RMSE_LSTM, 'LSTM model for a Store'])
```

In [122…
```
models_error = pd.DataFrame(models_error)
```

In [123…
```
plt.figure(figsize=(12,6))
plt.barh(models_error[1],models_error[0], color = "#6699ff")
plt.title('RMSE Error graph of different models')
plt.show()
```



- From the above graph we can clearly says that Random forest performs best out of all models.

3. Cluster stores using sales and customer visits as features. Find out how many clusters or groups are possible. Also visualize the results.

In [124…

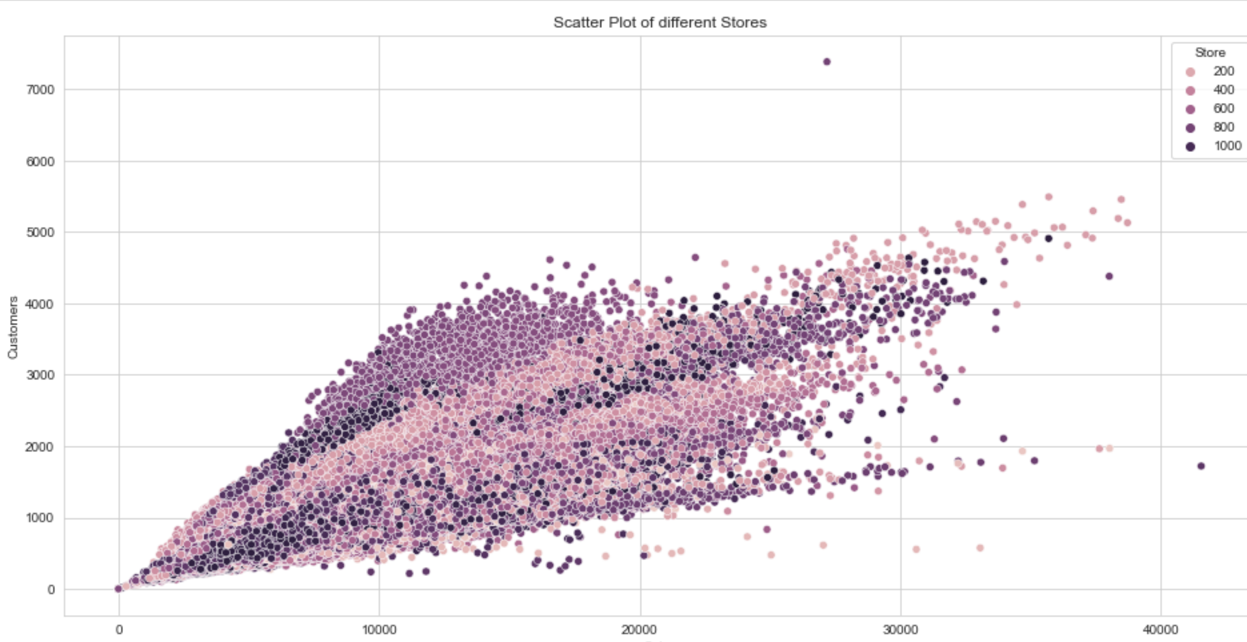
```
cluster_data = train_data[train_data.Open==1]
cluster_data = cluster_data[['Store','Sales','Customers']]
cluster_data.head()
```

Out[124…

	Store	Sales	Customers
0	1	5735	568
1	2	9863	877
2	3	13261	1072
3	4	13106	1488
4	5	6635	645

In [125…

```
plt.figure(figsize=(16,8))
sns.scatterplot(data=cluster_data,x='Sales',y='Customers', hue = 'Store')
plt.title('Scatter Plot of different Stores')
plt.show()
```



In [126…

```
kmeans = KMeans(n_clusters=5, random_state=24).fit(np.array(cluster_data[['Sales','Customers']]))
```

In [127…

```
cluster_data['forecast'] = kmeans.predict(np.array(cluster_data[['Sales','Customers']]))
```

In [128…

```
cluster_data.head()
```

Out[128…

	Store	Sales	Customers	forecast

	Store	Sales	Customers	forecast
0	1	5735	568	4
1	2	9863	877	2
2	3	13261	1072	0
3	4	13106	1488	0
4	5	6635	645	4

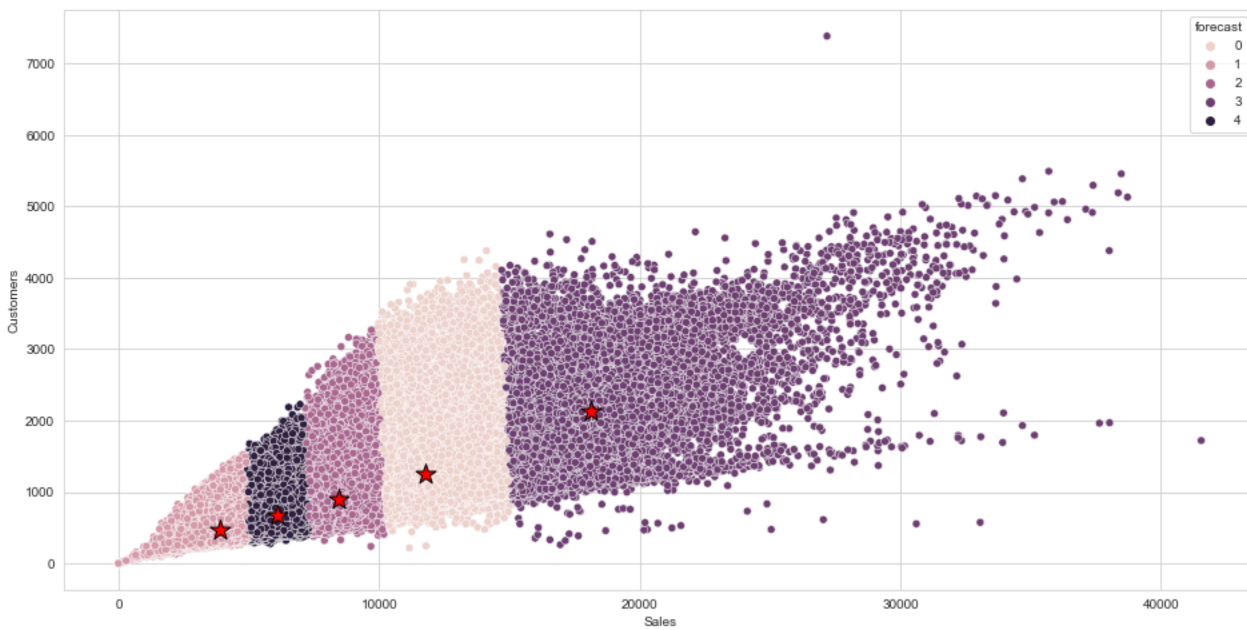
In [129…
```
kmeans.labels_
```

Out[129…
```
array([4, 2, 0, ..., 4, 1, 4])
```

In [130…
```
kmeans.cluster_centers_
```

Out[130…
```
array([[11803.70771128,  1251.32555353],
       [ 3916.63178054,   464.9306038 ],
       [ 8474.27675303,   891.97780749],
       [18132.02765881,  2136.40709015],
       [ 6093.47766668,   672.3953813 ]])
```

In [131…
```
plt.figure(figsize=(16,8))
sns.scatterplot(data=cluster_data,x='Sales',y='Customers', hue = 'forecast')
plt.scatter(
    kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1],
    s=250, marker='*',
    c='red', edgecolor='black',
    label='centroids'
)
plt.show()
```



4. Is it possible to have separate prediction models for each cluster? Compare results with the previous models.

- We will choose Random Forest Regression and prepare a separate prediction model for each cluster.

In [132…
```
cluster_data = train_data[train_data.Open == 1]
cluster_data.drop('Open',axis=1,inplace=True)
```

In [133…
```
kmeans = KMeans(n_clusters=5, random_state=24).fit(np.array(cluster_data[['Sales','Customers']]))
cluster_data['forecast'] = kmeans.predict(np.array(cluster_data[['Sales','Customers']]))
```

In [134…
```
cluster_data.head()
```

Out[134…

	Store	DayOfWeek	Sales	Customers	Promo	StateHoliday	SchoolHoliday	forecast
0	1	2	5735	568	1	0	0	4
1	2	2	9863	877	1	0	0	2
2	3	2	13261	1072	1	0	1	0
3	4	2	13106	1488	1	0	0	0
4	5	2	6635	645	1	0	0	4

In [135…
```
cluster_data.drop('Store',axis=1,inplace=True)
```

In [136…
```
RMSE_cluster_rdm = []
MAE_cluster_rdm=[]
for clust in range(5):
    data = cluster_data[cluster_data.forecast==clust]
    data.drop('forecast',axis=1,inplace=True)
    y=np.array(data['Sales'])
    x=np.array(data.drop('Sales',axis=1))
    y_test,y_pred  = random_forest(np.array(x),np.array(y))
    RMSE_1,MAE_1 = error_cal(y_test,y_pred)
    RMSE_cluster_rdm.append(RMSE_1)
    MAE_cluster_rdm.append(MAE_1)
```

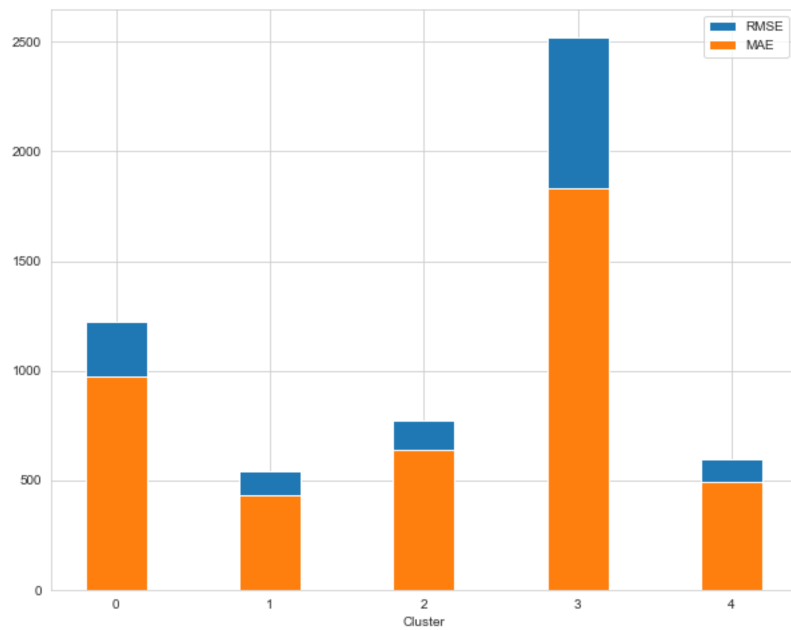
In [137…
```
cluster = [0,1,2,3,4]
```

In [138…
```
error_output_cluster_rdm = pd.DataFrame()
error_output_cluster_rdm['cluster'] = cluster
error_output_cluster_rdm['RMSE'] = RMSE_cluster_rdm
error_output_cluster_rdm['MAE'] = MAE_cluster_rdm
error_output_cluster_rdm
```

Out[138…

	cluster	RMSE	MAE
0	0	1222.541431	971.439099
1	1	543.165394	433.964896
2	2	770.898979	636.430048
3	3	2522.615663	1831.995396
4	4	596.155043	494.302385

In [139…
```
plt.figure(figsize=(10,8))
plt.bar(x=error_output_cluster_rdm.cluster,height=error_output_cluster_rdm.RMSE,label = 'RMSE',width=0.4)
plt.bar(x=error_output_cluster_rdm.cluster,height=error_output_cluster_rdm.MAE,label = 'MAE',width=0.4)
plt.xlabel('Cluster')
plt.legend()
plt.show()
```



- since data is not suitable for clustring, we can not separate data into different clusters.
- so while predicting sales based on clusters, it shows unpredictible result (RMSE, and MAE)

Project Task: Week 4

Applying ANN:

1. Use ANN (Artificial Neural Network) to predict Store Sales.

```
a)    Fine-tune number of layers,

b)    Number of Neurons in each layers.

c)    Experiment in batch-size.
```

```
d)    Experiment with number of epochs. Carefully observe the loss and accuracy? What are the observations?

e)    Play with different  Learning Rate  variants of Gradient Descent like Adam, SGD, RMS-prop.

f)    Which activation performs best for this use case and why?

g)    Check how it performed in the dataset, calculate RMSE.
```

- I have tested so many combinations of hyper-parameters and finally i found best result as follow: -

In [140…

```
train_data.head()
```

Out[140…

	Store	DayOfWeek	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
0	1	2	5735	568	1	1	0	0
1	2	2	9863	877	1	1	0	0
2	3	2	13261	1072	1	1	0	1
3	4	2	13106	1488	1	1	0	0
4	5	2	6635	645	1	1	0	0

In [141…

```
train_data = train_data[train_data.Store<=100]
train_data = train_data[train_data.Open == 1]
train_data.reset_index(drop=True, inplace=True)

y = train_data['Sales']
x = train_data.drop(['Sales','Open'],axis=1)

std = StandardScaler()
x = std.fit_transform(x)
```

In [142…

```
x_train,x_test,y_train,y_test = train_test_split(np.array(x),np.array(y),random_state=42,test_size=0.3)
```

In [143…

```
model_1 = Sequential()
model_1.add(layers.Dense(32, activation='elu', input_shape = (x_train.shape[1],)))
model_1.add(layers.Dense(64, activation='elu'))
model_1.add(layers.Dense(64, activation='elu'))
model_1.add(layers.BatchNormalization())

## block 2
model_1.add(layers.Dense(128, activation='elu'))
model_1.add(layers.Dense(128, activation='elu'))
model_1.add(layers.BatchNormalization())

## block 3
model_1.add(layers.Dense(256, activation='elu'))
model_1.add(layers.Dense(256, activation='elu'))
model_1.add(layers.BatchNormalization())

## block 4
model_1.add(layers.Dense(128, activation='elu'))
model_1.add(layers.Dense(128, activation='elu'))
model_1.add(layers.BatchNormalization())

## block 5
model_1.add(layers.Dense(64, activation='elu'))
model_1.add(layers.Dense(64, activation='elu'))
model_1.add(layers.Dense(32, activation='elu'))
model_1.add(layers.Dense(1))
```

In [144…

```
model_1.compile(loss='mse',
              optimizer = Adam(learning_rate=0.001),
              metrics=['mae'])
```

In [145…

```
checkpoint = ModelCheckpoint('Sales_ann.h5',
                            monitor='loss',
                            mode=min,
                            save_best_only=True,
                            verbose=1)

early_stopping = EarlyStopping(monitor='loss',
                              patience=9,
                              min_delta=0,
                              restore_best_weights=True,
                              verbose=1)

Reduce_ler_rate = ReduceLROnPlateau(monitor='loss',
                                   factor=0.2,
                                   patience=3,
                                   verbose=1,
                                   min_delta=0.001)

callback = [checkpoint,early_stopping,Reduce_ler_rate]
```

```
WARNING:tensorflow:ModelCheckpoint mode <built-in function min> is unknown, fallback to auto mode.
```

In [146…

```python
history = model_1.fit(x_train,y_train,epochs=100,batch_size=20,verbose=1,callbacks=callback)
```

```
Epoch 1/100
2529/2552 [============================>.] - ETA: 0s - loss: 4461951.5000 - mae: 1463.3401
Epoch 00001: loss improved from inf to 4444669.00000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 4444669.0000 - mae: 1461.0424
Epoch 2/100
2530/2552 [============================>.] - ETA: 0s - loss: 2292817.0000 - mae: 1133.0732
Epoch 00002: loss improved from 4444669.00000 to 2293801.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 2293801.5000 - mae: 1133.1906
Epoch 3/100
2549/2552 [============================>.] - ETA: 0s - loss: 2178314.7500 - mae: 1096.1555
Epoch 00003: loss improved from 2293801.50000 to 2177962.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 2177962.5000 - mae: 1096.0912
Epoch 4/100
2544/2552 [============================>.] - ETA: 0s - loss: 2099596.5000 - mae: 1078.0367
Epoch 00004: loss improved from 2177962.50000 to 2098525.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 2098525.7500 - mae: 1077.8282
Epoch 5/100
2546/2552 [============================>.] - ETA: 0s - loss: 1872751.7500 - mae: 1027.8527
Epoch 00005: loss improved from 2098525.75000 to 1872328.00000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1872328.0000 - mae: 1027.9172
Epoch 6/100
2534/2552 [============================>.] - ETA: 0s - loss: 1762951.8750 - mae: 1001.2029
Epoch 00006: loss improved from 1872328.00000 to 1763881.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1763881.8750 - mae: 1001.3951
Epoch 7/100
2542/2552 [============================>.] - ETA: 0s - loss: 1748531.5000 - mae: 995.8099
Epoch 00007: loss improved from 1763881.87500 to 1748732.12500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1748732.1250 - mae: 995.6110
Epoch 8/100
2548/2552 [============================>.] - ETA: 0s - loss: 1703742.6250 - mae: 983.8541
Epoch 00008: loss improved from 1748732.12500 to 1704187.37500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1704187.3750 - mae: 983.9661
Epoch 9/100
2549/2552 [============================>.] - ETA: 0s - loss: 1676383.2500 - mae: 974.7777
Epoch 00009: loss improved from 1704187.37500 to 1675809.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1675809.5000 - mae: 974.6068
Epoch 10/100
2532/2552 [============================>.] - ETA: 0s - loss: 1667442.6250 - mae: 972.2385
Epoch 00010: loss improved from 1675809.50000 to 1668408.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1668408.6250 - mae: 972.5330
Epoch 11/100
2533/2552 [============================>.] - ETA: 0s - loss: 1647250.5000 - mae: 965.6171
Epoch 00011: loss improved from 1668408.62500 to 1644639.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1644639.6250 - mae: 965.1610
Epoch 12/100
2526/2552 [============================>.] - ETA: 0s - loss: 1615871.8750 - mae: 953.6132
Epoch 00012: loss improved from 1644639.62500 to 1613226.25000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1613226.2500 - mae: 953.1456
Epoch 13/100
2535/2552 [============================>.] - ETA: 0s - loss: 1593090.6250 - mae: 947.5179
Epoch 00013: loss improved from 1613226.25000 to 1592178.00000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1592178.0000 - mae: 947.2323
Epoch 14/100
2550/2552 [============================>.] - ETA: 0s - loss: 1577112.8750 - mae: 941.9357
Epoch 00014: loss improved from 1592178.00000 to 1577025.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1577025.8750 - mae: 941.8602
Epoch 15/100
2541/2552 [============================>.] - ETA: 0s - loss: 1566032.3750 - mae: 937.7127
Epoch 00015: loss improved from 1577025.87500 to 1566442.25000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1566442.2500 - mae: 937.7921
Epoch 16/100
2543/2552 [============================>.] - ETA: 0s - loss: 1546594.6250 - mae: 931.5505
Epoch 00016: loss improved from 1566442.25000 to 1548221.00000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1548221.0000 - mae: 932.1654
Epoch 17/100
2531/2552 [============================>.] - ETA: 0s - loss: 1517649.8750 - mae: 924.1529
Epoch 00017: loss improved from 1548221.00000 to 1516797.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1516797.6250 - mae: 923.8798
Epoch 18/100
2549/2552 [============================>.] - ETA: 0s - loss: 1499224.3750 - mae: 918.7906
Epoch 00018: loss improved from 1516797.62500 to 1499382.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1499382.6250 - mae: 918.8552
Epoch 19/100
2539/2552 [============================>.] - ETA: 0s - loss: 1472675.7500 - mae: 912.6802
Epoch 00019: loss improved from 1499382.62500 to 1473326.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1473326.6250 - mae: 912.9028
Epoch 20/100
2535/2552 [============================>.] - ETA: 0s - loss: 1468572.2500 - mae: 909.8781
Epoch 00020: loss improved from 1473326.62500 to 1469643.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1469643.6250 - mae: 909.9742
Epoch 21/100
2536/2552 [============================>.] - ETA: 0s - loss: 1436184.8750 - mae: 897.9775
Epoch 00021: loss improved from 1469643.62500 to 1437130.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1437130.7500 - mae: 898.1415
Epoch 22/100
2532/2552 [============================>.] - ETA: 0s - loss: 1420559.6250 - mae: 891.1735
Epoch 00022: loss improved from 1437130.75000 to 1420546.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1420546.8750 - mae: 891.1373
Epoch 23/100
2532/2552 [============================>.] - ETA: 0s - loss: 1410007.3750 - mae: 888.0452
Epoch 00023: loss improved from 1420546.87500 to 1408641.37500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1408641.3750 - mae: 887.6962
Epoch 24/100
2552/2552 [==============================] - ETA: 0s - loss: 1393509.5000 - mae: 883.4465
```

```
Epoch 00024: loss improved from 1408641.37500 to 1393509.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1393509.5000 - mae: 883.4465
Epoch 25/100
2535/2552 [============================>.] - ETA: 0s - loss: 1385574.6250 - mae: 882.0277
Epoch 00025: loss improved from 1393509.50000 to 1386443.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1386443.7500 - mae: 881.9350
Epoch 26/100
2546/2552 [============================>.] - ETA: 0s - loss: 1364715.6250 - mae: 872.7318
Epoch 00026: loss improved from 1386443.75000 to 1364199.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1364199.7500 - mae: 872.6283
Epoch 27/100
2531/2552 [============================>.] - ETA: 0s - loss: 1349756.7500 - mae: 868.5546
Epoch 00027: loss improved from 1364199.75000 to 1348304.12500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1348304.1250 - mae: 868.3804
Epoch 28/100
2532/2552 [============================>.] - ETA: 0s - loss: 1338930.1250 - mae: 863.4944
Epoch 00028: loss improved from 1348304.12500 to 1339281.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1339281.5000 - mae: 863.7580
Epoch 29/100
2535/2552 [============================>.] - ETA: 0s - loss: 1324684.0000 - mae: 858.8839
Epoch 00029: loss improved from 1339281.50000 to 1324433.25000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1324433.2500 - mae: 858.8866
Epoch 30/100
2552/2552 [==============================] - ETA: 0s - loss: 1308992.2500 - mae: 852.7219
Epoch 00030: loss improved from 1324433.25000 to 1308992.25000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1308992.2500 - mae: 852.7219
Epoch 31/100
2552/2552 [==============================] - ETA: 0s - loss: 1295025.1250 - mae: 849.9790
Epoch 00031: loss improved from 1308992.25000 to 1295025.12500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1295025.1250 - mae: 849.9790
Epoch 32/100
2544/2552 [============================>.] - ETA: 0s - loss: 1275258.2500 - mae: 840.5229
Epoch 00032: loss improved from 1295025.12500 to 1275054.25000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1275054.2500 - mae: 840.5606
Epoch 33/100
2543/2552 [============================>.] - ETA: 0s - loss: 1238651.6250 - mae: 830.3442
Epoch 00033: loss improved from 1275054.25000 to 1239158.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1239158.8750 - mae: 830.3959
Epoch 34/100
2539/2552 [============================>.] - ETA: 0s - loss: 1235691.1250 - mae: 827.0690
Epoch 00034: loss improved from 1239158.87500 to 1235172.12500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1235172.1250 - mae: 826.8186
Epoch 35/100
2528/2552 [============================>.] - ETA: 0s - loss: 1209391.1250 - mae: 817.1584
Epoch 00035: loss improved from 1235172.12500 to 1208047.37500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1208047.3750 - mae: 816.8604
Epoch 36/100
2533/2552 [============================>.] - ETA: 0s - loss: 1186389.3750 - mae: 808.1669
Epoch 00036: loss improved from 1208047.37500 to 1187663.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1187663.5000 - mae: 808.5308
Epoch 37/100
2552/2552 [==============================] - ETA: 0s - loss: 1169292.5000 - mae: 801.5657
Epoch 00037: loss improved from 1187663.50000 to 1169292.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1169292.5000 - mae: 801.5657
Epoch 38/100
2537/2552 [============================>.] - ETA: 0s - loss: 1155176.7500 - mae: 794.0585
Epoch 00038: loss improved from 1169292.50000 to 1157919.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 1157919.6250 - mae: 794.5342
Epoch 39/100
2535/2552 [============================>.] - ETA: 0s - loss: 1136514.0000 - mae: 788.1987
Epoch 00039: loss improved from 1157919.62500 to 1138389.37500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1138389.3750 - mae: 788.6247
Epoch 40/100
2540/2552 [============================>.] - ETA: 0s - loss: 1124602.6250 - mae: 783.5734
Epoch 00040: loss improved from 1138389.37500 to 1125951.12500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1125951.1250 - mae: 783.7244
Epoch 41/100
2542/2552 [============================>.] - ETA: 0s - loss: 1114318.8750 - mae: 775.1037
Epoch 00041: loss improved from 1125951.12500 to 1113751.00000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1113751.0000 - mae: 775.0018
Epoch 42/100
2546/2552 [============================>.] - ETA: 0s - loss: 1101419.1250 - mae: 770.6692
Epoch 00042: loss improved from 1113751.00000 to 1101597.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1101597.7500 - mae: 770.8145
Epoch 43/100
2539/2552 [============================>.] - ETA: 0s - loss: 1085153.0000 - mae: 763.4331
Epoch 00043: loss improved from 1101597.75000 to 1085373.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1085373.8750 - mae: 763.5228
Epoch 44/100
2536/2552 [============================>.] - ETA: 0s - loss: 1074871.1250 - mae: 757.6851
Epoch 00044: loss improved from 1085373.87500 to 1075420.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1075420.8750 - mae: 757.5859
Epoch 45/100
2531/2552 [============================>.] - ETA: 0s - loss: 1062451.0000 - mae: 755.1668
Epoch 00045: loss improved from 1075420.87500 to 1061920.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1061920.8750 - mae: 755.0569
Epoch 46/100
2537/2552 [============================>.] - ETA: 0s - loss: 1057195.8750 - mae: 749.1490
Epoch 00046: loss improved from 1061920.87500 to 1057397.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1057397.7500 - mae: 749.4203
Epoch 47/100
2527/2552 [============================>.] - ETA: 0s - loss: 1044205.0000 - mae: 745.9543
Epoch 00047: loss improved from 1057397.75000 to 1045839.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1045839.7500 - mae: 745.8461
Epoch 48/100
2546/2552 [============================>.] - ETA: 0s - loss: 1026479.1875 - mae: 738.7417
Epoch 00048: loss improved from 1045839.75000 to 1026169.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1026169.6250 - mae: 738.6118
```

```
Epoch 49/100
2530/2552 [============================>.] - ETA: 0s - loss: 1021247.2500 - mae: 736.6475
Epoch 00049: loss improved from 1026169.62500 to 1020720.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1020720.6250 - mae: 736.5558
Epoch 50/100
2529/2552 [============================>.] - ETA: 0s - loss: 1010111.0000 - mae: 730.5411
Epoch 00050: loss improved from 1020720.62500 to 1009079.62500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 1009079.6250 - mae: 730.3191
Epoch 51/100
2529/2552 [============================>.] - ETA: 0s - loss: 1000115.3125 - mae: 725.9989
Epoch 00051: loss improved from 1009079.62500 to 1000289.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 1000289.7500 - mae: 725.8548
Epoch 52/100
2548/2552 [============================>.] - ETA: 0s - loss: 993450.0000 - mae: 724.3928
Epoch 00052: loss improved from 1000289.75000 to 993707.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 993707.7500 - mae: 724.4769
Epoch 53/100
2549/2552 [============================>.] - ETA: 0s - loss: 975667.7500 - mae: 717.6727
Epoch 00053: loss improved from 993707.75000 to 975573.18750, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 975573.1875 - mae: 717.7094
Epoch 54/100
2533/2552 [============================>.] - ETA: 0s - loss: 977863.0625 - mae: 716.8930
Epoch 00054: loss did not improve from 975573.18750
2552/2552 [==============================] - 5s 2ms/step - loss: 976610.4375 - mae: 716.5394
Epoch 55/100
2530/2552 [============================>.] - ETA: 0s - loss: 957266.3750 - mae: 709.7372
Epoch 00055: loss improved from 975573.18750 to 957227.18750, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 957227.1875 - mae: 709.9012
Epoch 56/100
2552/2552 [==============================] - ETA: 0s - loss: 953254.9375 - mae: 710.8003
Epoch 00056: loss improved from 957227.18750 to 953254.93750, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 953254.9375 - mae: 710.8003
Epoch 57/100
2534/2552 [============================>.] - ETA: 0s - loss: 944713.8750 - mae: 705.0320
Epoch 00057: loss improved from 953254.93750 to 944173.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 944173.5000 - mae: 705.0037
Epoch 58/100
2541/2552 [============================>.] - ETA: 0s - loss: 936409.5000 - mae: 702.8905
Epoch 00058: loss improved from 944173.50000 to 936218.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 936218.7500 - mae: 702.7877
Epoch 59/100
2552/2552 [==============================] - ETA: 0s - loss: 912748.2500 - mae: 694.3871
Epoch 00059: loss improved from 936218.75000 to 912748.25000, saving model to Sales_ann.h5
2552/2552 [==============================] - 8s 3ms/step - loss: 912748.2500 - mae: 694.3871
Epoch 60/100
2542/2552 [============================>.] - ETA: 0s - loss: 902439.6875 - mae: 691.5548
Epoch 00060: loss improved from 912748.25000 to 902033.68750, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 902033.6875 - mae: 691.4305
Epoch 61/100
2538/2552 [============================>.] - ETA: 0s - loss: 883714.5000 - mae: 682.2023
Epoch 00061: loss improved from 902033.68750 to 882624.31250, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 882624.3125 - mae: 681.8970
Epoch 62/100
2537/2552 [============================>.] - ETA: 0s - loss: 886195.6875 - mae: 685.2569
Epoch 00062: loss did not improve from 882624.31250
2552/2552 [==============================] - 6s 2ms/step - loss: 885741.1250 - mae: 685.1020
Epoch 63/100
2542/2552 [============================>.] - ETA: 0s - loss: 870655.3125 - mae: 677.2553
Epoch 00063: loss improved from 882624.31250 to 871315.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 871315.8750 - mae: 677.2231
Epoch 64/100
2551/2552 [============================>.] - ETA: 0s - loss: 863284.7500 - mae: 675.7388
Epoch 00064: loss improved from 871315.87500 to 863500.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 863500.8750 - mae: 675.8042
Epoch 65/100
2552/2552 [==============================] - ETA: 0s - loss: 841904.9375 - mae: 666.6998
Epoch 00065: loss improved from 863500.87500 to 841904.93750, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 841904.9375 - mae: 666.6998
Epoch 66/100
2532/2552 [============================>.] - ETA: 0s - loss: 833314.0000 - mae: 662.7435
Epoch 00066: loss improved from 841904.93750 to 836731.68750, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 836731.6875 - mae: 663.0487
Epoch 67/100
2531/2552 [============================>.] - ETA: 0s - loss: 820179.3125 - mae: 655.5438
Epoch 00067: loss improved from 836731.68750 to 819511.31250, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 819511.3125 - mae: 655.5641
Epoch 68/100
2546/2552 [============================>.] - ETA: 0s - loss: 817279.1875 - mae: 656.0784
Epoch 00068: loss improved from 819511.31250 to 817239.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 817239.7500 - mae: 656.1539
Epoch 69/100
2529/2552 [============================>.] - ETA: 0s - loss: 812025.0625 - mae: 651.7642
Epoch 00069: loss improved from 817239.75000 to 811222.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 811222.5000 - mae: 651.6959
Epoch 70/100
2533/2552 [============================>.] - ETA: 0s - loss: 800828.0625 - mae: 646.4879
Epoch 00070: loss improved from 811222.50000 to 801707.25000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 801707.2500 - mae: 646.6812
Epoch 71/100
2548/2552 [============================>.] - ETA: 0s - loss: 788045.0625 - mae: 641.6236
Epoch 00071: loss improved from 801707.25000 to 788003.18750, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 788003.1875 - mae: 641.6391
Epoch 72/100
2550/2552 [============================>.] - ETA: 0s - loss: 783170.6250 - mae: 638.2745
Epoch 00072: loss improved from 788003.18750 to 782958.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 782958.7500 - mae: 638.2116
Epoch 73/100
2545/2552 [============================>.] - ETA: 0s - loss: 774015.1875 - mae: 634.6124
```

```
Epoch 00073: loss improved from 782958.75000 to 774113.56250, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 774113.5625 - mae: 634.7856
Epoch 74/100
2536/2552 [============================>.] - ETA: 0s - loss: 779740.2500 - mae: 636.6808
Epoch 00074: loss did not improve from 774113.56250
2552/2552 [==============================] - 5s 2ms/step - loss: 778898.2500 - mae: 636.2727
Epoch 75/100
2550/2552 [============================>.] - ETA: 0s - loss: 760765.6875 - mae: 628.2018
Epoch 00075: loss improved from 774113.56250 to 760768.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 5s 2ms/step - loss: 760768.8750 - mae: 628.2316
Epoch 76/100
2534/2552 [============================>.] - ETA: 0s - loss: 754688.0000 - mae: 626.0912
Epoch 00076: loss improved from 760768.87500 to 755313.37500, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 3ms/step - loss: 755313.3750 - mae: 625.9629
Epoch 77/100
2543/2552 [============================>.] - ETA: 0s - loss: 758326.1875 - mae: 625.4481
Epoch 00077: loss did not improve from 755313.37500
2552/2552 [==============================] - 6s 2ms/step - loss: 758324.8750 - mae: 625.4294
Epoch 78/100
2548/2552 [============================>.] - ETA: 0s - loss: 741719.0625 - mae: 619.1223
Epoch 00078: loss improved from 755313.37500 to 742320.50000, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 742320.5000 - mae: 619.2246
Epoch 79/100
2530/2552 [============================>.] - ETA: 0s - loss: 752748.1250 - mae: 624.0490
Epoch 00079: loss did not improve from 742320.50000
2552/2552 [==============================] - 7s 3ms/step - loss: 752347.1250 - mae: 623.8472
Epoch 80/100
2546/2552 [============================>.] - ETA: 0s - loss: 735614.5625 - mae: 616.8571
Epoch 00080: loss improved from 742320.50000 to 735595.75000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 735595.7500 - mae: 616.8911
Epoch 81/100
2529/2552 [============================>.] - ETA: 0s - loss: 729291.0000 - mae: 613.6129
Epoch 00081: loss improved from 735595.75000 to 729175.00000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 729175.0000 - mae: 613.5229
Epoch 82/100
2536/2552 [============================>.] - ETA: 0s - loss: 738066.8750 - mae: 615.2469
Epoch 00082: loss did not improve from 729175.00000
2552/2552 [==============================] - 6s 2ms/step - loss: 737757.1250 - mae: 614.9995
Epoch 83/100
2530/2552 [============================>.] - ETA: 0s - loss: 719013.1250 - mae: 610.8231
Epoch 00083: loss improved from 729175.00000 to 722517.56250, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 722517.5625 - mae: 611.2591
Epoch 84/100
2538/2552 [============================>.] - ETA: 0s - loss: 717942.6250 - mae: 606.2766
Epoch 00084: loss improved from 722517.56250 to 717757.81250, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 717757.8125 - mae: 606.2916
Epoch 85/100
2544/2552 [============================>.] - ETA: 0s - loss: 725121.7500 - mae: 610.8640
Epoch 00085: loss did not improve from 717757.81250
2552/2552 [==============================] - 6s 2ms/step - loss: 724865.2500 - mae: 610.8557
Epoch 86/100
2543/2552 [============================>.] - ETA: 0s - loss: 705668.2500 - mae: 601.7151
Epoch 00086: loss improved from 717757.81250 to 705479.68750, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 705479.6875 - mae: 601.6314
Epoch 87/100
2545/2552 [============================>.] - ETA: 0s - loss: 716355.0000 - mae: 606.7365
Epoch 00087: loss did not improve from 705479.68750
2552/2552 [==============================] - 6s 2ms/step - loss: 716592.4375 - mae: 606.8080
Epoch 88/100
2530/2552 [============================>.] - ETA: 0s - loss: 704095.8125 - mae: 600.3581
Epoch 00088: loss improved from 705479.68750 to 703681.18750, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 703681.1875 - mae: 600.1520
Epoch 89/100
2544/2552 [============================>.] - ETA: 0s - loss: 704575.5000 - mae: 599.5171
Epoch 00089: loss did not improve from 703681.18750
2552/2552 [==============================] - 6s 2ms/step - loss: 704088.6875 - mae: 599.2795
Epoch 90/100
2538/2552 [============================>.] - ETA: 0s - loss: 705700.7500 - mae: 598.5233
Epoch 00090: loss did not improve from 703681.18750
2552/2552 [==============================] - 6s 2ms/step - loss: 707221.2500 - mae: 598.7364
Epoch 91/100
2545/2552 [============================>.] - ETA: 0s - loss: 699977.2500 - mae: 600.1812
Epoch 00091: loss improved from 703681.18750 to 700657.00000, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 700657.0000 - mae: 600.4050
Epoch 92/100
2550/2552 [============================>.] - ETA: 0s - loss: 700018.3750 - mae: 595.4828
Epoch 00092: loss improved from 700657.00000 to 699920.93750, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 699920.9375 - mae: 595.4360
Epoch 93/100
2537/2552 [============================>.] - ETA: 0s - loss: 685793.9375 - mae: 590.9415
Epoch 00093: loss improved from 699920.93750 to 684823.18750, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 684823.1875 - mae: 590.7275
Epoch 94/100
2537/2552 [============================>.] - ETA: 0s - loss: 692054.9375 - mae: 594.4563
Epoch 00094: loss did not improve from 684823.18750
2552/2552 [==============================] - 7s 3ms/step - loss: 691788.4375 - mae: 594.2739
Epoch 95/100
2552/2552 [==============================] - ETA: 0s - loss: 679983.1250 - mae: 588.7256
Epoch 00095: loss improved from 684823.18750 to 679983.12500, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 679983.1250 - mae: 588.7256
Epoch 96/100
2549/2552 [============================>.] - ETA: 0s - loss: 678964.3750 - mae: 586.9264
Epoch 00096: loss improved from 679983.12500 to 679199.06250, saving model to Sales_ann.h5
2552/2552 [==============================] - 6s 2ms/step - loss: 679199.0625 - mae: 586.9881
Epoch 97/100
2552/2552 [==============================] - ETA: 0s - loss: 680674.8750 - mae: 586.9417
Epoch 00097: loss did not improve from 679199.06250
2552/2552 [==============================] - 7s 3ms/step - loss: 680674.8750 - mae: 586.9417
```

```
Epoch 98/100
2552/2552 [==============================] - ETA: 0s - loss: 670993.5625 - mae: 584.3502
Epoch 00098: loss improved from 679199.06250 to 670993.56250, saving model to Sales_ann.h5
2552/2552 [==============================] - 8s 3ms/step - loss: 670993.5625 - mae: 584.3502
Epoch 99/100
2533/2552 [============================>.] - ETA: 0s - loss: 672261.1875 - mae: 586.1032
Epoch 00099: loss did not improve from 670993.56250
2552/2552 [==============================] - 6s 3ms/step - loss: 671468.6875 - mae: 585.9103
Epoch 100/100
2538/2552 [============================>.] - ETA: 0s - loss: 663033.5625 - mae: 579.5814
Epoch 00100: loss improved from 670993.56250 to 663814.87500, saving model to Sales_ann.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 663814.8750 - mae: 579.7318
```

In [147…

```
model_1 = load_model('Sales_ann.h5')

y_pred = model_1.predict(x_test)
```

In [148…

```
RMSE = math.sqrt(mean_squared_error(y_test,y_pred))
RMSE
```

Out[148… 788.7476204485714

In [149…

```
#importing testing data
test_data = pd.read_csv(r'D:\Simplilearn\project\Artificial-Intelligence-Capstone-Project-Datasets-master\Project 3-Retail-Datasets_data\tes
test_data.head()
```

Out[149…

	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 [150…

```
test_data.drop('Date',axis = 1, inplace=True)
test_data.loc[test_data.StateHoliday==0,'StateHoliday'] = '0'
labelencoder= LabelEncoder()
test_data.StateHoliday = labelencoder.fit_transform(test_data['StateHoliday'])
test_data = test_data[test_data.Store<=100]
test_data = test_data[test_data.Open == 1]
test_data.reset_index(drop=True, inplace=True)

y = test_data['Sales']
x = test_data.drop(['Sales','Open'],axis=1)

std = StandardScaler()
x = std.fit_transform(x)
```

In [151…

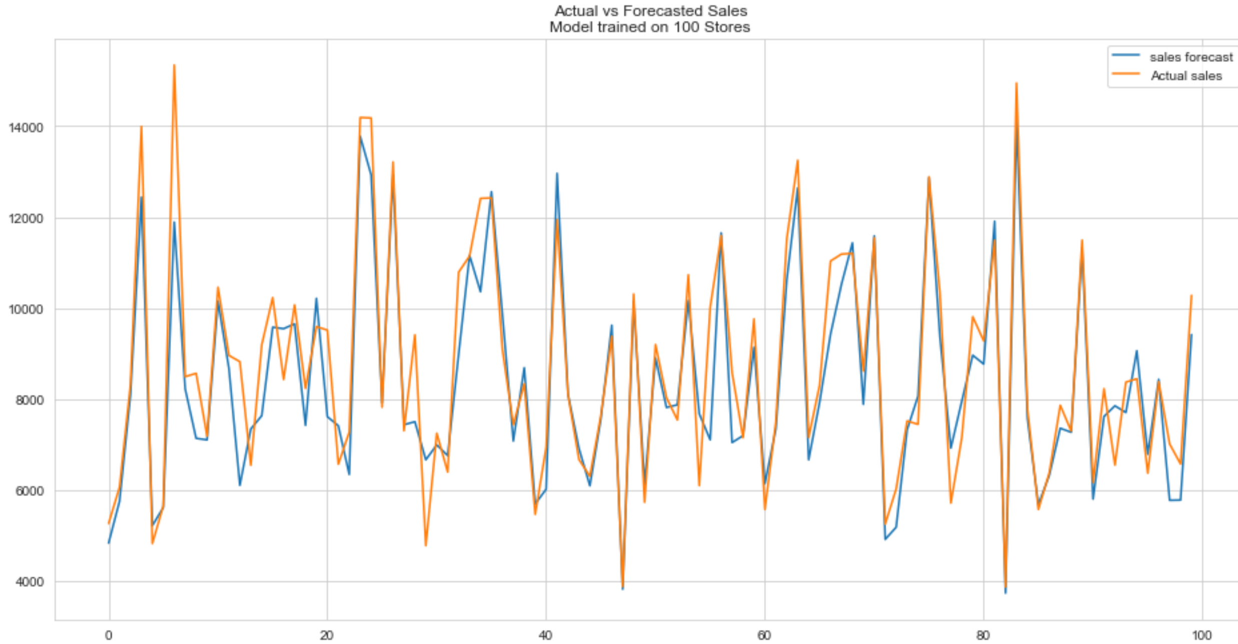
```
y_pred = model_1.predict(x)
math.sqrt(mean_squared_error(y,y_pred))
```

Out[151… 793.2148083439479

In [152…

```
plt.figure(figsize=(16,8))
plt.plot(y_pred[:100],label = 'sales forecast')
plt.plot(y[:100],label = 'Actual sales')
plt.legend()
plt.title('Actual vs Forecasted Sales\nModel trained on 100 Stores')
```

Out[152… Text(0.5, 1.0, 'Actual vs Forecasted Sales\nModel trained on 100 Stores')



2. Use Dropout for ANN and find the optimum number of clusters (clusters formed considering the features: sales and customer visits). Compare model performance with traditional ML based prediction models.

In [153…

```
train_data = train_data[train_data.Store<=100]
train_data = train_data[train_data.Open == 1]
train_data.reset_index(drop=True, inplace=True)

y = train_data['Sales']
x = train_data.drop(['Sales','Open'],axis=1)

std = StandardScaler()
x = std.fit_transform(x)

x_train,x_test,y_train,y_test = train_test_split(np.array(x),np.array(y),random_state=42,test_size=0.3)
```

In [154…

```
model_2 = Sequential()
model_2.add(layers.Dense(32, activation='elu', input_shape = (x_train.shape[1],)))
model_2.add(layers.Dense(64, activation='elu'))
model_2.add(layers.Dense(64, activation='elu'))
model_2.add(layers.BatchNormalization())

## block 2
model_2.add(layers.Dense(128, activation='elu'))
model_2.add(layers.Dense(128, activation='elu'))
model_2.add(layers.BatchNormalization())

## block 3
model_2.add(layers.Dense(256, activation='elu'))
model_2.add(layers.Dense(256, activation='elu'))
model_2.add(layers.BatchNormalization())
model_2.add(layers.Dropout(0.8))

## block 4
model_2.add(layers.Dense(128, activation='elu'))
model_2.add(layers.Dense(128, activation='elu'))
model_2.add(layers.BatchNormalization())
model_2.add(layers.Dropout(0.8))

## block 5
model_2.add(layers.Dense(64, activation='elu'))
model_2.add(layers.Dense(64, activation='elu'))
model_2.add(layers.Dense(32, activation='elu'))
model_2.add(layers.Dropout(0.8))
model_2.add(layers.Dense(1))
```

In [155…

```
model_2.compile(loss='mse',
              optimizer = Adam(learning_rate=0.001),
              metrics=['mae'])
```

In [156…

```
checkpoint = ModelCheckpoint('Sales_ann_with_dropout.h5',
                            monitor='loss',
                            mode=min,
                            save_best_only=True,
                            verbose=1)

early_stopping = EarlyStopping(monitor='loss',
                              patience=9,
                              min_delta=0,
```

```
                                restore_best_weights=True,
                                verbose=1)

Reduce_ler_rate = ReduceLROnPlateau(monitor='loss',
                                    factor=0.2,
                                    patience=3,
                                    verbose=1,
                                    min_delta=0.001)

callback = [checkpoint,early_stopping,Reduce_ler_rate]
```

```
WARNING:tensorflow:ModelCheckpoint mode <built-in function min> is unknown, fallback to auto mode.
```

In [157…

```
history = model_2.fit(x_train,y_train,epochs=50,batch_size=20,verbose=1,callbacks=callback)
```

```
Epoch 1/50
2544/2552 [============================>.] - ETA: 0s - loss: 16398228.0000 - mae: 3228.0522
Epoch 00001: loss improved from inf to 16381824.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 16381824.0000 - mae: 3226.1885
Epoch 2/50
2534/2552 [============================>.] - ETA: 0s - loss: 12938798.0000 - mae: 2782.3682
Epoch 00002: loss improved from 16381824.00000 to 12924422.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 12924422.0000 - mae: 2780.5972
Epoch 3/50
2543/2552 [============================>.] - ETA: 0s - loss: 12470462.0000 - mae: 2722.1392
Epoch 00003: loss improved from 12924422.00000 to 12468093.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 12468093.0000 - mae: 2721.7695
Epoch 4/50
2544/2552 [============================>.] - ETA: 0s - loss: 12054509.0000 - mae: 2671.0891
Epoch 00004: loss improved from 12468093.00000 to 12053403.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 12053403.0000 - mae: 2671.1885
Epoch 5/50
2533/2552 [============================>.] - ETA: 0s - loss: 11776718.0000 - mae: 2638.0352
Epoch 00005: loss improved from 12053403.00000 to 11769261.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 11769261.0000 - mae: 2637.1785
Epoch 6/50
2534/2552 [============================>.] - ETA: 0s - loss: 11808806.0000 - mae: 2637.6265
Epoch 00006: loss did not improve from 11769261.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 11796014.0000 - mae: 2635.9973
Epoch 7/50
2543/2552 [============================>.] - ETA: 0s - loss: 11773491.0000 - mae: 2630.4912
Epoch 00007: loss did not improve from 11769261.00000
2552/2552 [==============================] - 6s 2ms/step - loss: 11780501.0000 - mae: 2631.5269
Epoch 8/50
2548/2552 [============================>.] - ETA: 0s - loss: 11410224.0000 - mae: 2593.7710
Epoch 00008: loss improved from 11769261.00000 to 11413413.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 11413413.0000 - mae: 2594.0405
Epoch 9/50
2542/2552 [============================>.] - ETA: 0s - loss: 11477007.0000 - mae: 2602.1445
Epoch 00009: loss did not improve from 11413413.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 11482330.0000 - mae: 2602.6887
Epoch 10/50
2533/2552 [============================>.] - ETA: 0s - loss: 11473589.0000 - mae: 2595.6792
Epoch 00010: loss did not improve from 11413413.00000
2552/2552 [==============================] - 6s 2ms/step - loss: 11467055.0000 - mae: 2595.1279
Epoch 11/50
2537/2552 [============================>.] - ETA: 0s - loss: 11310850.0000 - mae: 2573.1970
Epoch 00011: loss improved from 11413413.00000 to 11307446.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 11307446.0000 - mae: 2572.6165
Epoch 12/50
2537/2552 [============================>.] - ETA: 0s - loss: 11263909.0000 - mae: 2565.8721
Epoch 00012: loss improved from 11307446.00000 to 11261945.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 11261945.0000 - mae: 2565.8633
Epoch 13/50
2535/2552 [============================>.] - ETA: 0s - loss: 11052614.0000 - mae: 2540.7686
Epoch 00013: loss improved from 11261945.00000 to 11048080.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 11048080.0000 - mae: 2540.4583
Epoch 14/50
2552/2552 [==============================] - ETA: 0s - loss: 10977256.0000 - mae: 2529.8826
Epoch 00014: loss improved from 11048080.00000 to 10977256.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 6s 3ms/step - loss: 10977256.0000 - mae: 2529.8826
Epoch 15/50
2552/2552 [==============================] - ETA: 0s - loss: 10978681.0000 - mae: 2531.9739
Epoch 00015: loss did not improve from 10977256.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10978681.0000 - mae: 2531.9739
Epoch 16/50
2547/2552 [============================>.] - ETA: 0s - loss: 10926982.0000 - mae: 2524.9026
Epoch 00016: loss improved from 10977256.00000 to 10929672.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 10929672.0000 - mae: 2525.2720
Epoch 17/50
2540/2552 [============================>.] - ETA: 0s - loss: 10933453.0000 - mae: 2528.8457
Epoch 00017: loss did not improve from 10929672.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10932690.0000 - mae: 2528.5454
Epoch 18/50
2552/2552 [==============================] - ETA: 0s - loss: 10958056.0000 - mae: 2526.7356
Epoch 00018: loss did not improve from 10929672.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10958056.0000 - mae: 2526.7356
Epoch 19/50
2549/2552 [============================>.] - ETA: 0s - loss: 10698152.0000 - mae: 2501.9910
Epoch 00019: loss improved from 10929672.00000 to 10698043.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 10698043.0000 - mae: 2501.8535
Epoch 20/50
2552/2552 [==============================] - ETA: 0s - loss: 10590126.0000 - mae: 2482.8389
Epoch 00020: loss improved from 10698043.00000 to 10590126.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 10590126.0000 - mae: 2482.8389
Epoch 21/50
2549/2552 [============================>.] - ETA: 0s - loss: 10632596.0000 - mae: 2487.5144
```

```
Epoch 00021: loss did not improve from 10590126.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10635348.0000 - mae: 2487.8513
Epoch 22/50
2540/2552 [============================>.] - ETA: 0s - loss: 10298840.0000 - mae: 2456.2319
Epoch 00022: loss improved from 10590126.00000 to 10307616.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 8s 3ms/step - loss: 10307616.0000 - mae: 2456.9985
Epoch 23/50
2536/2552 [============================>.] - ETA: 0s - loss: 10506089.0000 - mae: 2474.8745
Epoch 00023: loss did not improve from 10307616.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10507997.0000 - mae: 2475.2720
Epoch 24/50
2535/2552 [============================>.] - ETA: 0s - loss: 10498220.0000 - mae: 2474.0825
Epoch 00024: loss did not improve from 10307616.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10495792.0000 - mae: 2473.3821
Epoch 25/50
2531/2552 [============================>.] - ETA: 0s - loss: 10346649.0000 - mae: 2447.7864
Epoch 00025: loss did not improve from 10307616.00000

Epoch 00025: ReduceLROnPlateau reducing learning rate to 0.00020000000949949026.
2552/2552 [==============================] - 6s 2ms/step - loss: 10349193.0000 - mae: 2447.5525
Epoch 26/50
2535/2552 [============================>.] - ETA: 0s - loss: 10236920.0000 - mae: 2439.5657
Epoch 00026: loss improved from 10307616.00000 to 10235311.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 10235311.0000 - mae: 2438.8916
Epoch 27/50
2544/2552 [============================>.] - ETA: 0s - loss: 10322588.0000 - mae: 2444.4839
Epoch 00027: loss did not improve from 10235311.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10320304.0000 - mae: 2444.3755
Epoch 28/50
2535/2552 [============================>.] - ETA: 0s - loss: 10134172.0000 - mae: 2421.0508
Epoch 00028: loss improved from 10235311.00000 to 10137645.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 10137645.0000 - mae: 2421.6196
Epoch 29/50
2543/2552 [============================>.] - ETA: 0s - loss: 10211897.0000 - mae: 2427.2002
Epoch 00029: loss did not improve from 10137645.00000
2552/2552 [==============================] - 6s 3ms/step - loss: 10214770.0000 - mae: 2427.6831
Epoch 30/50
2532/2552 [============================>.] - ETA: 0s - loss: 10187428.0000 - mae: 2431.4751
Epoch 00030: loss did not improve from 10137645.00000
2552/2552 [==============================] - 6s 2ms/step - loss: 10181649.0000 - mae: 2430.4302
Epoch 31/50
2539/2552 [============================>.] - ETA: 0s - loss: 10096846.0000 - mae: 2416.5667
Epoch 00031: loss improved from 10137645.00000 to 10100567.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 6s 3ms/step - loss: 10100567.0000 - mae: 2416.8806
Epoch 32/50
2532/2552 [============================>.] - ETA: 0s - loss: 10075171.0000 - mae: 2418.5813
Epoch 00032: loss improved from 10100567.00000 to 10077890.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 10077890.0000 - mae: 2418.4783
Epoch 33/50
2545/2552 [============================>.] - ETA: 0s - loss: 10115017.0000 - mae: 2424.2163
Epoch 00033: loss did not improve from 10077890.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10114135.0000 - mae: 2424.3064
Epoch 34/50
2552/2552 [==============================] - ETA: 0s - loss: 10164784.0000 - mae: 2426.7688
Epoch 00034: loss did not improve from 10077890.00000
2552/2552 [==============================] - 6s 2ms/step - loss: 10164784.0000 - mae: 2426.7688
Epoch 35/50
2545/2552 [============================>.] - ETA: 0s - loss: 10078321.0000 - mae: 2420.8113
Epoch 00035: loss improved from 10077890.00000 to 10076715.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 10076715.0000 - mae: 2420.5137
Epoch 36/50
2544/2552 [============================>.] - ETA: 0s - loss: 10142046.0000 - mae: 2419.8135
Epoch 00036: loss did not improve from 10076715.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 10133889.0000 - mae: 2418.8313
Epoch 37/50
2545/2552 [============================>.] - ETA: 0s - loss: 10010784.0000 - mae: 2414.0059
Epoch 00037: loss improved from 10076715.00000 to 10006042.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 10006042.0000 - mae: 2413.4749
Epoch 38/50
2537/2552 [============================>.] - ETA: 0s - loss: 9931496.0000 - mae: 2407.3765
Epoch 00038: loss improved from 10006042.00000 to 9934969.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 9934969.0000 - mae: 2407.9373
Epoch 39/50
2534/2552 [============================>.] - ETA: 0s - loss: 9954558.0000 - mae: 2406.0986
Epoch 00039: loss did not improve from 9934969.00000
2552/2552 [==============================] - 6s 3ms/step - loss: 9953788.0000 - mae: 2406.5073
Epoch 40/50
2537/2552 [============================>.] - ETA: 0s - loss: 9973398.0000 - mae: 2408.9146
Epoch 00040: loss did not improve from 9934969.00000
2552/2552 [==============================] - 6s 3ms/step - loss: 9974653.0000 - mae: 2408.7122
Epoch 41/50
2538/2552 [============================>.] - ETA: 0s - loss: 9974096.0000 - mae: 2409.1743
Epoch 00041: loss did not improve from 9934969.00000

Epoch 00041: ReduceLROnPlateau reducing learning rate to 4.0000001899898055e-05.
2552/2552 [==============================] - 6s 3ms/step - loss: 9973910.0000 - mae: 2408.9688
Epoch 42/50
2537/2552 [============================>.] - ETA: 0s - loss: 10000799.0000 - mae: 2409.8755
Epoch 00042: loss did not improve from 9934969.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 9998048.0000 - mae: 2409.2266
Epoch 43/50
2537/2552 [============================>.] - ETA: 0s - loss: 10013632.0000 - mae: 2407.5769
Epoch 00043: loss did not improve from 9934969.00000
2552/2552 [==============================] - 6s 2ms/step - loss: 10015456.0000 - mae: 2407.8313
Epoch 44/50
2548/2552 [============================>.] - ETA: 0s - loss: 9911859.0000 - mae: 2393.1902
Epoch 00044: loss improved from 9934969.00000 to 9918804.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 9918804.0000 - mae: 2393.8408
```

```
Epoch 45/50
2538/2552 [============================>.] - ETA: 0s - loss: 9971439.0000 - mae: 2406.7202
Epoch 00045: loss did not improve from 9918804.00000
2552/2552 [==============================] - 6s 3ms/step - loss: 9963577.0000 - mae: 2405.7185
Epoch 46/50
2545/2552 [============================>.] - ETA: 0s - loss: 9979100.0000 - mae: 2410.5796
Epoch 00046: loss did not improve from 9918804.00000
2552/2552 [==============================] - 6s 3ms/step - loss: 9983546.0000 - mae: 2411.0754
Epoch 47/50
2542/2552 [============================>.] - ETA: 0s - loss: 10008547.0000 - mae: 2409.3301
Epoch 00047: loss did not improve from 9918804.00000

Epoch 00047: ReduceLROnPlateau reducing learning rate to 8.000000525498762e-06.
2552/2552 [==============================] - 6s 3ms/step - loss: 10012625.0000 - mae: 2409.4800
Epoch 48/50
2541/2552 [============================>.] - ETA: 0s - loss: 9879518.0000 - mae: 2395.3740
Epoch 00048: loss improved from 9918804.00000 to 9882870.00000, saving model to Sales_ann_with_dropout.h5
2552/2552 [==============================] - 7s 3ms/step - loss: 9882870.0000 - mae: 2395.6499
Epoch 49/50
2552/2552 [==============================] - ETA: 0s - loss: 9987444.0000 - mae: 2410.5007
Epoch 00049: loss did not improve from 9882870.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 9987444.0000 - mae: 2410.5007
Epoch 50/50
2551/2552 [============================>.] - ETA: 0s - loss: 9974626.0000 - mae: 2403.5461
Epoch 00050: loss did not improve from 9882870.00000
2552/2552 [==============================] - 7s 3ms/step - loss: 9977927.0000 - mae: 2403.8557
```

In [158…

```python
model_2 = load_model('Sales_ann_with_dropout.h5')

y_pred = model_2.predict(x_test)
math.sqrt(mean_squared_error(y_test,y_pred))
```

Out[158…

```
1938.6130502881572
```

- When I used dropout root mean squared error increased.
- It is not useful for our model.

3. Find the best setting of neural net that minimizes the loss and can predict the sales best. Use techniques like Grid search, cross-validation and Random search.

In [159…

```python
#cross validation
```

In [160…

```python
def modelkf(x_train,y_train,x_test,y_test):

    model = Sequential()
    model.add(layers.Dense(32, activation='elu', input_shape = (x_train.shape[1],)))
    model.add(layers.Dense(64, activation='elu'))
    model.add(layers.Dense(64, activation='elu'))
    model.add(layers.BatchNormalization())

    ## block 2
    model.add(layers.Dense(128, activation='elu'))
    model.add(layers.Dense(128, activation='elu'))
    model.add(layers.BatchNormalization())

    ## block 3
    model.add(layers.Dense(256, activation='elu'))
    model.add(layers.Dense(256, activation='elu'))
    model.add(layers.BatchNormalization())
    model.add(layers.Dropout(0.8))

    ## block 4
    model.add(layers.Dense(128, activation='elu'))
    model.add(layers.Dense(128, activation='elu'))
    model.add(layers.BatchNormalization())
    model.add(layers.Dropout(0.8))

    ## block 5
    model.add(layers.Dense(64, activation='elu'))
    model.add(layers.Dense(64, activation='elu'))
    model.add(layers.Dense(32, activation='elu'))
    model.add(layers.Dropout(0.8))
    model.add(layers.Dense(1))

    model.compile(loss='mse',
              optimizer = Adam(learning_rate=0.001),
              metrics=['mae'])

    model.fit(x_train,y_train,epochs=50,batch_size=20)

    y_pred = model.predict(x_test)

    return (math.sqrt(mean_squared_error(y_test,y_pred)))
```

In [164…

```python
score_kf_ann = []
kf = StratifiedKFold(n_splits=4,random_state=100)
for train_index,test_index in kf.split(x,y):
    x_train,x_test,y_train,y_test = train_test_split(np.array(x),np.array(y),test_size = 0.3,random_state=53)
    score_kf_ann.append(modelkf(x_train,y_train,x_test,y_test))
```

```
Epoch 1/50
```

```
2552/2552 [==============================] - 6s 2ms/step - loss: 17566532.0000 - mae: 3377.9365
Epoch 2/50
2552/2552 [==============================] - 6s 2ms/step - loss: 14413125.0000 - mae: 2977.6260
Epoch 3/50
2552/2552 [==============================] - 6s 2ms/step - loss: 13257686.0000 - mae: 2819.8572
Epoch 4/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12824226.0000 - mae: 2774.3896
Epoch 5/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12856587.0000 - mae: 2773.2393
Epoch 6/50
2552/2552 [==============================] - 7s 3ms/step - loss: 12702968.0000 - mae: 2742.2686
Epoch 7/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12494999.0000 - mae: 2728.3423
Epoch 8/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12321444.0000 - mae: 2702.2090
Epoch 9/50
2552/2552 [==============================] - 6s 3ms/step - loss: 12245067.0000 - mae: 2685.0029
Epoch 10/50
2552/2552 [==============================] - 7s 3ms/step - loss: 12196711.0000 - mae: 2686.3503
Epoch 11/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11972784.0000 - mae: 2652.3049
Epoch 12/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12074139.0000 - mae: 2671.6843
Epoch 13/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11766183.0000 - mae: 2626.1831
Epoch 14/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11898129.0000 - mae: 2642.4058
Epoch 15/50
2552/2552 [==============================] - 6s 3ms/step - loss: 11656050.0000 - mae: 2616.9380
Epoch 16/50
2552/2552 [==============================] - 7s 3ms/step - loss: 11678933.0000 - mae: 2619.3018
Epoch 17/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11513614.0000 - mae: 2598.8547
Epoch 18/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11529306.0000 - mae: 2604.3337
Epoch 19/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11547620.0000 - mae: 2599.6025
Epoch 20/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11385510.0000 - mae: 2580.3074
Epoch 21/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11218248.0000 - mae: 2561.9421
Epoch 22/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11119511.0000 - mae: 2545.8469
Epoch 23/50
2552/2552 [==============================] - 6s 3ms/step - loss: 11116573.0000 - mae: 2546.0977
Epoch 24/50
2552/2552 [==============================] - 6s 3ms/step - loss: 10970522.0000 - mae: 2535.3154
Epoch 25/50
2552/2552 [==============================] - 9s 3ms/step - loss: 10958937.0000 - mae: 2528.1348
Epoch 26/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10967729.0000 - mae: 2524.0779
Epoch 27/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10801237.0000 - mae: 2509.8208
Epoch 28/50
2552/2552 [==============================] - 8s 3ms/step - loss: 10813656.0000 - mae: 2504.3167
Epoch 29/50
2552/2552 [==============================] - 8s 3ms/step - loss: 10724290.0000 - mae: 2497.9106
Epoch 30/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10687492.0000 - mae: 2490.1313
Epoch 31/50
2552/2552 [==============================] - 6s 3ms/step - loss: 10652594.0000 - mae: 2485.8013
Epoch 32/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10500441.0000 - mae: 2468.3936
Epoch 33/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10415348.0000 - mae: 2465.0557
Epoch 34/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10408022.0000 - mae: 2455.9656
Epoch 35/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10337027.0000 - mae: 2448.5195
Epoch 36/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10231469.0000 - mae: 2434.0737
Epoch 37/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10149552.0000 - mae: 2421.1714
Epoch 38/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10155253.0000 - mae: 2427.4475
Epoch 39/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10129573.0000 - mae: 2426.9302
Epoch 40/50
2552/2552 [==============================] - 6s 3ms/step - loss: 10076572.0000 - mae: 2413.3250
Epoch 41/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9916726.0000 - mae: 2390.3591
Epoch 42/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9890037.0000 - mae: 2393.0613
Epoch 43/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9844442.0000 - mae: 2383.4263
Epoch 44/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9915834.0000 - mae: 2394.5034
Epoch 45/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9762171.0000 - mae: 2371.0574
Epoch 46/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9836067.0000 - mae: 2374.6521
Epoch 47/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9676452.0000 - mae: 2363.0876
Epoch 48/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9502549.0000 - mae: 2344.4480
Epoch 49/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9577526.0000 - mae: 2353.5457
Epoch 50/50
```

```
2552/2552 [==============================] - 6s 2ms/step - loss: 9655903.0000 - mae: 2353.7095
Epoch 1/50
2552/2552 [==============================] - 7s 3ms/step - loss: 16884504.0000 - mae: 3282.8550
Epoch 2/50
2552/2552 [==============================] - 7s 3ms/step - loss: 13583743.0000 - mae: 2865.3757
Epoch 3/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12874711.0000 - mae: 2780.7393
Epoch 4/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12448922.0000 - mae: 2716.2039
Epoch 5/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12292633.0000 - mae: 2697.8584
Epoch 6/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12129657.0000 - mae: 2677.4976
Epoch 7/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12179174.0000 - mae: 2677.1648
Epoch 8/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11932257.0000 - mae: 2649.2422
Epoch 9/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11836843.0000 - mae: 2642.2026
Epoch 10/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11732250.0000 - mae: 2632.2380
Epoch 11/50
2552/2552 [==============================] - 7s 3ms/step - loss: 11683431.0000 - mae: 2617.6111
Epoch 12/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11613225.0000 - mae: 2611.6160
Epoch 13/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11457106.0000 - mae: 2590.4065
Epoch 14/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11402754.0000 - mae: 2585.0781
Epoch 15/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11345341.0000 - mae: 2575.1021
Epoch 16/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11287931.0000 - mae: 2570.0281
Epoch 17/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11079540.0000 - mae: 2544.9917
Epoch 18/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11173065.0000 - mae: 2555.3584
Epoch 19/50
2552/2552 [==============================] - 6s 3ms/step - loss: 10944918.0000 - mae: 2531.4187
Epoch 20/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10931648.0000 - mae: 2524.8218
Epoch 21/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10914286.0000 - mae: 2522.4988
Epoch 22/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10790132.0000 - mae: 2504.6360
Epoch 23/50
2552/2552 [==============================] - 6s 3ms/step - loss: 10749350.0000 - mae: 2503.8442
Epoch 24/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10667798.0000 - mae: 2490.4990
Epoch 25/50
2552/2552 [==============================] - 6s 3ms/step - loss: 10674207.0000 - mae: 2490.5974
Epoch 26/50
2552/2552 [==============================] - 6s 3ms/step - loss: 10620540.0000 - mae: 2482.1846
Epoch 27/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10464455.0000 - mae: 2463.9490
Epoch 28/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10383692.0000 - mae: 2453.6833
Epoch 29/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10395321.0000 - mae: 2450.7388
Epoch 30/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10241138.0000 - mae: 2433.8442
Epoch 31/50
2552/2552 [==============================] - 6s 3ms/step - loss: 10196844.0000 - mae: 2431.4590
Epoch 32/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10138275.0000 - mae: 2428.2532
Epoch 33/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10050278.0000 - mae: 2406.3906
Epoch 34/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10055814.0000 - mae: 2411.4702
Epoch 35/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9872073.0000 - mae: 2398.0591
Epoch 36/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9930746.0000 - mae: 2402.9763
Epoch 37/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9879443.0000 - mae: 2390.2324
Epoch 38/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9709597.0000 - mae: 2367.9446
Epoch 39/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9674239.0000 - mae: 2367.6731
Epoch 40/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9785248.0000 - mae: 2378.1294
Epoch 41/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9683441.0000 - mae: 2367.0459
Epoch 42/50
2552/2552 [==============================] - 6s 3ms/step - loss: 9604216.0000 - mae: 2348.4836
Epoch 43/50
2552/2552 [==============================] - 6s 3ms/step - loss: 9509667.0000 - mae: 2344.3069
Epoch 44/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9517353.0000 - mae: 2339.6970
Epoch 45/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9401689.0000 - mae: 2321.5459
Epoch 46/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9375852.0000 - mae: 2324.6509
Epoch 47/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9389733.0000 - mae: 2323.0637
Epoch 48/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9277929.0000 - mae: 2304.7224
Epoch 49/50
```

```
2552/2552 [==============================] - 6s 2ms/step - loss: 9222029.0000 - mae: 2302.3789
Epoch 50/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9314222.0000 - mae: 2310.5757
Epoch 1/50
2552/2552 [==============================] - 6s 2ms/step - loss: 15832158.0000 - mae: 3191.6045
Epoch 2/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12805516.0000 - mae: 2819.0872
Epoch 3/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11848999.0000 - mae: 2664.5554
Epoch 4/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11280316.0000 - mae: 2570.9939
Epoch 5/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11007945.0000 - mae: 2544.1697
Epoch 6/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10879562.0000 - mae: 2521.1245
Epoch 7/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10944423.0000 - mae: 2534.3159
Epoch 8/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10706482.0000 - mae: 2500.8440
Epoch 9/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10709613.0000 - mae: 2504.5005
Epoch 10/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10687799.0000 - mae: 2498.8010
Epoch 11/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10620401.0000 - mae: 2490.4619
Epoch 12/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10430875.0000 - mae: 2466.1118
Epoch 13/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10294827.0000 - mae: 2455.3137
Epoch 14/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10306225.0000 - mae: 2450.7505
Epoch 15/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10265098.0000 - mae: 2447.7026
Epoch 16/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10257273.0000 - mae: 2433.9993
Epoch 17/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10237582.0000 - mae: 2437.9363
Epoch 18/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10213595.0000 - mae: 2430.9033
Epoch 19/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10144026.0000 - mae: 2426.4299
Epoch 20/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10031101.0000 - mae: 2413.5938
Epoch 21/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9946373.0000 - mae: 2401.9331
Epoch 22/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9862399.0000 - mae: 2395.5625
Epoch 23/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9903685.0000 - mae: 2396.7937
Epoch 24/50
2552/2552 [==============================] - 6s 3ms/step - loss: 9876878.0000 - mae: 2386.9436
Epoch 25/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9896874.0000 - mae: 2396.5615
Epoch 26/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9693521.0000 - mae: 2373.6221
Epoch 27/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9672782.0000 - mae: 2369.1860
Epoch 28/50
2552/2552 [==============================] - 6s 3ms/step - loss: 9648526.0000 - mae: 2367.1008
Epoch 29/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9533269.0000 - mae: 2350.8022
Epoch 30/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9533561.0000 - mae: 2353.6941
Epoch 31/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9573922.0000 - mae: 2357.2705
Epoch 32/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9530800.0000 - mae: 2346.4885
Epoch 33/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9392158.0000 - mae: 2333.2302
Epoch 34/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9372496.0000 - mae: 2326.0989
Epoch 35/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9298345.0000 - mae: 2319.3040
Epoch 36/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9223981.0000 - mae: 2311.1248
Epoch 37/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9196602.0000 - mae: 2309.8469
Epoch 38/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9188497.0000 - mae: 2297.8704
Epoch 39/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9171496.0000 - mae: 2298.4294
Epoch 40/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9154146.0000 - mae: 2291.9849
Epoch 41/50
2552/2552 [==============================] - 6s 3ms/step - loss: 9077735.0000 - mae: 2286.6165
Epoch 42/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9092909.0000 - mae: 2289.1409
Epoch 43/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9009198.0000 - mae: 2272.9507
Epoch 44/50
2552/2552 [==============================] - 6s 2ms/step - loss: 8878445.0000 - mae: 2267.7686
Epoch 45/50
2552/2552 [==============================] - 7s 3ms/step - loss: 8998868.0000 - mae: 2277.1655
Epoch 46/50
2552/2552 [==============================] - 6s 2ms/step - loss: 8876458.0000 - mae: 2264.0166
Epoch 47/50
2552/2552 [==============================] - 6s 2ms/step - loss: 8866974.0000 - mae: 2255.6162
Epoch 48/50
```

```
2552/2552 [==============================] - 6s 2ms/step - loss: 8786142.0000 - mae: 2246.2830
Epoch 49/50
2552/2552 [==============================] - 6s 2ms/step - loss: 8875744.0000 - mae: 2258.7148
Epoch 50/50
2552/2552 [==============================] - 6s 2ms/step - loss: 8850392.0000 - mae: 2250.0864
Epoch 1/50
2552/2552 [==============================] - 6s 2ms/step - loss: 17837674.0000 - mae: 3395.9463
Epoch 2/50
2552/2552 [==============================] - 6s 2ms/step - loss: 14216090.0000 - mae: 2963.3379
Epoch 3/50
2552/2552 [==============================] - 6s 2ms/step - loss: 13585268.0000 - mae: 2863.9958
Epoch 4/50
2552/2552 [==============================] - 7s 3ms/step - loss: 13218924.0000 - mae: 2806.9775
Epoch 5/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12638703.0000 - mae: 2746.7881
Epoch 6/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12614963.0000 - mae: 2738.9614
Epoch 7/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12486517.0000 - mae: 2716.1177
Epoch 8/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12317018.0000 - mae: 2702.1919
Epoch 9/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12287008.0000 - mae: 2685.1597
Epoch 10/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12187813.0000 - mae: 2684.5249
Epoch 11/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12022763.0000 - mae: 2661.8286
Epoch 12/50
2552/2552 [==============================] - 6s 2ms/step - loss: 12035305.0000 - mae: 2658.8931
Epoch 13/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11845768.0000 - mae: 2632.1892
Epoch 14/50
2552/2552 [==============================] - 7s 3ms/step - loss: 11710547.0000 - mae: 2624.3416
Epoch 15/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11703332.0000 - mae: 2620.0723
Epoch 16/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11514197.0000 - mae: 2601.4104
Epoch 17/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11584175.0000 - mae: 2602.9097
Epoch 18/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11547387.0000 - mae: 2599.3206
Epoch 19/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11420318.0000 - mae: 2586.4636
Epoch 20/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11423256.0000 - mae: 2580.7239
Epoch 21/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11252576.0000 - mae: 2553.8804
Epoch 22/50
2552/2552 [==============================] - 6s 2ms/step - loss: 11066578.0000 - mae: 2536.4375
Epoch 23/50
2552/2552 [==============================] - 7s 3ms/step - loss: 11118775.0000 - mae: 2548.8286
Epoch 24/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10984117.0000 - mae: 2523.8801
Epoch 25/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10932860.0000 - mae: 2526.0134
Epoch 26/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10887445.0000 - mae: 2516.2024
Epoch 27/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10797892.0000 - mae: 2506.2734
Epoch 28/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10722296.0000 - mae: 2489.0110
Epoch 29/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10757751.0000 - mae: 2500.0737
Epoch 30/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10589613.0000 - mae: 2476.2490
Epoch 31/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10579409.0000 - mae: 2482.3616
Epoch 32/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10397580.0000 - mae: 2450.6057
Epoch 33/50
2552/2552 [==============================] - 7s 3ms/step - loss: 10288356.0000 - mae: 2444.8892
Epoch 34/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10309119.0000 - mae: 2440.1094
Epoch 35/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10290784.0000 - mae: 2444.7590
Epoch 36/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10171493.0000 - mae: 2433.0620
Epoch 37/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10225411.0000 - mae: 2434.8694
Epoch 38/50
2552/2552 [==============================] - 6s 2ms/step - loss: 10117674.0000 - mae: 2420.9517
Epoch 39/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9951164.0000 - mae: 2395.5596
Epoch 40/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9908467.0000 - mae: 2389.8723
Epoch 41/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9921104.0000 - mae: 2383.3489
Epoch 42/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9759175.0000 - mae: 2372.5828
Epoch 43/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9805114.0000 - mae: 2367.2539
Epoch 44/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9757836.0000 - mae: 2371.1882
Epoch 45/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9661389.0000 - mae: 2353.8367
Epoch 46/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9699932.0000 - mae: 2359.6655
Epoch 47/50
```

```
2552/2552 [==============================] - 8s 3ms/step - loss: 9558757.0000 - mae: 2348.5386
Epoch 48/50
2552/2552 [==============================] - 7s 3ms/step - loss: 9459721.0000 - mae: 2334.2488
Epoch 49/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9417600.0000 - mae: 2323.8438
Epoch 50/50
2552/2552 [==============================] - 6s 2ms/step - loss: 9413961.0000 - mae: 2324.5425
```

In [165…

```
score_kf_ann
```

Out[165…

```
[67055.82603188697, 1705.5901814265628, 5139.764923450266, 2165.0387761147335]
```

In [168…

```
RMSE = sum(score_kf_ann)/len(score_kf_ann)
RMSE
```

Out[168…

```
19016.554978219632
```

- Prediction is quite good in comparision to mean.

In [ ]: