Capstone Project
Machine Learning Engineer

Vineet Kumar
Dec 20, 2018

Definition

Project Overview

Focus of this project is to predict sales of weather-sensitive products around the time of major weather events. The features - sales data for 111 products (milk, bread etc.) whose sales may be affected by weather patterns. These 111 products are sold in 45 different Walmart locations and these locations are covered by 20 weather stations that provide weather data. This will help Walmart stores to correctly maintain the level of inventory and avoid being out-of-stock on several items.

This project will use supervised learning of Regression analysis to build models on training data set and make predictions based on given test data. The test data contains dates, store and item numbers for which we need to predict "units" sold. This project is proposed from Walmart's competition at Kaggle: https://www.kaggle.com/c/walmart-recruiting-sales-in-stormy-weather

In 27 countries Walmart operates around 11,450 stores. Extreme weather events, like hurricanes, blizzards, and floods, often have a huge impact on sales at the store and product level. In this challenge Walmart is asking to predict sales of weather- sensitive products like milk, bread and umbrellas during a major weather event.

Problem Statement

The goal of this project is to develop a forecasting model that will allow stores to predict sales of weather-related items for upcoming weather events e.g. rain or snow. This may help stores to maintain optimal inventory and increase sales based on weather patterns.

Here are the high-level steps for solution to the problem

- Download and unzip data from Kaggle: https://www.kaggle.com/c/walmart-recruiting-sales-in-stormy-weather
- Preprocess and join data files.

- Take a random sample of 50 records from training data and exclude this data from training dataset. This sample will be later used to test model accuracy before making the final prediction on provided test file.
- Derive new features from existing weather data.
- Select and normalize features that are important.
- Split training data into training and validation data.
- Tune Hyper-parameters for Regressor.
- Make prediction on validation and Sample set.
- Make predictions on test data and create submission file in predefined format

Metrics

Coefficient of Determination (R-Square): R-Square is to measure the goodness of fit of regression model. The value of R^2 varies between 0 and 1, 0 being worst fit and 1 being best fit.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

Mean Square Error (MSE): MSE is to measure the average square difference between the estimated values and true value. The value closer to 0 is better.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2$$

Mean Absolute Error (MAE): MAE is to measure the model's accuracy. It is the average of absolute difference between true values and predicted values.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \widehat{y_i}|$$

Analysis

Data Exploration

This dataset is based on Kaggle competition. Following is the list of datasets provided:

• key.csv - the relational mapping between stores and the weather stations that cover them.

• sampleSubmission.csv - file that gives the prediction format.

• train.csv - sales data for all stores & dates in the training set.

• test.csv - stores & dates for forecasting (missing 'units', which need to be predicted) - password for the file: Work4WalmarT

• weather.csv - a file containing daily data from NOAA weather stations.

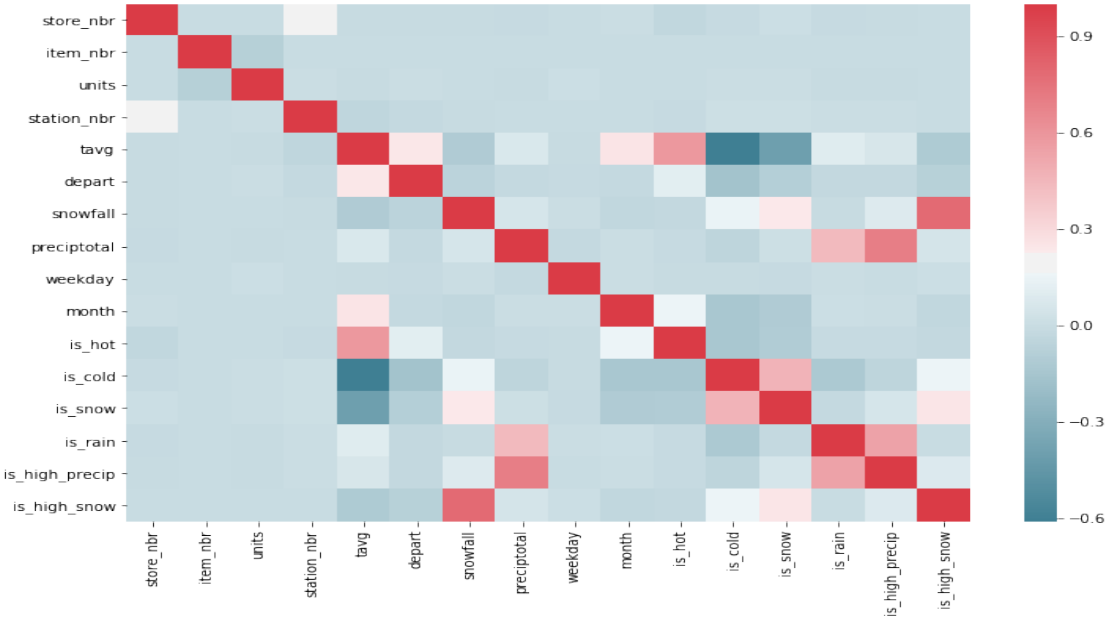
• noaa_weather_qclcd_documentaton.pdf - a guide to understand the data provided in the weather.csv file.

```
Data Summary

Training data:
        Number of Samples: 4617600
        Number of features: 5

Weather data: (20517, 20)
       Weather data from :2012-01-01 to :2014-10-31

Test data:
        Number of Samples: 526917
        Number of features: 4
        Test data dates from :2013-04-01 to :2014-10-26

Number of Weather Stations 20
Number of Walmart Locations 45
```

Exploratory Visualization

Correlation Matrix:

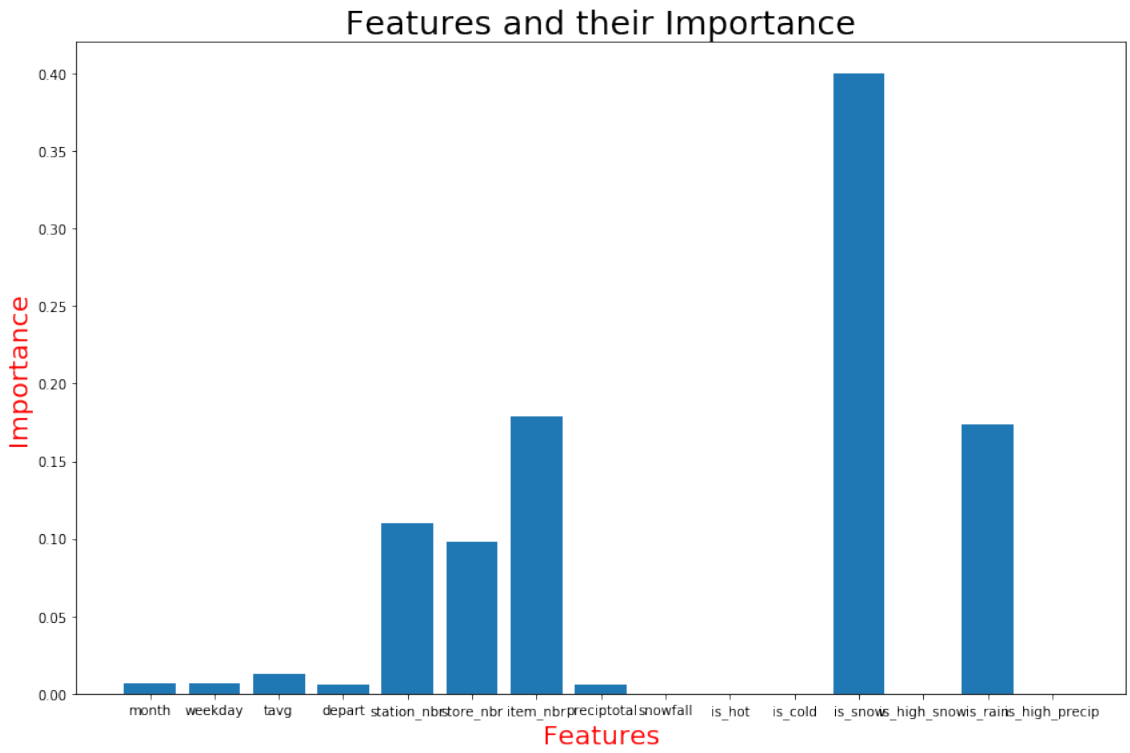
Fig 1:



This correlation heatmap matrix shows feature dependencies with each other. The darker the square color the strong correlation among them, for example – the strong relation between "snowfall" and "is_high_snow" and weak correlation with "month". Other than "rain" and "snowfall" features the above doesn't show much of the correlation among other listed features.

Feature Selection:

Fig 2:



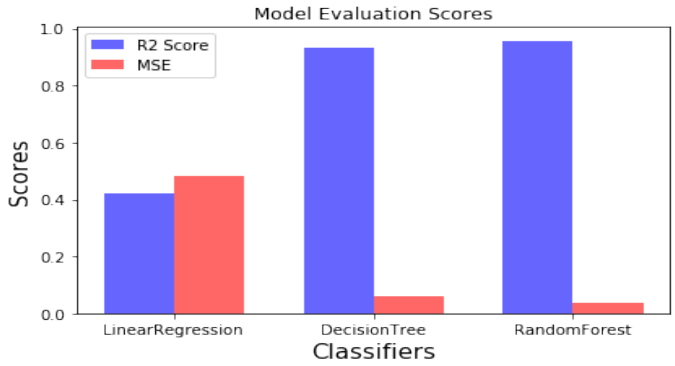
From Features graph, the features "is_snow", "is_rain" and "item_nbr" are important features to consider. Initially, I had only "is_rain" and "is_snow" – this feature selection was giving an R2-score of 0.49, MSE of 0.37 and accuracy of 78%. Once I added features: "item_nbr" and "station_nbr" the R2-score jumped to 0.94, MSE dropped to 0.03 and accuracy increased to 91%.

Here is the final list of features for Regression:

```
"month","weekday","tavg","depart","station_nbr","store_nbr","ite
m_nbr","preciptotal","snowfall","is_hot","is_cold","is_snow","is
_high_snow","is_rain","is_high_precip"
```

Model Selection

Fig 3:



As you can see clearly that Random Forest Regressor performed best with our initial training and validation data. Further, we used GridSearchCV to tune its parameters. This tuning took around 79 minutes.

Algorithms and Techniques:

The following tools were used to build this model.

- Anaconda3-5.2, with Python 3 on Jupyter notebook.
- Scikit-learn, Numpy, Pandas tools for data preprocessing.
- Matplotlib and seaborn for data visualization.
- Scikit-learn ensemble algorithm – Random Forest Regressor.
- Scikit-learn Decision Tree Regressor.

This project uses supervised learning of Regression to build models on training data set and validate it's accuracy with R2-Score, MSE and MAE on validation data. Since there is no validation data provided, we will split training data with 80/20 rule. A separate test data file contains stores, dates and item for which we need to predict "units" sold.

Initially, simple Liner Regression model was used, but it didn't perform well so it was decided to use ensemble methods algorithms. Ensemble method combines predictions of several base estimators to improve its performance over choosing a single one.

Random Forest Regressor is an average ensemble algorithm based on randomized decision trees. The prediction is usually better as it's given as the average prediction, rather than prediction of individual classifiers. The RandomSearchCV is used to further tune it's hyper parameters. Listed below are the parameters set values for tuning. It took approximately 79 minutes for tuning:

```
'bootstrap': [True, False]
'max_depth': [10, 20, 30, 40, 50, None],
'max_features': ['auto', 'sqrt'],
'min_samples_leaf': [1, 2, 4],
'min_samples_split': [2, 5, 10],
'n_estimators': [20,50,100]
```

The final values parameters for Random Forest after tuning has been completed:

```
bootstrap= False,
criterion= 'mse',
max_depth= 10,
max_features= 'auto',
max_leaf_nodes= None,
min_impurity_decrease=0.0,
min_impurity_split= None,
min_samples_leaf=4,
min_samples_split=2,
min_weight_fraction_leaf=0.0,
n_estimators= 20,
n_jobs= 1,
oob_score= False,
random_state= None,
verbose= 0,
warm_start= False
```

Final parameter for Decision Tree Regressor

```
max_depth=10
```

Benchmark

This project used 80/20 rule to split training data into training and validation data since no validation data was provided. Table below shows list of algorithms and it's results.

Algorithm	R2 Score	MSE	Training (Sec)	Accuracy
Linear Regression	0.39	0.62	10	0.23%
Decision Tree	0.69	0.43	8	78.62%
Random Forest	0.59	0.25	15	89.22%
Added New Features : Weekday, Item and Year				
Decision Tree	0.92	0.05	5	89.03%
Random Forest	0.95	0.03	15	90.48%

Initial benchmark was done on base model of Linear Regression, Decision Tree and Random Forest Regressor. Later, to Improve accuracy and R2 score four new features were added. Finally, Random Forest Regressor was chosen as it performed best among three.

Random Forest Regressor hyper parameter was further tuned with the help of Random Grid Search. This improved accuracy of 1% to 91%.

Methodology

Data Preprocessing

The “Preprocessing Data” section of notebook consists of following steps:

- Load weather and key data then join these datasets based on Station number. Remove duplicates if any.
- Load Train and Test data; and concatenate them horizontally so all the preprocessing can be done together. Tag the test rows with “Test” and train data rows as “Train”. This will also help to identify any missing features between train and test data.
- Join data created in Step 1 and Step 2 on dates and Store number.
- Convert values like “M”, “T” or “Nan” in feature class to appropriate values. A function is defined in notebook for this purpose.
- Normalize “Units” with natural log function. As there are some values which are high(284) and some are 0, this will bring values close together.
- Use Scikit-learn MinMaxScaler on following features: `'month','weekday','tavg","depart","preciptotal",and "snowfall".`
- Create necessary functions to convert non-numeric dataset into numeric dataset. Please see the “Modules for Features” section in notebook.
- Take a random sample of 50 records from original dataset and delete this sample from training set. This will be later use for model evaluation.
- Split training data into train and validation data using 80/20 rules, with the help of Scikit-learn library.

Refinement

The Random Forest Regressor with training data provides accuracy of around 91%. Initially the model was trained on the features provided with training data and this was giving accuracy of around 89%, mentioned in benchmarking section. After adding more features, like “item_nbr”, “store_nbr” and derived features like “Month” and “Weekend” the accuracy jumped to 91%.

Hyper-parameters tuning for Random Forest Regressors was done with Random Grid Search function. This was tried with GridSearchCV first but it took almost 2 hours before it returned any tuning results .

Results

Model Evaluation and Validation

There is no validation data provided so 80/20 rule was applied on training data to create train and validation data. The following metrics were used for model evaluation.

- R2-score
- Mean Square Error
- Accuracy function

```
{'LinearRegression': {'R2_score': 0.42192949615477515,
  'MSE': 0.4835229075976055,
   'Accuracy': 0.0},

'DecisionTreeRegressor': {'R2_score': 0.9295966896474112,
  'MSE': 0.0588883416464626,
  'Accuracy': 89.57079895622172},

 'RandomForestRegressor': {'R2_score': 0.9561344097474908,
  'MSE': 0.03669105688889691,
  'Accuracy': 91.64555939309462}}
```

The Random Forest Regressor was chosen as it performed the best.

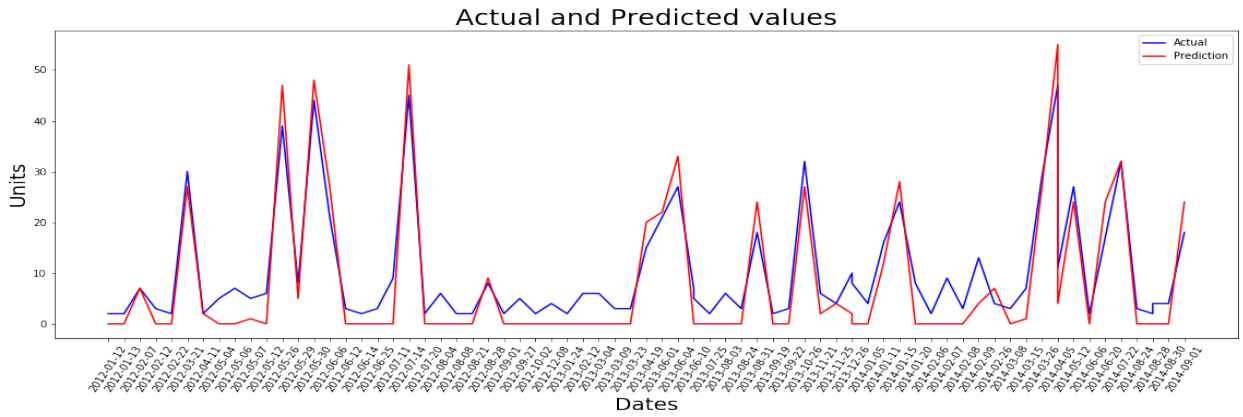
Justification:

While looking into Kaggle competition for the Capstone project I found this to be like a real-world problem. Event such as Weather, Marketing campaign or any Sporting event can impact sales of products within surrounding stores. This can help department stores to predict and maintain inventory of items in case of such events.

Conclusion:

Free-Form Visualization:

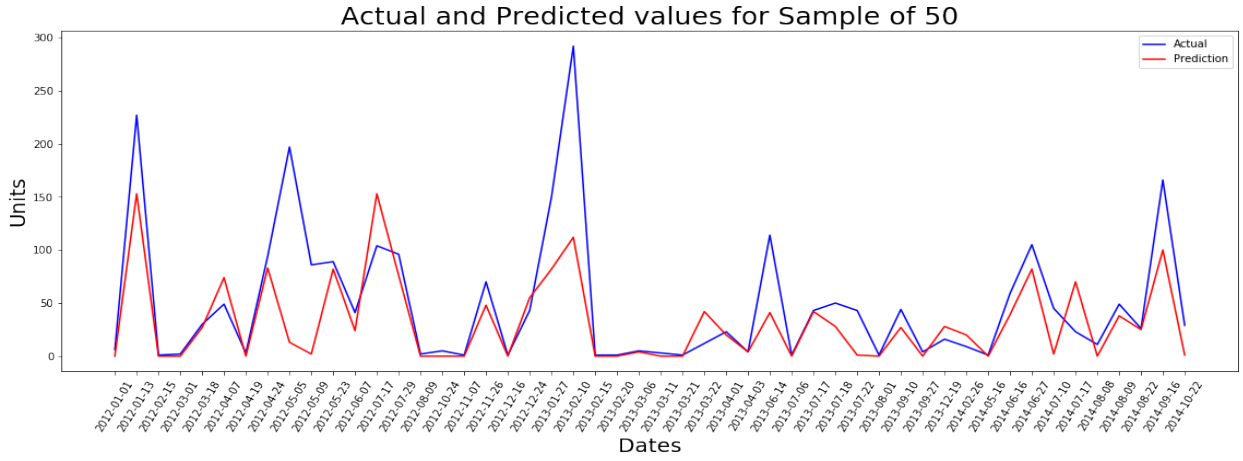
Fig 4



The above chart displays Validation data results on around 75 random Items. There are few exceptions with actual and predicted values.

As mentioned previously in data processing steps, we made the prediction on Sample of 50 records that were drawn initially. This sample was not used for training or validation. Below figure shows the final results with actual and predicted data

Fig 5:



The test file contains around 57k Items for prediction and the final results submitted in submissions.csv file.

This is what it looks like:

	id	units
0	2013-04-01_2_1	0.0
1	2013-04-01_2_2	0.0
2	2013-04-01_2_3	0.0
3	2013-04-01_2_4	0.0
4	2013-04-01_2_5	4.0

Reflection

I live in an area where there are numerous snow events each year. This project will allow us to apply learning to other areas as well.

Following are the project implementation process steps:

- Load, join and preprocess data.
- Select features and identify features that are important.
- Normalized features.
- Select Regression algorithms and tune hyper parameters.
- Model evaluation based on metrics
- Select the best performed model for final prediction.

Improvement:

Key areas for improvement:

- Provided data for the prediction requirements was very limited with few features.
- Half of the training data have unit value of 0.
- A lot of weather data was missing or incorrect. Replacing these values ('M','T') with 0 or some other value can create bias towards that feature.
- Some of the Item sales can be impacted by holidays like "Black Friday" or any other sale events in certain locations.
- Adding events like holidays will definitely help.
- Applying other regression technique(s) might improve accuracy.

- Some features in training data had no impact on model results. Some derived features can be added to improve accuracy. Such as holidays, current event, local festivals etc.
- Some bigger store might have high sale then smaller stores. No features were provided with data to create equivalency.

Bibliography:

- Tuning Hyper-parameters - https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74

- Scikit-learn GridSearchCV - https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

- Pandas Dataframe documentation - https://pandas.pydata.org/pandas-docs/version/0.17.0

- Random Forest Regressor - https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

- Walmart Kaggle Competation - https://www.kaggle.com/c/walmart-recruiting-sales-in-stormy-weather/data