Predicting Retail Sales

Author: Oprea Dragos
Date:21/04/2019

Introduction

This is my analysis on the Kaggle retail dataset, trying to predict Retail sales for the next 8 weeks for each of the 45 stores and for the entire network

How the analysis is organised

My goal is to predict the next 8 weeks of Sales at store level and for the entire Store chain.
To get there, I split the analysis in the following steps:

1. Data exploration and cleanup
2. Data pre-processing and features transformations
3. Modelling
4. Prediction

Define the problem

Before I start, I need to understand why do we need to predict the retail sales? What is the benefit for the company in knowing the retail sales 8 weeks ahead. A brainstorming would help framing the problem and defining the right approach, algorithms and performance metrics

My intention for this exercise task is to use algorithms I understand and to showcase what analysis I can do:

- The data is labeled already - use Supervised Learning category of machine learning
- Predict sales using Linear Regression
- Measure Performance of the model using mean squared error

Data exploration and cleanup

Data

The kaggle retail dataset consists in 3 tables
['Features data set.csv', 'sales data-set.csv', 'stores data-set.csv']

Stores
stores data-set.csv

- Anonymized information about 45 stores, indicating the type and size of the store.
- Stores: are numbered 1 - 45
- Types: I see there are 3 types of stores A,B,C apparently categorical based on the size
- Size: I think it is the store capacity. Size is ranging from ~30k -220k

Features
Features data set.csv

Contains additional data related to the store, department, and regional activity for the given dates.

- Store: store number 1 - 45
- Date: Date of the week when the data was recorded
- mperature: average temperature in the region
- Fuel_Price: cost of fuel in the region
- MarkDown1-5: - anonymized data related to promotional markdowns. MarkDown data is only available after Nov 2011 and is not available for all stores all the time. Any missing value is marked with an NA
- CPI: the consumer price index
- Unemployment: the unemployment rate
- IsHoliday: boolean, whether the week is a special holiday week

Sales
sales data-set.csv

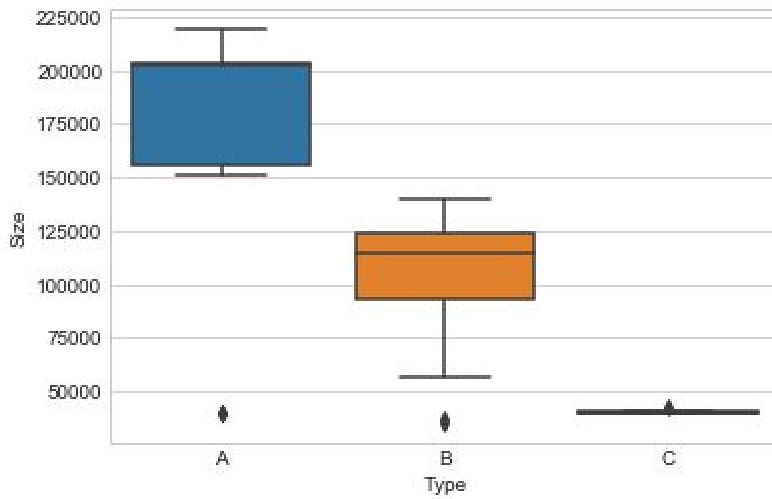
- Store - store number
- Dept - department number
- Date – week number
- Weekly_Sales - sales for the given department in the given store
- IsHoliday - whether the week is a special holiday week

Take a peek at the data

Stores
There are 3 types of stores, each type belongs to a size range. Stores count by type:
A 22
B 17
C 6

```
stores.head()
```

	Store	Type	Size
0	1	A	151315
1	2	A	202307
2	3	B	37392
3	4	A	205863
4	5	B	34875



Features

On features there are 8190 entries, some of the columns like Markdown, CPI and Unemployment have missing data that we'll have to handle.

```
features.head()
```

	Store	Date	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment	IsHoliday
0	1	05/02/2010	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106	False
1	1	12/02/2010	38.51	2.548	NaN	NaN	NaN	NaN	NaN	211.242170	8.106	True
2	1	19/02/2010	39.93	2.514	NaN	NaN	NaN	NaN	NaN	211.289143	8.106	False
3	1	26/02/2010	46.63	2.561	NaN	NaN	NaN	NaN	NaN	211.319643	8.106	False
4	1	05/03/2010	46.50	2.625	NaN	NaN	NaN	NaN	NaN	211.350143	8.106	False

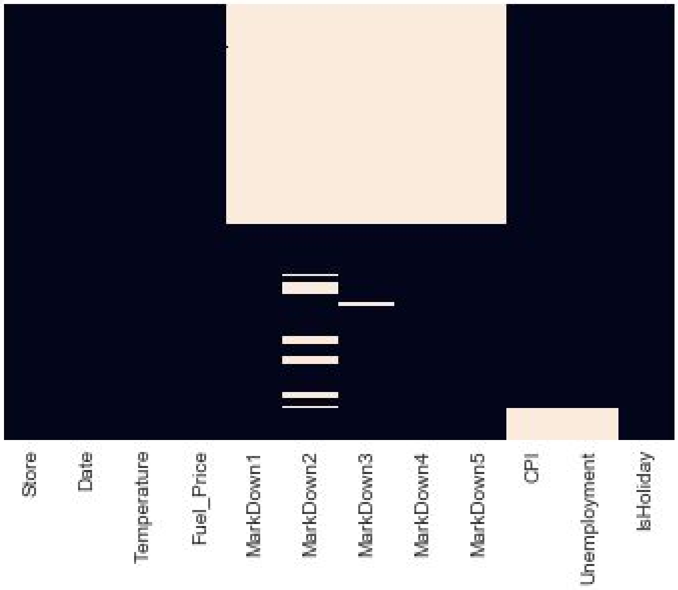
```
features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8190 entries, 0 to 8189
Data columns (total 12 columns):
Store           8190 non-null int64
Date            8190 non-null object
Temperature     8190 non-null float64
Fuel_Price      8190 non-null float64
MarkDown1       4032 non-null float64
MarkDown2       2921 non-null float64
MarkDown3       3613 non-null float64
MarkDown4       3464 non-null float64
MarkDown5       4050 non-null float64
CPI             7605 non-null float64
Unemployment    7605 non-null float64
IsHoliday       8190 non-null bool
dtypes: bool(1), float64(9), int64(1), object(1)
memory usage: 711.9+ KB
```

```
features.isna().sum()
```

```
Store            0
Date             0
Temperature      0
Fuel_Price       0
CPI            585
Unemployment   585
IsHoliday        0
dtype: int64
```

Here is an series overview on the missing data for one Store (dark - data is present):



Features Cleanup
Markdown: drop the markdowns columns because I do not intend to use it
CPI: Fillna-s backward, do not drop the lines because we loose timeseries data
Unemployment, do not drop the lines because we loose timeseries data

```
for column in markdown_cols:
    features = features.drop(column,axis=1)

features['CPI'] = features['CPI'].fillna(method='pad')
features['Unemployment'] = features['Unemployment'].fillna(method='pad')
```

Sales
Sales data is the biggest dataset. Having 421570 entries. The data is clean.

```
sales.head()
```

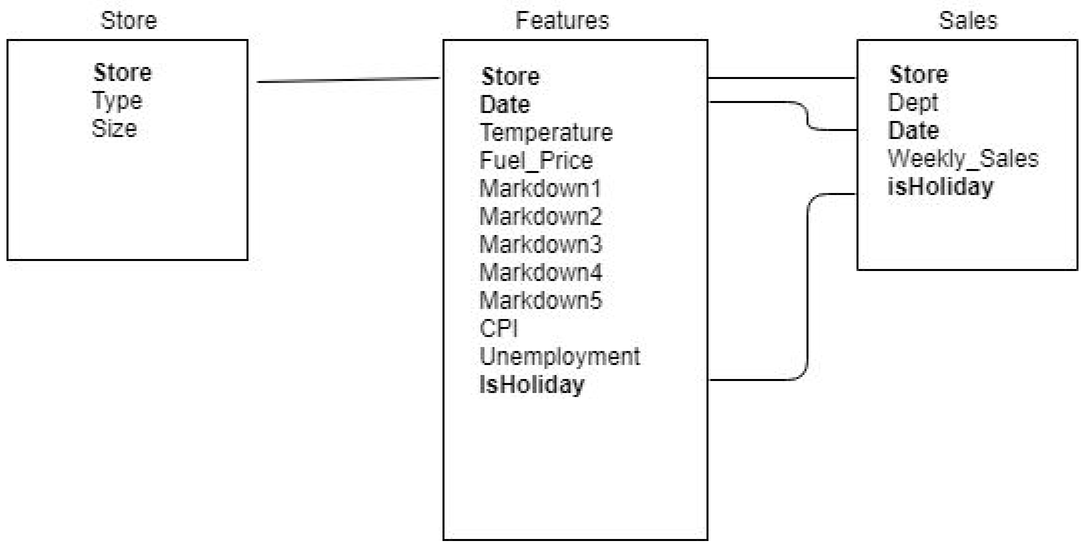
	Store	Dept	Date	Weekly_Sales	IsHoliday
0	1	1	05/02/2010	24924.50	False
1	1	1	12/02/2010	46039.49	True
2	1	1	19/02/2010	41595.55	False
3	1	1	26/02/2010	19403.54	False
4	1	1	05/03/2010	21827.90	False

```
sales.info()
```

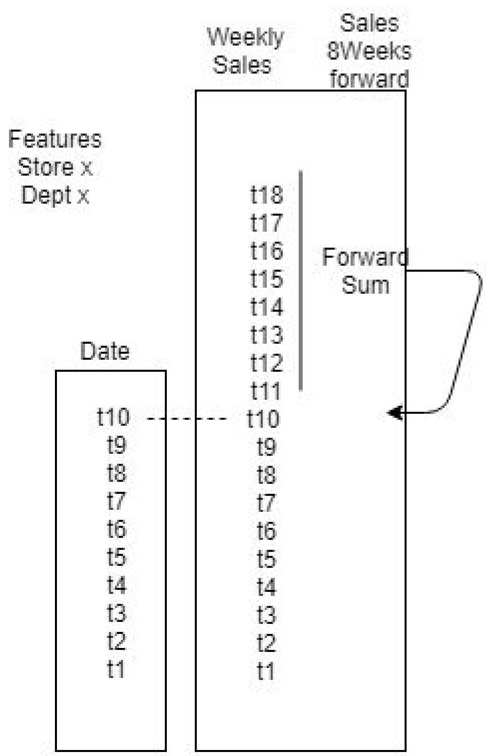
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 421570 entries, 0 to 421569
Data columns (total 5 columns):
Store           421570 non-null int64
Dept            421570 non-null int64
Date            421570 non-null object
Weekly_Sales    421570 non-null float64
IsHoliday       421570 non-null bool
dtypes: bool(1), float64(1), int64(2), object(1)
memory usage: 13.3+ MB
```

Data pre-processing and features transformations

I am de-normalising the datasets by merging all three into a single dataset called ‘retail’:



Calculate the predicted variable - the 8 weeks forward sales, to be used in training the model



- I perform further cleaning by removing weekly sales that are negative
- turn IsHoliday into an Integer, useful later for numeric computations
- I create the following extra column:

  Year: the year of the observation

  Year-week: for better navigation through the data. The Date information is not really weekly data by the standards as it is not always starting from the same weekday, so this might help in calculations later
- Calculate the Forward sum of sales for the next 8 weeks, this will be the predicted variable

	Store	Dept	Date	Weekly_Sales	IsHoliday	Temperature	Fuel_Price	CPI	Unemployment	Type	Size	Year	Year-Week	Sales_F8W
3007	1	23	2010-01-10	18377.92	0	71.89	2.603	211.671989	7.838	A	151315	2010	2010-02	164102.41
2864	1	22	2010-01-10	8353.58	0	71.89	2.603	211.671989	7.838	A	151315	2010	2010-02	69391.90
2721	1	21	2010-01-10	7880.07	0	71.89	2.603	211.671989	7.838	A	151315	2010	2010-02	67097.08
9955	1	97	2010-01-10	32954.82	0	71.89	2.603	211.671989	7.838	A	151315	2010	2010-02	252620.20
10098	1	98	2010-01-10	10344.16	0	71.89	2.603	211.671989	7.838	A	151315	2010	2010-02	89771.77

Heatmap with correlations to forward sales

The Sales are slightly corelated to FuelPrice and CPI

The Sales are negatively corelated to Unemployment

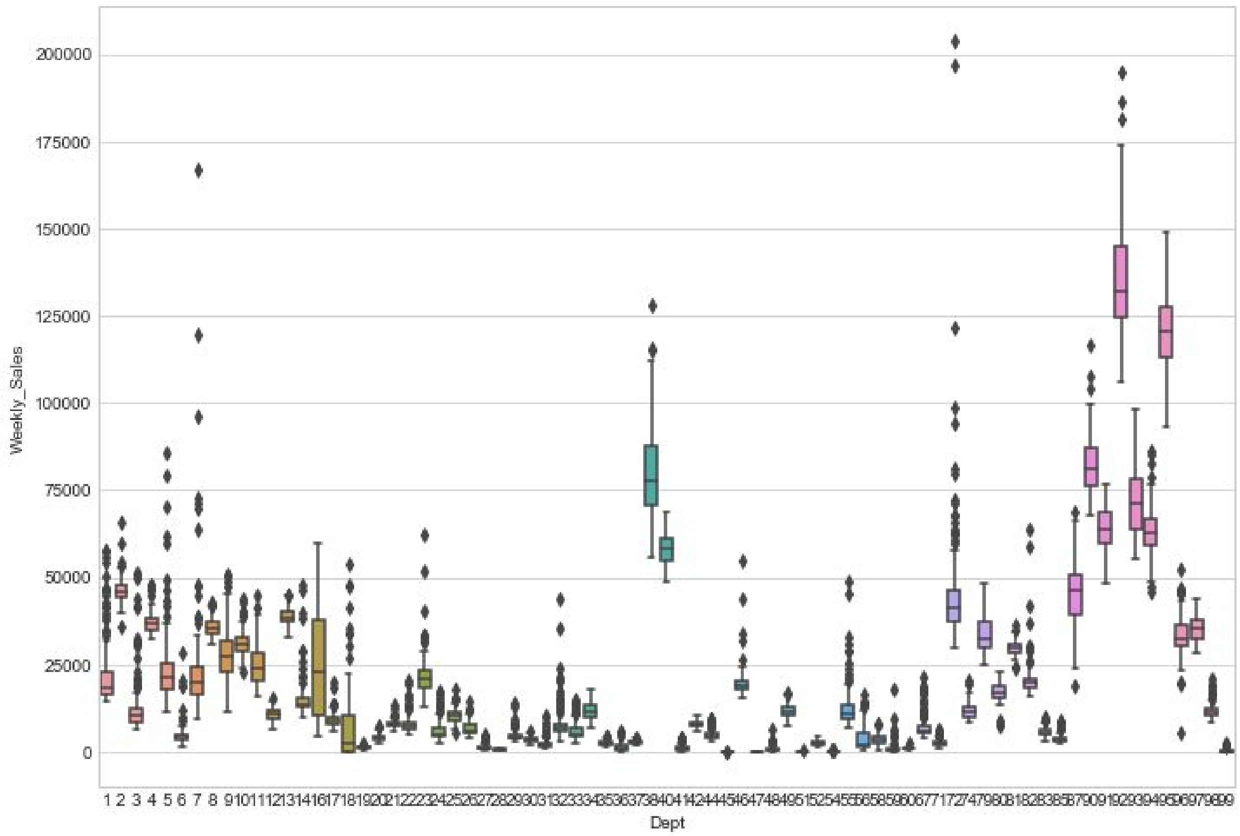
```
col_features = ['Weekly_Sales','Sales_F8W','Temperature','Fuel_Price','CPI','Unemployment',
a4_dims = (11.7, 8.27)
fig, ax = plt.subplots(figsize=a4_dims)
sns.heatmap(retail_S1[col_features].corr(),annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x12f36b6c470>
```

Differences between Departments

sns.boxplot(x='Dept',y='Weekly_Sales',data=retail_S1)

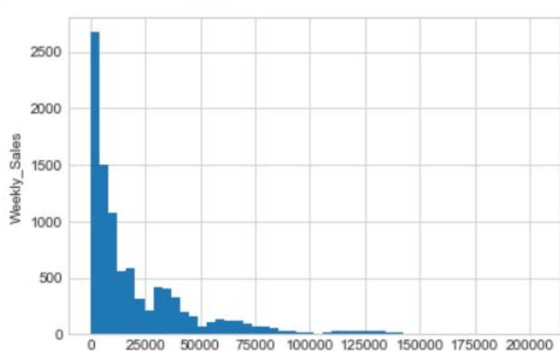
Some departments have a higher weight in the Store total sales



Weekly Sales distribution histogram

Some weeks stand out with lower sales

```
plt.hist(retail_S1['Weekly_Sales'].values,  bins=50)
plt.ylabel('Weekly_Sales');
```



Modelling

For predictions I used Linear regression model
col_X = ['Temperature','Fuel_Price','CPI','Unemployment','Size', 'Dept', 'IsHoliday']
col_y = ['Sales_F8W']

Training Data between: 2010-2012
Test Data: 2013

```
# train Linear regression
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)

# generate predictions
y_test_pred = lin_reg.predict(X_test)
y_test_pred = pd.DataFrame(lin_reg.predict(X_test), index=X_test.index)
```

Results:
mean_squared_error
print(mean_squared_error(y_test, y_test_pred))
#A non-negative floating point value (the best value is 0.0)
My result is not too good: 44411492510.12314

explained_variance_score:
print(explained_variance_score(y_test, y_test_pred))
# Best possible score is 1.0, lower values are worse
0.1414745756828233

Coefficients:
'Temperature' = -57.01353071
'Fuel_Price' = -636.61483545
'CPI' =   845.59944705
'Unemployment' = -1400.45148188
'Size' = 0
'Dept'=2596.82394467
'IsHoliday' =3210.07478693

Prediction for one Store one Department:
Sales forecast for the next 8 weeks on date 2012-09-14

```
# Predict one Date
pred_data = test[(test['Store']==1) & (test['Dept']==1) & (test['Date']=='2012-09-14')]
X_test_one = pred_data[col_X]
y_test_pred = lin_reg.predict(X_test_one)
print('On 2012-09-14 the predicted next 8 weeks sales for Store1, Dept1 is:')
print(y_test_pred)
```

```
On 2012-09-14 the predicted next 8 weeks sales for Store1, Dept1 is:
[[68269.37310616]]
```

Prediction for one Store:

Sales forecast for the next 8 weeks on date 2012-09-14

```
print('Sales forecast for Store 1 for the date 2012-09-14:' + str(results_df['Sales_F8W_Pred'].sum().astype(str)))

results_df.groupby('Store').sum(axis=0).astype(str)
```

```
Sales forecast for Store 1 for the date 2012-09-14:[['102585273.00406696']]
```

Prediction for The entire network:

Sales forecast for the next 8 weeks on date 2012-09-14