Machine Learning Engineer Nanodegree

Capstone Project

Walmart Store Sales Forecasting

Nitin Pai

15th Nov 2019

1. Definition

A. Project Overview

Walmart is an American multinational retail corporation that operates a chain of hypermarkets, department stores and grocery stores. As of Jul 2019, Walmart has 11,200 stores in 27 countries with revenues exceeding $500 billion. A challenge facing the retail industry such as Walmart's is to ensure the supply chain and warehouse space usage is optimized to ensure supply meets demand effectively, especially during spikes such as the holiday seasons.

This is where accurate sales forecasting enable companies to make informed business decisions. Companies can base their forecasts on past sales data, industry-wide comparisons and economic trends. However, a forecasting challenge is the need to make decisions based on limited history. If Christmas comes but once a year, so does the chance to see how strategic decisions impacted the bottom line.

B. Problem Statement

Historical sales data for 45 Walmart stores located in different regions has been provided. Each store contains many departments, and the sales for each department in each store needs to be projected. Additionally, Walmart runs several promotional markdown events throughout the year. These markdowns precede prominent holidays, the four largest of which are the Super Bowl, Labor Day, Thanksgiving, and Christmas. The weeks including these holidays are weighted five times higher in the evaluation than non-holiday weeks. These markdowns are known to affect sales, but it is challenging to predict which departments are affected and the extent of the impact.

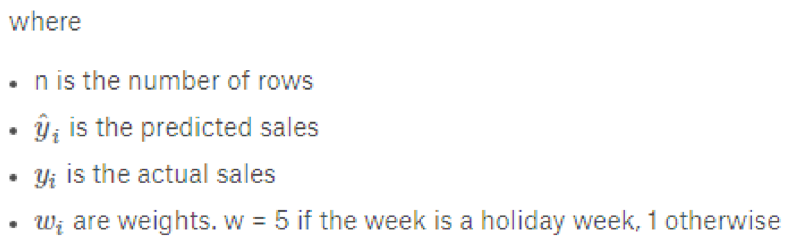
This problem is sourced from a Kaggle Competition and the data source is available here.

Predicting output values (in this case sales) is a regression problem and machine learning can be applied for this problem (Citation #1, #2 & #3). Also, as the dataset is labelled, Supervised Learning can be leveraged.

C. Metrics

Model prediction for this problem can be evaluated in several ways. However, since Kaggle's evaluation is based on weighted mean absolute error (WMAE), same will be leveraged here:

$$\mathrm{WMAE} = \frac{1}{\sum w_i} \sum_{i=1}^{n} w_i |y_i - \hat{y}_i|$$



where

- n is the number of rows
- $\hat{y}_i$ is the predicted sales
- y_i is the actual sales
- w_i are weights. w = 5 if the week is a holiday week, 1 otherwise

2. Analysis

A. Data Exploration

The datasets are provided by Walmart on Kaggle's website (dataset URL) and includes 4 CSV files:

File Details

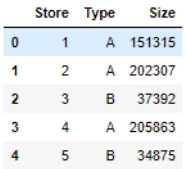
#	File Name	Description	Row Count	File Size
1	stores.csv	Contains anonymized information about the 45 stores, indicating the type and size of store.	45	1 KB
2	features.csv	Contains additional data related to the store, department, and regional activity for the given dates	8,191	579 KB
3	train.csv	This is the historical training data, which covers to 2010-02-05 to 2012-11-01	422,000	12,542 KB
4	test.csv	Identical to train.csv, except weekly sales data is withheld.	115,000	2,538 KB

Feature Details

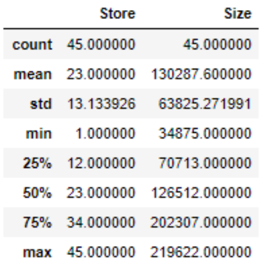
#	File Name	Feature Name	Description	Type
1	stores.csv	Store	Store Number	Integer
2	stores.csv	Type	Store Type	String (Categorical)
3	stores.csv	Size	Store Size	Integer
4	features.csv	Store	Store Number	Integer
5	features.csv	Date	The Week	Date (YYYY-MM-DD)
6	features.csv	Temperature	Average Temperature	Float
7	features.csv	Fuel_Price	Cost of Fuel	Float
8	features.csv	MarkDown1	Promotional Markdown	Float
9	features.csv	MarkDown2	Promotional Markdown	Float
10	features.csv	MarkDown3	Promotional Markdown	Float
11	features.csv	MarkDown4	Promotional Markdown	Float
12	features.csv	MarkDown5	Promotional Markdown	Float
13	features.csv	CPI	Consumer Price Index	Float
14	features.csv	Unemployment	Unemployment Rate	Float
15	features.csv	IsHoliday	Is Holiday Week (Y/N)	Boolean (Categorical)
16	train.csv / test.csv	Store	Store Number	Integer
17	train.csv / test.csv	Dept	Department Number	Integer
18	train.csv / test.csv	Date	The Week	Date (YYYY-MM-DD)
19	train.csv / test.csv	IsHoliday	Is Holiday Week (Y/N)	Boolean (Categorical)
20	train.csv	Weekly_Sales	Weekly Sales	Float

Data Sampling & Statistics:

stores.csv:

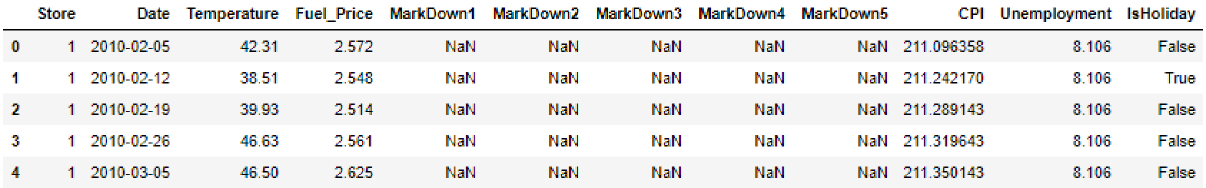


	Store	Type	Size
0	1	A	151315
1	2	A	202307
2	3	B	37392
3	4	A	205863
4	5	B	34875

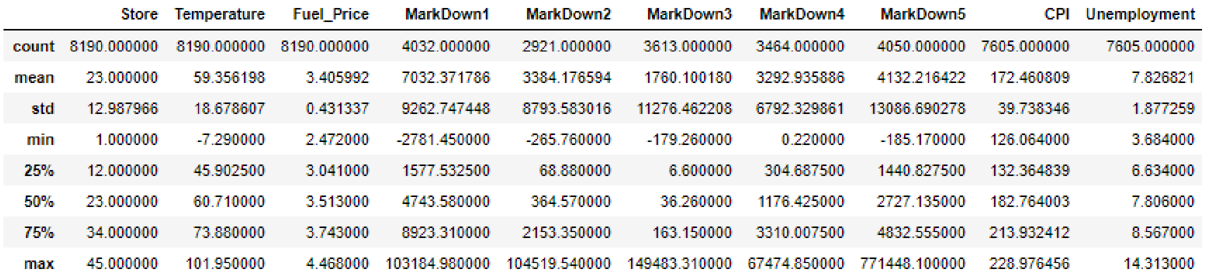


	Store	Size
count	45.000000	45.000000
mean	23.000000	130287.600000
std	13.133926	63825.271991
min	1.000000	34875.000000
25%	12.000000	70713.000000
50%	23.000000	126512.000000
75%	34.000000	202307.000000
max	45.000000	219622.000000

features.csv:

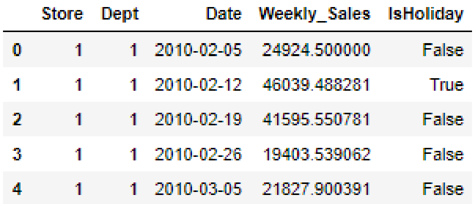


	Store	Date	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment	IsHoliday
0	1	2010-02-05	42.31	2.572	NaN	NaN	NaN	NaN	NaN	211.096358	8.106	False
1	1	2010-02-12	38.51	2.548	NaN	NaN	NaN	NaN	NaN	211.242170	8.106	True
2	1	2010-02-19	39.93	2.514	NaN	NaN	NaN	NaN	NaN	211.289143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

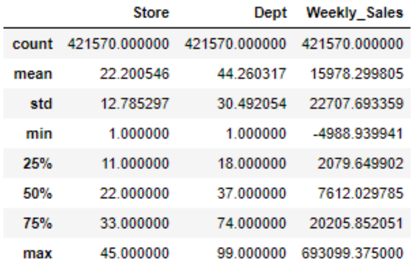


	Store	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment
count	8190.000000	8190.000000	8190.000000	4032.000000	2921.000000	3613.000000	3464.000000	4050.000000	7605.000000	7605.000000
mean	23.000000	59.356198	3.405992	7032.371786	3384.176594	1760.100180	3292.935886	4132.216422	172.460809	7.826821
std	12.987966	18.678607	0.431337	9262.747448	8793.583016	11276.462208	6792.329861	13086.690278	39.738346	1.877259
min	1.000000	-7.290000	2.472000	-2781.450000	-265.760000	-179.260000	0.220000	-185.170000	126.064000	3.684000
25%	12.000000	45.902500	3.041000	1577.532500	68.880000	6.600000	304.687500	1440.827500	132.364839	6.634000
50%	23.000000	60.710000	3.513000	4743.580000	364.570000	36.260000	1176.425000	2727.135000	182.764003	7.806000
75%	34.000000	73.880000	3.743000	8923.310000	2153.350000	163.150000	3310.007500	4832.555000	213.932412	8.567000
max	45.000000	101.950000	4.468000	103184.980000	104519.540000	149483.310000	67474.850000	771448.100000	228.976456	14.313000

train.csv:

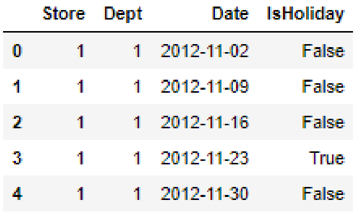


	Store	Dept	Date	Weekly_Sales	IsHoliday
0	1	1	2010-02-05	24924.500000	False
1	1	1	2010-02-12	46039.488281	True
2	1	1	2010-02-19	41595.550781	False
3	1	1	2010-02-26	19403.539062	False
4	1	1	2010-03-05	21827.900391	False

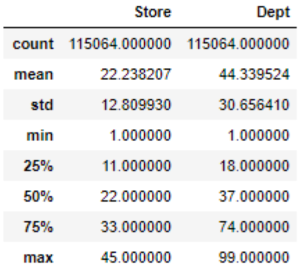


	Store	Dept	Weekly_Sales
count	421570.000000	421570.000000	421570.000000
mean	22.200546	44.260317	15978.299805
std	12.785297	30.492054	22707.693359
min	1.000000	1.000000	-4988.939941
25%	11.000000	18.000000	2079.649902
50%	22.000000	37.000000	7612.029785
75%	33.000000	74.000000	20205.852051
max	45.000000	99.000000	693099.375000

test.csv:



	Store	Dept	Date	IsHoliday
0	1	1	2012-11-02	False
1	1	1	2012-11-09	False
2	1	1	2012-11-16	False
3	1	1	2012-11-23	True
4	1	1	2012-11-30	False



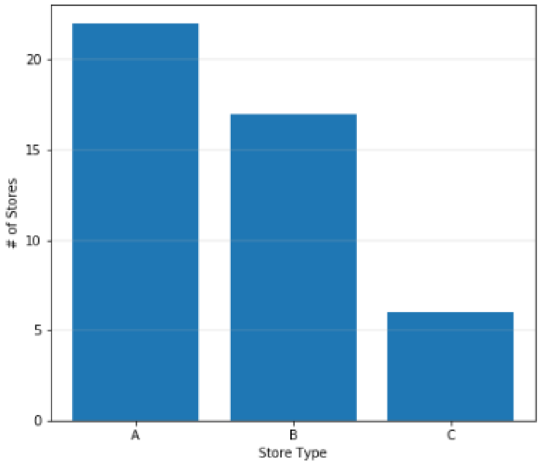
	Store	Dept
count	115064.000000	115064.000000
mean	22.238207	44.339524
std	12.809930	30.656410
min	1.000000	1.000000
25%	11.000000	18.000000
50%	22.000000	37.000000
75%	33.000000	74.000000
max	45.000000	99.000000

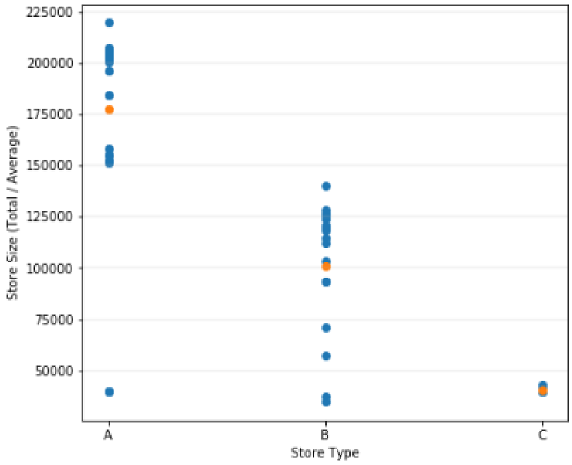
Observations

1. Date Range of Dataset
    a. Training: 2010 to 2012
    b. Test: 2012 to 2013
    c. Feature: 2010 to 2013
2. Features has missing values for columns Unemployment, CPI, MarkDown1, MarkDown2, MarkDown3, MarkDown4 and MarkDown5.
3. Stores has a categorical column Type.
4. Train and Test have a string column Date (format: YYYY-MM-DD) and a Boolean column IsHoliday.

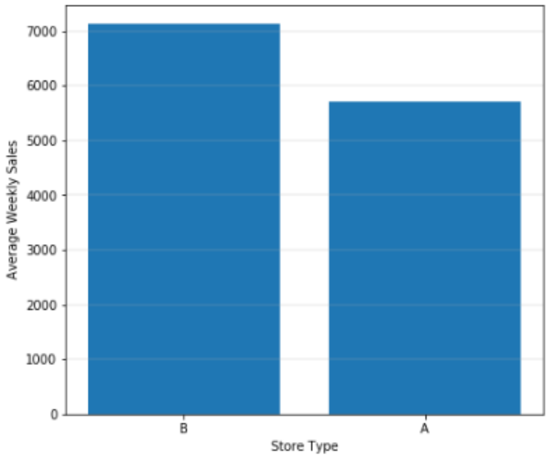
B. Exploratory Visualization

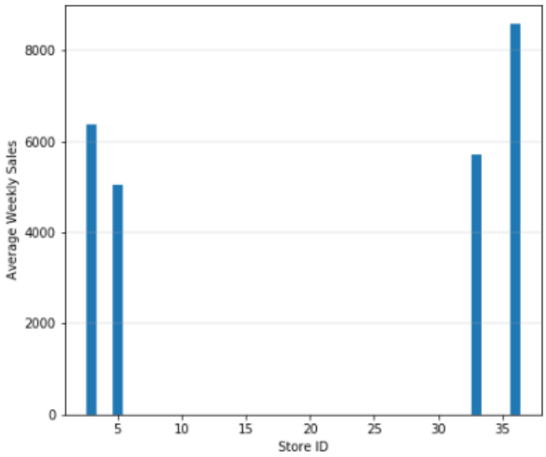
- **Stores Data**
  - Stores are classified into 3 types – A, B & C.
  - Most stores are Type A, followed by Type B and then Type C.
  - Store Size seems to be linked to Store Type. Type A have the largest average size (~ 175K), followed by Type B (~100K) and Type C (~40K). Type A and B seems to have a few outliers with store sizes way below the average.



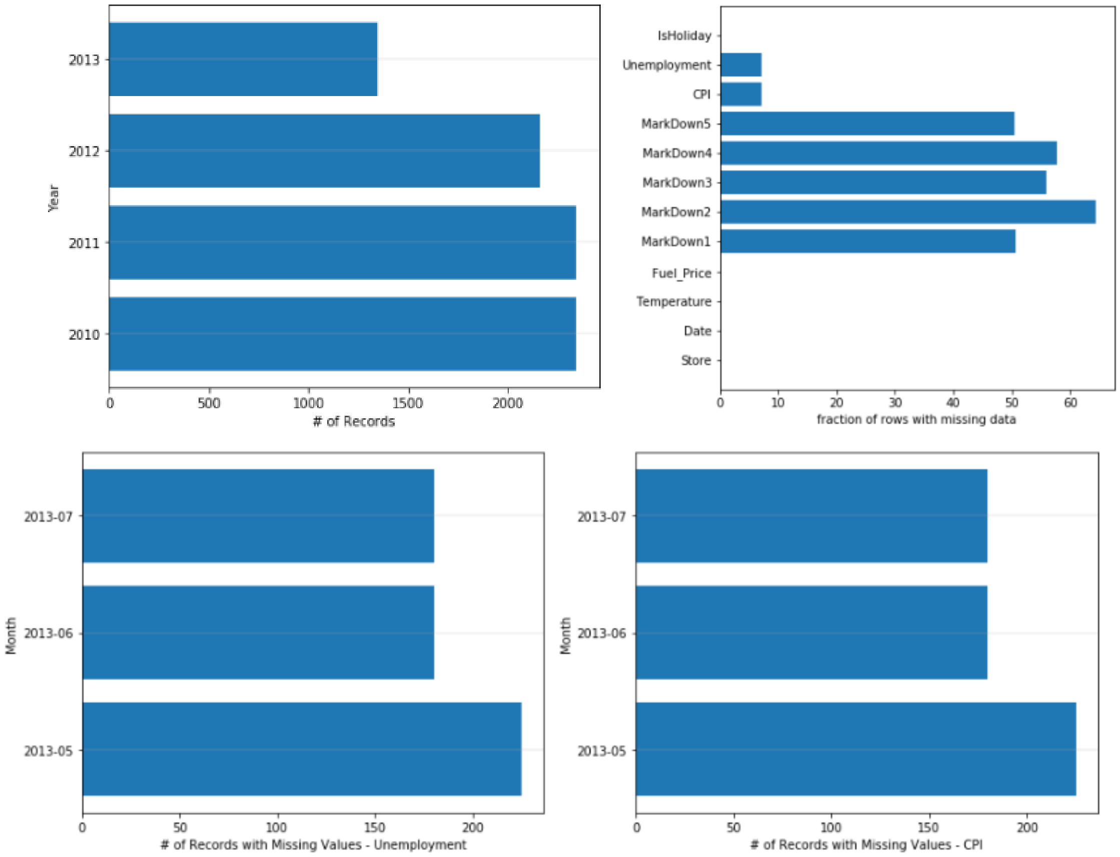


  - After merging Stores with Train and plotting Average Weekly Sales by Store, we see that Average Weekly Sales also seems to be linked to Store Type. Store Size outliers are also outliers when it comes to Average Weekly Sales. So, it seems like these outliers have been incorrectly classified as Types A & B and would need to be reclassified as Type C.

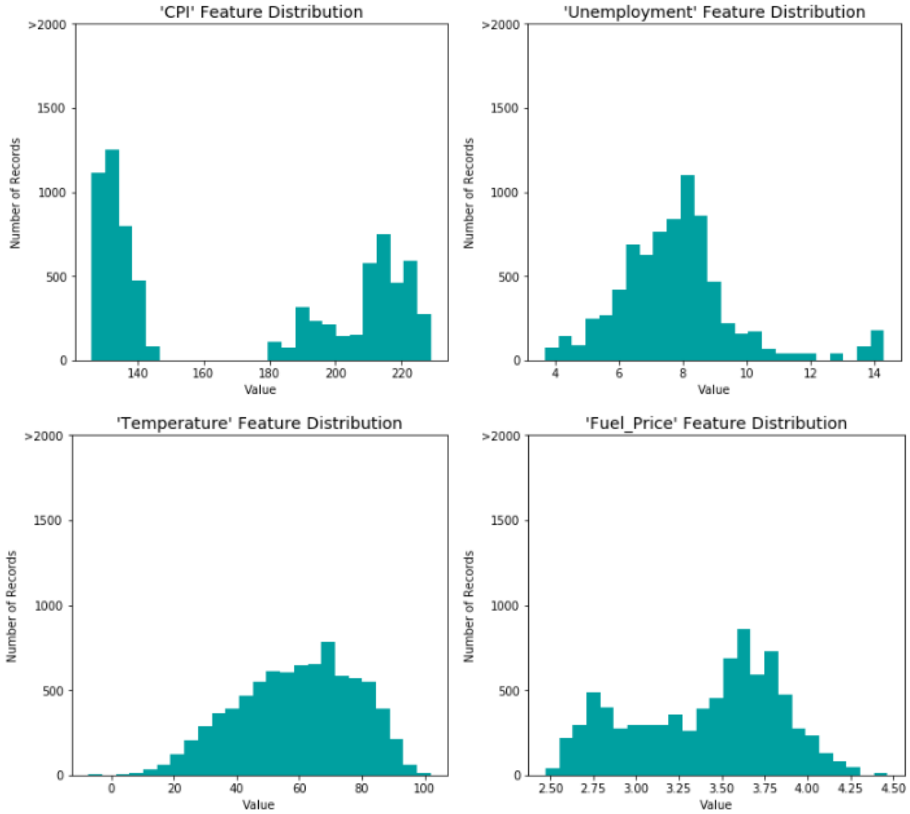




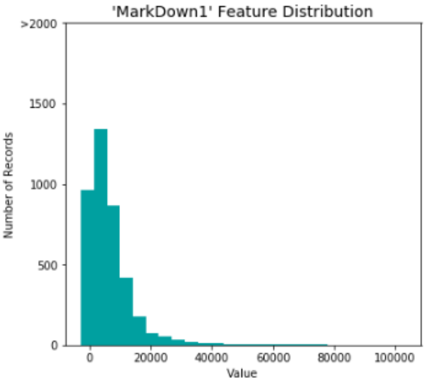
- **Features Data**
  - Data is distributed across 4 years – 2010, 2011, 2012 and 2013. Data for 2013 is only until July.
  - A number of columns have missing values. These include columns Unemployment, CPI & MarkDowns (MarkDown1, MarkDown2, Markdown3, MarkDown4 and MarkDown5).
  - For Unemployment and CPI, fraction of rows with missing data is ~10%. For all stores these columns are missing values for the months May, June & July 2013. As per the available data, values for these columns does not change significantly across months. This being the case, data from Apr 2013 would be propagated to months with missing data.
  - For the MarkDowns, the data is missing for the whole of 2010 and until Nov 2011 – as mentioned in the problem description.

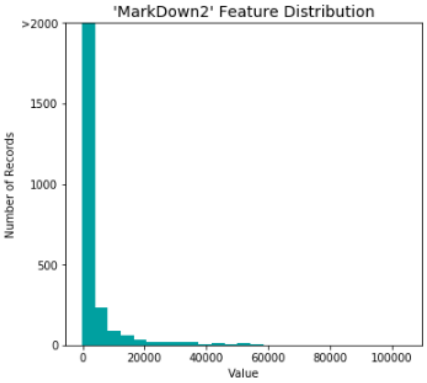


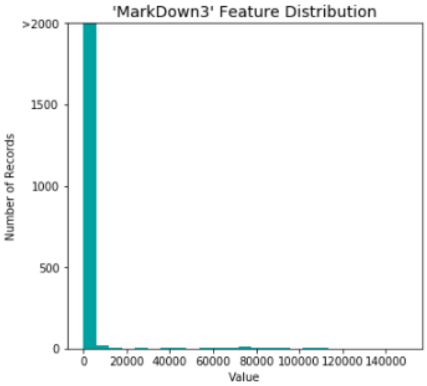
- CPI and Unemployment are a little skewed. Temperature and Fuel Price are not skewed.

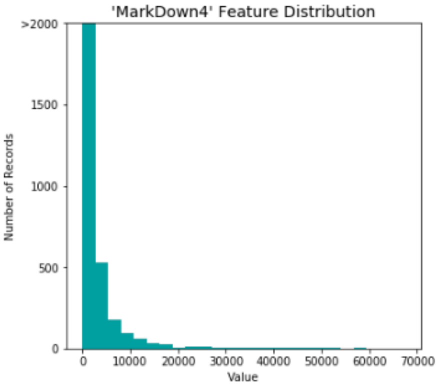


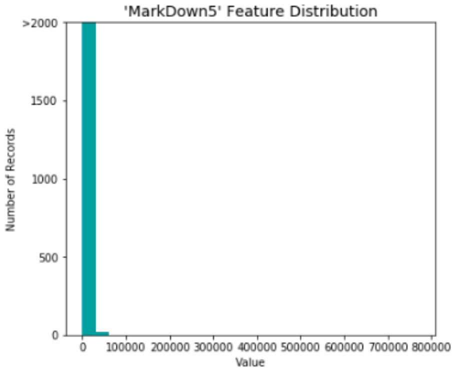
- MarkDowns are skewed and would need to be transformed.



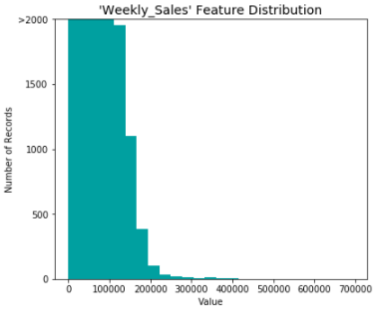








- **Train Data**
  - Values for Weekly Sales seems skewed.



C. Algorithms and Techniques

The most basic regression algorithm is Linear Regression. If a linear model can explain the data well, there is no need for further complexity. Regularization techniques can be applied to penalize the coefficient values of the features, since higher values generally tend towards overfitting and loss of generalization. Regularization techniques enhance performance of linear models greatly.

In the case of regularization, there are two kinds: L1 [adds absolute values of coefficients to loss function] and L2 [adds squares of coefficients to loss function]. Elastic Net combines the penalties (L1 and L2) to get the best of both worlds.

Linear Models:

- Linear Regression
- Elastic Net Regression

The next category of algorithms is of the Tree based Regression models. An advantage of tree based models is that they are robust to outliers compared to linear models. Given the number of features, it is fairly likely that Decision Tree will over fit the data. Hence, it has been skipped and ensemble methods listed below have been picked. Ensemble methods include building multiple Regressors on copies of same training data and combining their output either through mean, median, Bagging (growing trees sequentially) and Boosting (using weighted average of weak learners).

Random Forests is one of the primary Bagging methods and works well on high dimensional data. Gradient Boosting Machine, Light GBM and XGBoost are types of Boosting methods. These builds additive models in a way that performance always increases.

Tree Models:

- Random Forest
- Gradient Boosting Machines (GBM)
- Light Gradient Boosting Machines (Light GBM)
- Extreme Gradient Boosting (XGBoost)

Finally, one of the popular algorithms for non-linear problems are neural networks. Neural networks work great when there is a complex non-linear relationship between the inputs and the output. Although they generally have superior performance, one of their downside is that they take very long time to train. I will use Multi-layer Perceptron as my choice of neural network. The error function is Mean Absolute Error (L1 Loss).

Neural Networks:

- Multi-layer Perceptron (PyTorch)

D. Benchmark

Benchmark model is Linear Regression on the scaled data.

Observation:

- WMAE on validation data: 14774
- Time taken to Train: 0.224 secs (for 337256 records)
- Time taken to Predict: 0.003 secs (for 84314 records)

3. Methodology

A. Data Pre-Processing

The provided dataset requires some pre-processing before it can be fed into a machine learning model. This includes:

1. Correct Values

Stores file has four stores which have been incorrectly categorized as Types A & B.
Column: Type
The size and average weekly sales of these four stores are in line with Type C stores. So, the Type of these stores have been changed to C.

Features file has negative values for MarkDown columns.
Columns: MarkDown1, MarkDown2, MarkDown3, MarkDown4 and MarkDown5
As per the definition, MarkDowns are discounts provided from time to time by the store. A negative discount seems to be invalid. So, negative valued MarkDowns are being set to 0.

Train file has negative values for Sales column.
Column: Weekly Sales
Weekly Sales is the target variable. There are 1200+ records with a negative sales value. A negative sales value seems invalid. So, negative valued sales are being set to 0.

2. Missing Values

Features file has a number of columns with missing values.
Columns: CPI & Unemployment
These columns are missing values for 3 months – May, Jun & Jul 2013. As values for these columns does not change significantly month on month, values from Apr 2019 would be propagated to records with missing values.

Columns: MarkDown1, MarkDown2, MarkDown3, MarkDown4 & MarkDown5
These columns are missing values for 2010 (entire year) and 2011 (up to Nov). As values for these columns seem to be similar for similar times of the year, values from 2012 would be copied over to the corresponding weeks of 2010 and 2011.

3. Merge Datasets

Train & Test: Left merge these files with Stores on the column Store. Then, left merge the output of this with Features on the columns Store and Date.

4. Feature Engineering

Column #1: *IsHoliday* – This is a boolean values column and would need to be converted to numeric. Convert False → 0 and True → 1.

Column #2: *Type* – This is a categorical column with values A, B & C. This would need to be converted to numeric via one-hot encoding.

Column #3: *Week* – From column *Date*, derive and create a new column *Week*. As the data is at the weekly grain, this new numeric column can replace the Date column.

5. Log Transform Skewed Features

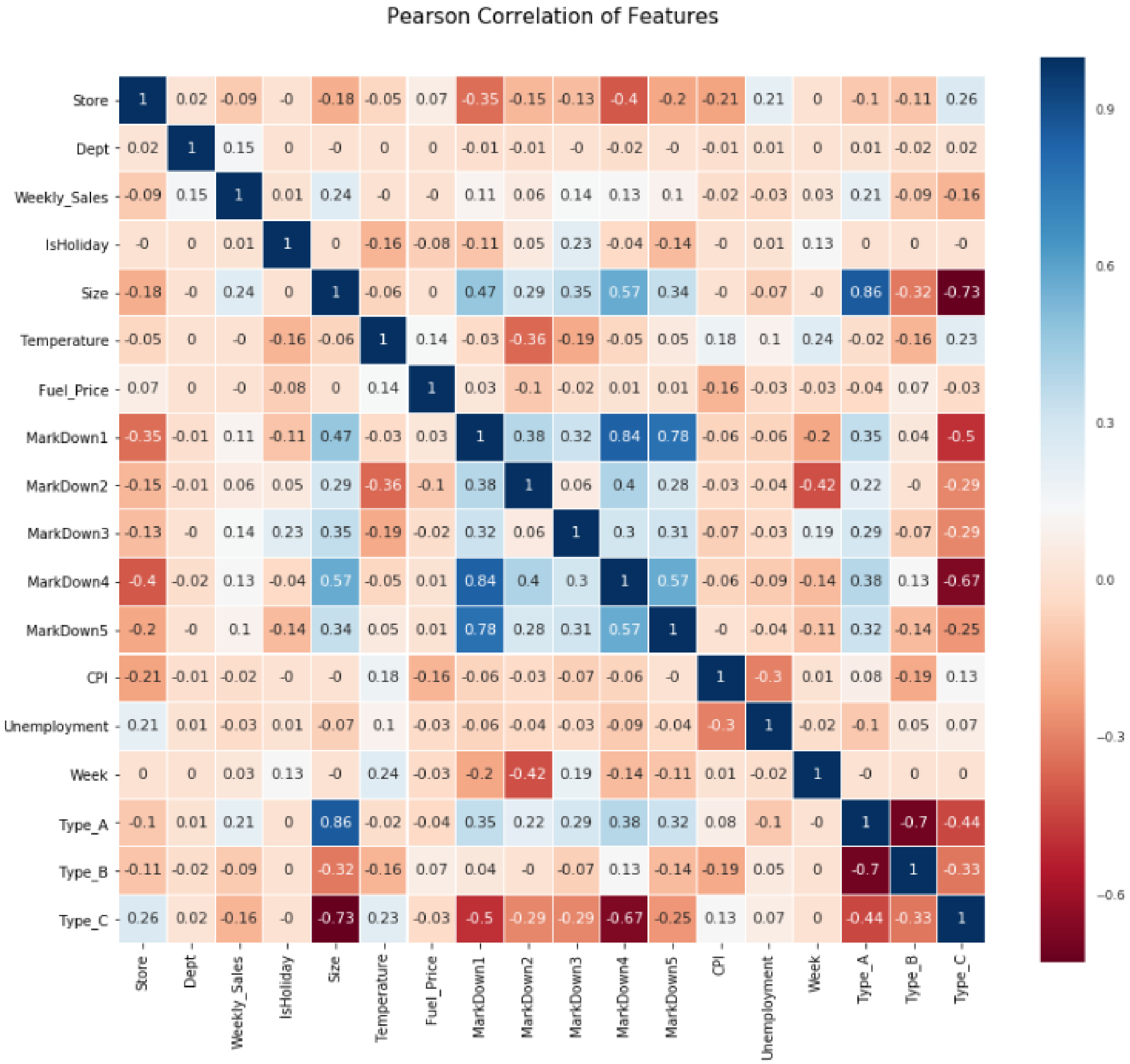
As the distribution of some numerical features is highly skewed, logarithmic transformation would need to be applied on these so that the very large and very small values do not negatively affect the performance of a learning algorithm. Skewed features include: CPI, Unemployment, MarkDown1, MarkDown2, MarkDown3, MarkDown4, MarkDown5 and Weekly Sales.

6. Scale Datasets

As the distribution of numerical values across columns is varied, the data needs to be scaled to ensure each column is given equal weightage by the model. To achieve this, the MinMaxScaler would be used to scale the numerical data between 0 and 1.

7. Feature Correlation

Based on Pearson Correlation of features, it was determined that the columns *MarkDown4* and *Type_A* are highly correlated (> 0.8) to other features and these could be dropped from the dataset.



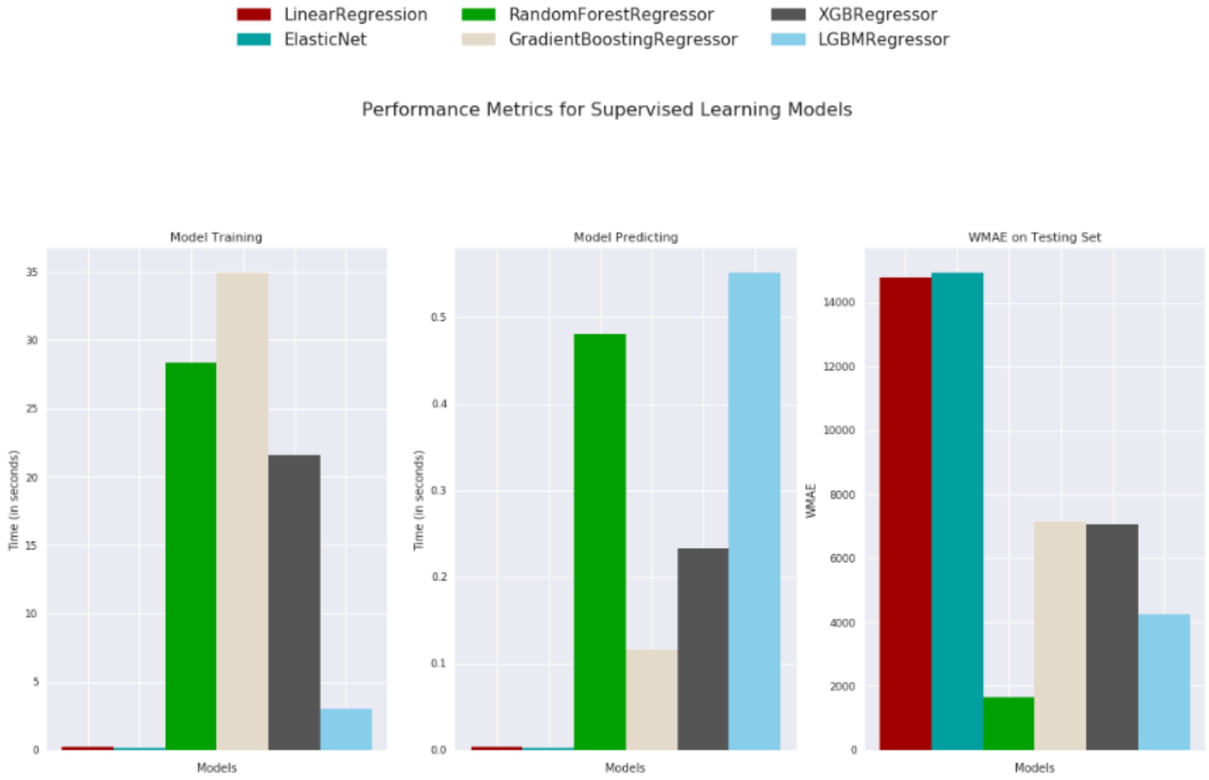
	Store	Dept	Weekly_Sales	IsHoliday	Size	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment	Week	Type_A	Type_B	Type_C
Store	1	0.02	-0.09	-0	-0.18	-0.05	0.07	-0.35	-0.15	-0.13	-0.4	-0.2	-0.21	0.21	0	-0.1	-0.11	0.26
Dept	0.02	1	0.15	0	-0	0	0	-0.01	-0.01	-0	-0.02	-0	-0.01	0.01	0	0.01	-0.02	0.02
Weekly_Sales	-0.09	0.15	1	0.01	0.24	-0	-0	0.11	0.06	0.14	0.13	0.1	-0.02	-0.03	0.03	0.21	-0.09	-0.16
IsHoliday	-0	0	0.01	1	0	-0.16	-0.08	-0.11	0.05	0.23	-0.04	-0.14	-0	0.01	0.13	0	0	-0
Size	-0.18	-0	0.24	0	1	-0.06	0	0.47	0.29	0.35	0.57	0.34	-0	-0.07	-0	0.86	-0.32	-0.73
Temperature	-0.05	0	-0	-0.16	-0.06	1	0.14	-0.03	-0.36	-0.19	-0.05	0.05	0.18	0.1	0.24	-0.02	-0.16	0.23
Fuel_Price	0.07	0	-0	-0.08	0	0.14	1	0.03	-0.1	-0.02	0.01	0.01	-0.16	-0.03	-0.03	-0.04	0.07	-0.03
MarkDown1	-0.35	-0.01	0.11	-0.11	0.47	-0.03	0.03	1	0.38	0.32	0.84	0.78	-0.06	-0.06	-0.2	0.35	0.04	-0.5
MarkDown2	-0.15	-0.01	0.06	0.05	0.29	-0.36	-0.1	0.38	1	0.06	0.4	0.28	-0.03	-0.04	-0.42	0.22	-0	-0.29
MarkDown3	-0.13	-0	0.14	0.23	0.35	-0.19	-0.02	0.32	0.06	1	0.3	0.31	-0.07	-0.03	0.19	0.29	-0.07	-0.29
MarkDown4	-0.4	-0.02	0.13	-0.04	0.57	-0.05	0.01	0.84	0.4	0.3	1	0.57	-0.06	-0.09	-0.14	0.38	0.13	-0.67
MarkDown5	-0.2	-0	0.1	-0.14	0.34	0.05	0.01	0.78	0.28	0.31	0.57	1	-0	-0.04	-0.11	0.32	-0.14	-0.25
CPI	-0.21	-0.01	-0.02	-0	-0	0.18	-0.16	-0.06	-0.03	-0.07	-0.06	-0	1	-0.3	0.01	0.08	-0.19	0.13
Unemployment	0.21	0.01	-0.03	0.01	-0.07	0.1	-0.03	-0.06	-0.04	-0.03	-0.09	-0.04	-0.3	1	-0.02	-0.1	0.05	0.07
Week	0	0	0.03	0.13	-0	0.24	-0.03	-0.2	-0.42	0.19	-0.14	-0.11	0.01	-0.02	1	-0	0	0
Type_A	-0.1	0.01	0.21	0	0.86	-0.02	-0.04	0.35	0.22	0.29	0.38	0.32	0.08	-0.1	-0	1	-0.7	-0.44
Type_B	-0.11	-0.02	-0.09	0	-0.32	-0.16	0.07	0.04	-0	-0.07	0.13	-0.14	-0.19	0.05	0	-0.7	1	-0.33
Type_C	0.26	0.02	-0.16	-0	-0.73	0.23	-0.03	-0.5	-0.29	-0.29	-0.67	-0.25	0.13	0.07	0	-0.44	-0.33	1

B. Implementation

The following approach was adopted:

1. Create an evaluation pipeline to execute base model of each regressor. Further, execution such as training time, predicting time and error would be evaluated via visualizations.
2. Create a training and predicting pipeline to execute tuned model of shortlisted regressor.
3. Pass each shortlisted Regressor to the training and predicting pipeline and consolidate the obtained metrics.
4. Stack/blend the predicted outputs of the two shortlisted models to reduce the error.

Base Regressor Evaluation:



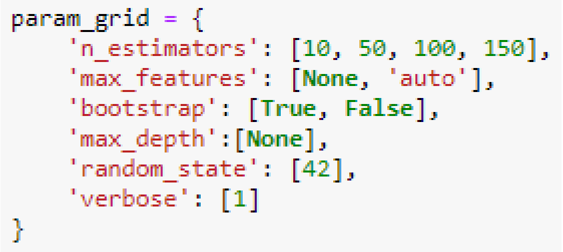
#	Model	Training Time	Prediction Time	WMAE on Validation
1	Linear Regression	0.21	0.01	14774
2	Elastic Net Regression	0.20	0.01	14939
3	Random Forest Regressor	28.95	0.50	1671
4	Gradient Boosting Regressor	37.34	0.12	7051
5	XGBoost Regressor	22.10	0.23	7063
6	Light GBM Regressor	3.17	0.54	4223

As observed from the results, Random Forest Regressor outperformed the others in terms on WMAE, although the training and predicting time is on the higher side. Light GBM Regressor comes in the second place on the WMAE metric.

C. Refinement

Top two models – Random Forest Regressor and Light GBM Regressor – from the evaluation were selected and further tuned via hyperparameters.

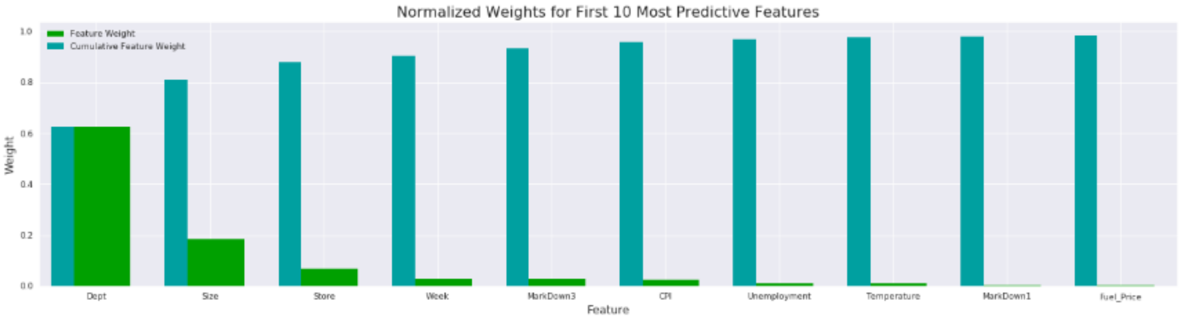
Random Forest: Hypermarameters listed below were chosen for tuning –



```
param_grid = {
    'n_estimators': [10, 50, 100, 150],
    'max_features': [None, 'auto'],
    'bootstrap': [True, False],
    'max_depth':[None],
    'random_state': [42],
    'verbose': [1]
}
```

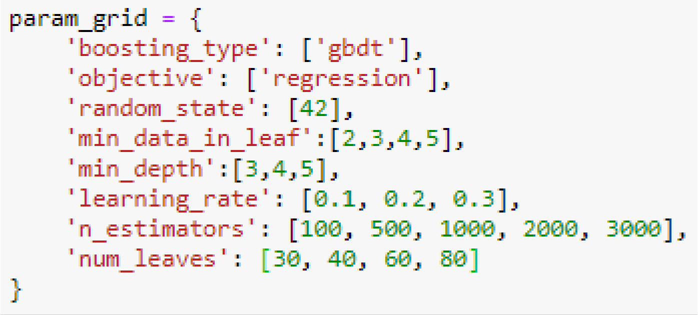
#	Hyperparameter	Description	Initial Default Value	Range	Final Selected Value
1	n_estimators	Number of trees in the forest	10	10,50, 100, 150	150
2	max_features	Number of features to consider when looking for the best split	'auto'	None, 'auto'	None
3	bootstrap	Whether bootstrap samples are used when building trees	True	True, False	True

Tuned Model's Feature Importance:



Before tuning, the validation set's WMAE was 1671. After tuning, it reduced to 1575. A performance gain of 5.7%.

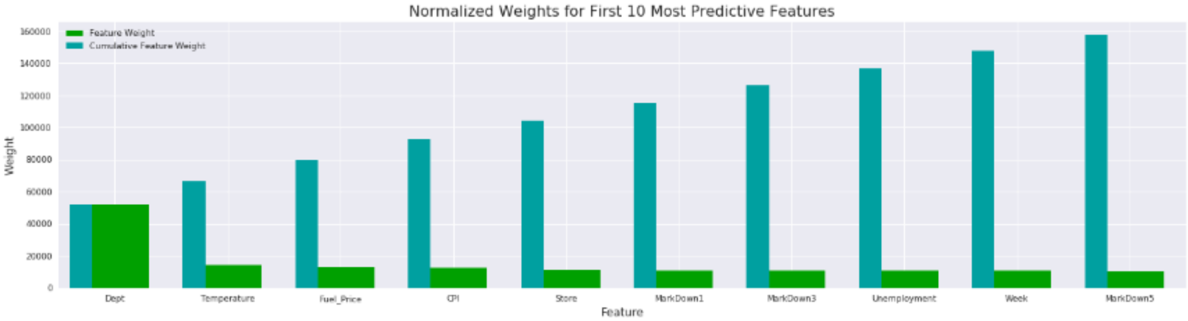
<u>Light GBM</u>: Hypermarameters listed below were chosen for tuning –



```
param_grid = {
    'boosting_type': ['gbdt'],
    'objective': ['regression'],
    'random_state': [42],
    'min_data_in_leaf':[2,3,4,5],
    'min_depth':[3,4,5],
    'learning_rate': [0.1, 0.2, 0.3],
    'n_estimators': [100, 500, 1000, 2000, 3000],
    'num_leaves': [30, 40, 60, 80]
}
```

#	Hyperparameter	Description	Initial Default Value	Range	Final Selected Value
1	n_estimators	Number of boosting iterations	100	100, 500, 1000, 2000, 3000	3000
2	min_data_in_leaf	Minimum number of data in one leaf	0	2, 3, 4, 5	2
3	min_depth	Minimum depth of tree	0	3, 4, 5	3
4	num_leaves	Max number of leaves in one tree	31	30, 40, 60, 80	80
5	Learning_rate	Shrinkage Rate	0.1	0.1, 0.2, 0.3	0.3

Tuned Model's Feature Importance:



Before tuning, the validation set's WMAE was 4223. After tuning, it reduced to 1479. A performance gain of 64%.

Additionally, as the two shortlisted models operate in different way as is evident from the Feature Importance of each, model stacking/blending was employed on the predictions of each model to arrive at the final prediction.

Each model's prediction was given a weightage and after some trial and error, a weightage of 0.8 for the Random Forest model and 0.2 for the Light GBM model provides the least WMEA of 1474.

```
pred_y = (pred_y_rf_test * 0.8) + (pred_y_lgbm_test * 0.2)
```

4. Results

A. Model Evaluation and Validation

During development, the validation data was used to evaluate the model. The final model architecture and hyperparameters were chosen because they performed the best among the tried combinations. This architecture is described in detail in Section 3.

Additionally, to verify the robustness of the final model, the test dataset (without the target variable) was processed through the model and predictions submitted to Kaggle. This had a WMAE of 3357 as compared to the leader board's score of 2301. This score however, still make the Top 40%, which is satisfactory.

B. Justification

The final model design with tuned hyperparameters trained on 80% of the training data scored WMAE of 1454 on the validation data, dwarfing the benchmark model's WMAE of 14774. This means the final model far surpasses the benchmark in terms of learning the target concept.

Based on the improvements records above, the final tuned model can be deemed as a satisfactory solution, although there is scope for improvement.