Store Sales Forecasting

The Problem

The problem is to predict weekly sales data based on historical sales data for 45 Walmart stores located in different regions.

If we are able to predict the weekly sales we can use this knowledge to better manage the supply chain and we can also see how each departments are affected by the markdown and the extent of the impact.

Dataset

Summary

The dataset is from the Kaggle competition Store Sales Forecasting. This file contains anonymized information about the 45 stores, indicating the type and size of store. The data contains 8191 rows with 12 features each row.

Major features are

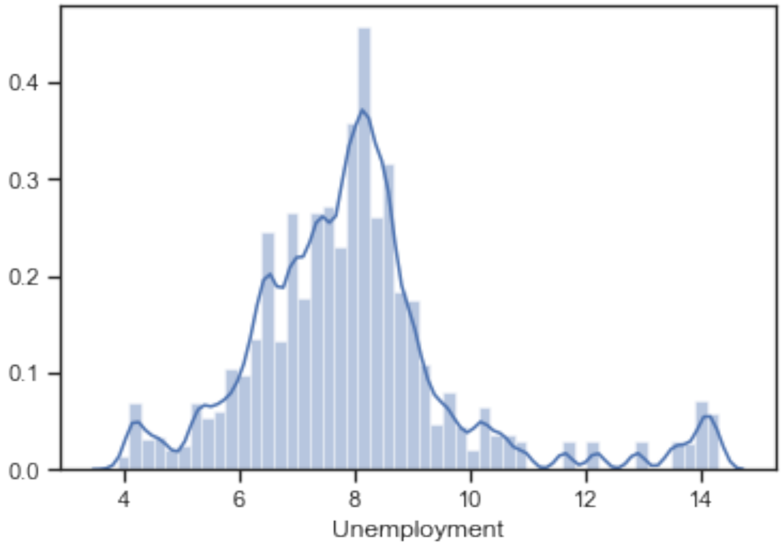
- Store – the store number
- Date – the week
- Temperature – average temperature in the region
- Fuel_Price – cost of fuel in the region
- MarkDown1–5 – anonymized data related to promotional markdowns that Walmart is running. MarkDown data is only available after Nov 2011, and is not available for all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

In addition, 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. Part of the challenge presented by this competition is modeling the effects of markdowns on these holiday weeks in the absence of complete/ideal historical data.

Data analysis

We perform some common techniques and found several patterns in the dataset.

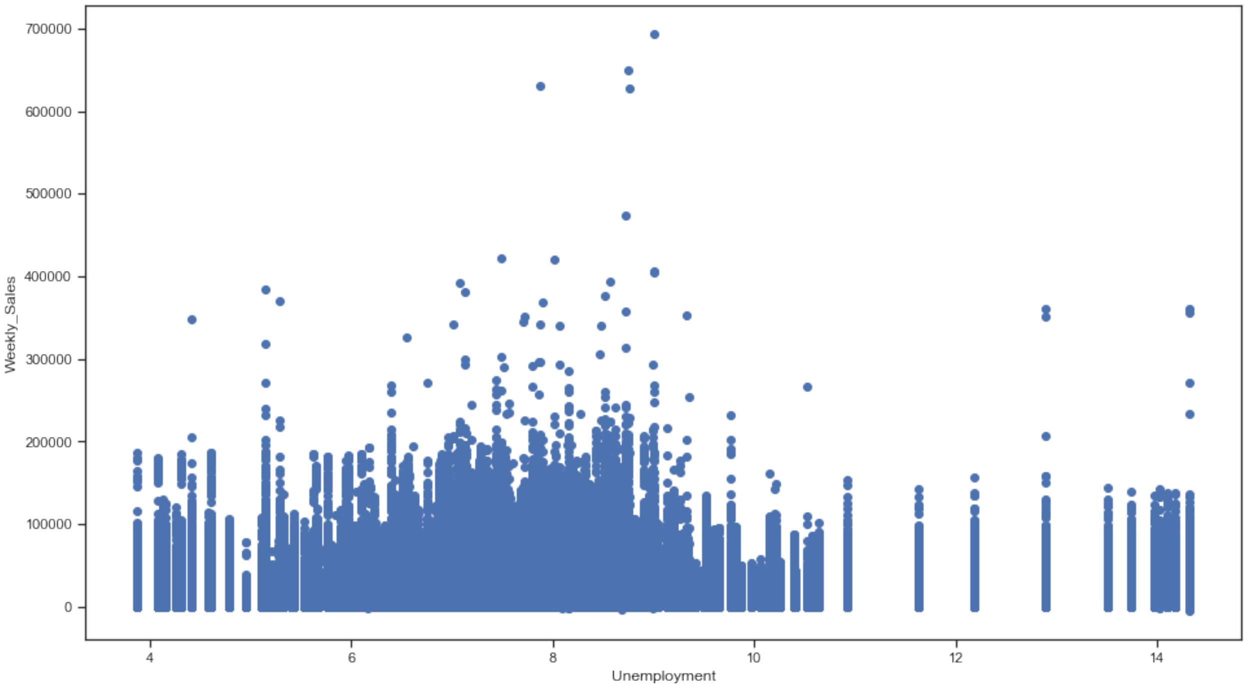
Firstly, we tried to observe the distribution of the dataset



We found the unemployment rate is also not balanced, the peak is at 8%

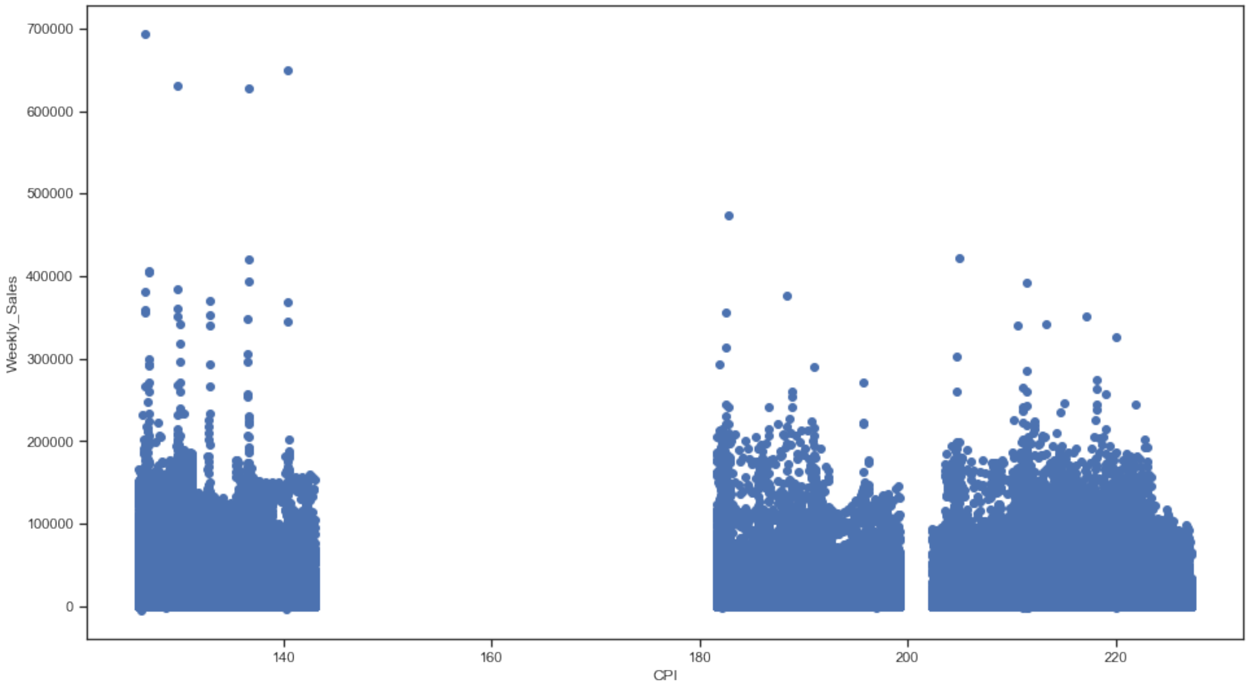
Also, the type of stores are unbalanced. there are more type A store than B and C combined

Then we explore the relations between different features and weekly sales

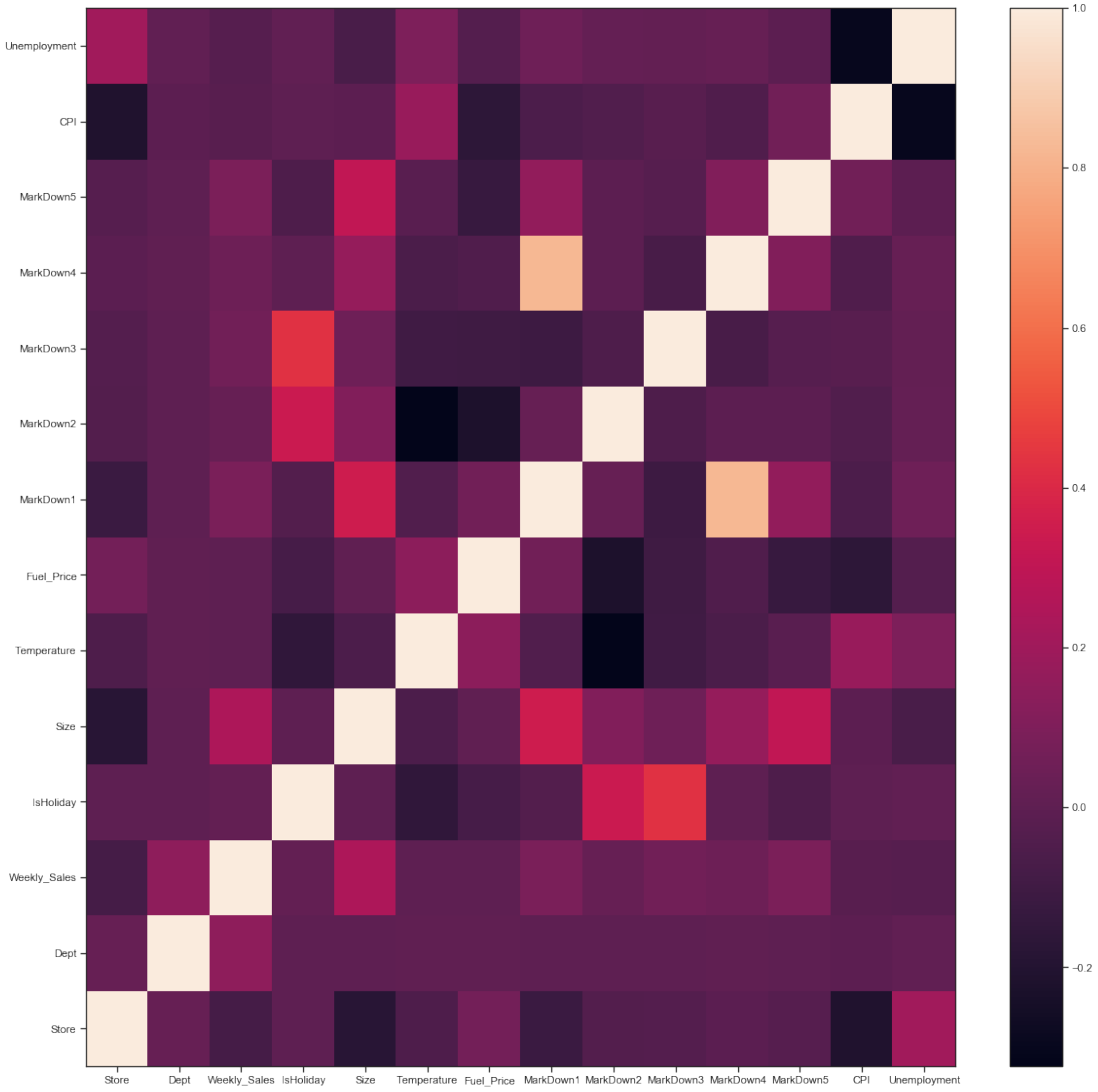


althought there are so many data points, we can still see that high sales are closely related to low unemployment rates.

and we also noticed that lower CPIs creates more sales. And there seems to be no data in the CPI range of [143, 181]



Finally we looked at the correlation coefficient between differen features



We particular focus on the correlation between weekly sales and other factors. It appears that the size of the store is very relavant to the weekly sales.

With all these findings, we tried several models to predict the sales.

(*We only include significant findings in this paragraph, other explorations can be found in the notebook* `data-analysis.ipynb`)

Methods

We use three methods to analyze this problem. They are KNN, Random Forest and Extra Trees.

Data preprocessing

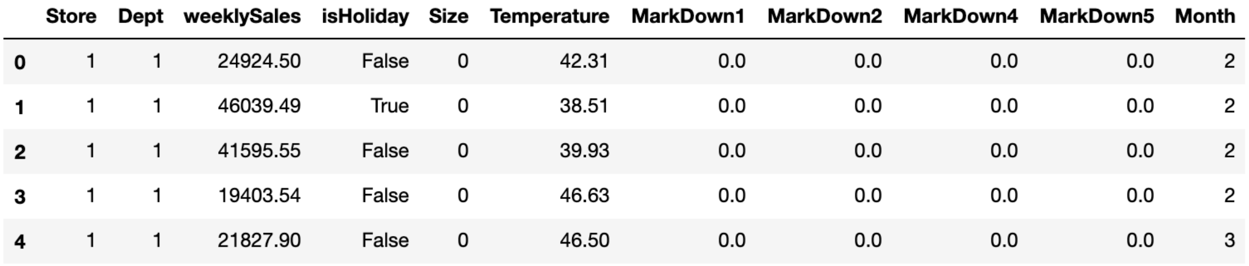
Reading the data:

Reading the data from train.csv, features.csv, stores.csv, test.csv.

Preprocessing

Drop “IsHoliday” feature from features.csv. Merge train.csv, features.csv and stores.csv as train dataset. Merge test.csv, features.csv and stores.csv as test dataset.

There are many null values for the field MarkDown(Because markdowns are special events that are not held everyday, we replace these values with 0)



	Store	Dept	weeklySales	isHoliday	Size	Temperature	MarkDown1	MarkDown2	MarkDown4	MarkDown5	Month
0	1	1	24924.50	False	0	42.31	0.0	0.0	0.0	0.0	2
1	1	1	46039.49	True	0	38.51	0.0	0.0	0.0	0.0	2
2	1	1	41595.55	False	0	39.93	0.0	0.0	0.0	0.0	2
3	1	1	19403.54	False	0	46.63	0.0	0.0	0.0	0.0	2
4	1	1	21827.90	False	0	46.50	0.0	0.0	0.0	0.0	3

Train-valid split:

We split train dataset into small size that containing 10,000 rows in order to find best estimator fast. Otherwise, it will take whole day to optimize hyper parameters. Then, we split train dataset into valid dataset and another train dataset.The valid dataset may contain about 100,000 rows. This is for checking metrics to judge the performance of our three methods.

Fine-tuning hyper-parameters:

With smaller size train dataset, we use GridSearchCV function to search best hyper parameters. During this process, we use 5 times cross validation to improve the precision. And we use mean absolute error metrics to judge whether we find the best one.

For this problem, we tried three different models

KNN:

In pattern recognition, the k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of $\frac{1}{d}$, where d is the distance to the neighbor.

Random Forest:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set.

Extra Trees:

Adding one further step of randomization yields extremely randomized trees, or ExtraTrees. While similar to ordinary random forests in that they are an ensemble of individual trees, there are two main differences: first, each tree is trained using the whole learning sample (rather than a bootstrap sample), and second, the top-down splitting in the tree learner is randomized. Instead of computing the locally optimal cut-point for each feature under consideration (based on, e.g., information gain or the Gini impurity), a random cut-point is selected. This value is selected from a uniform distribution within the feature's empirical range (in the tree's training set). Then, of all the randomly generated splits, the split that yields the highest score is chosen to split the node. Similar to ordinary random forests, the number of randomly selected features to be considered at each node can be specified. Default values for this parameter are $\sqrt{n}$ for classification and n for regression, where n is the number of features in the model.

Results

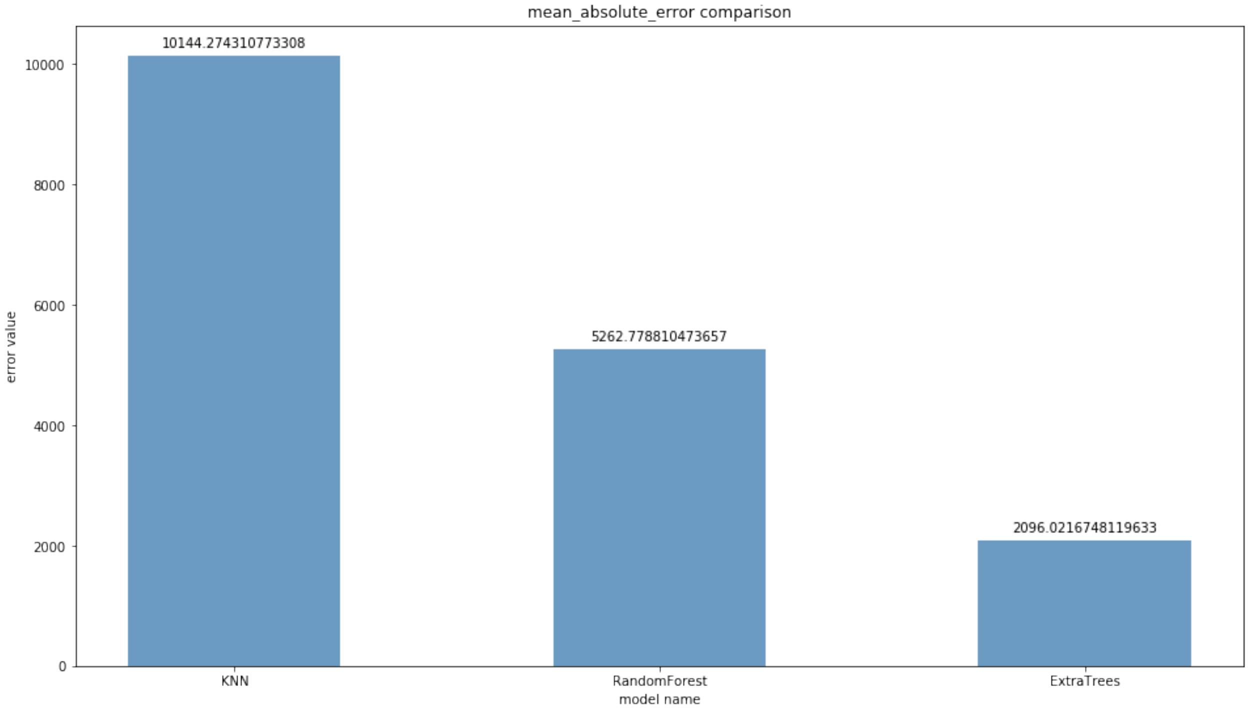
Since this is a regression problem, the most common metrics for evaluating performance of such models are `mean absolute error` and `mean squared error`

Mean Absolute Error:

In statistics, mean absolute error (MAE) is a measure of difference between two continuous variables. Assume X and Y are variables of paired observations that express the same phenomenon. Examples of Y versus X include comparisons of predicted versus observed, subsequent time versus initial time, and one technique of measurement versus an alternative technique of measurement. Consider a scatter plot of n points, where point

i has coordinates (x_i, y_i). Mean Absolute Error (MAE) is the average vertical distance between each point and the identity line. MAE is also the average horizontal distance between each point and the identity line. The mean absolute error is given by:

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

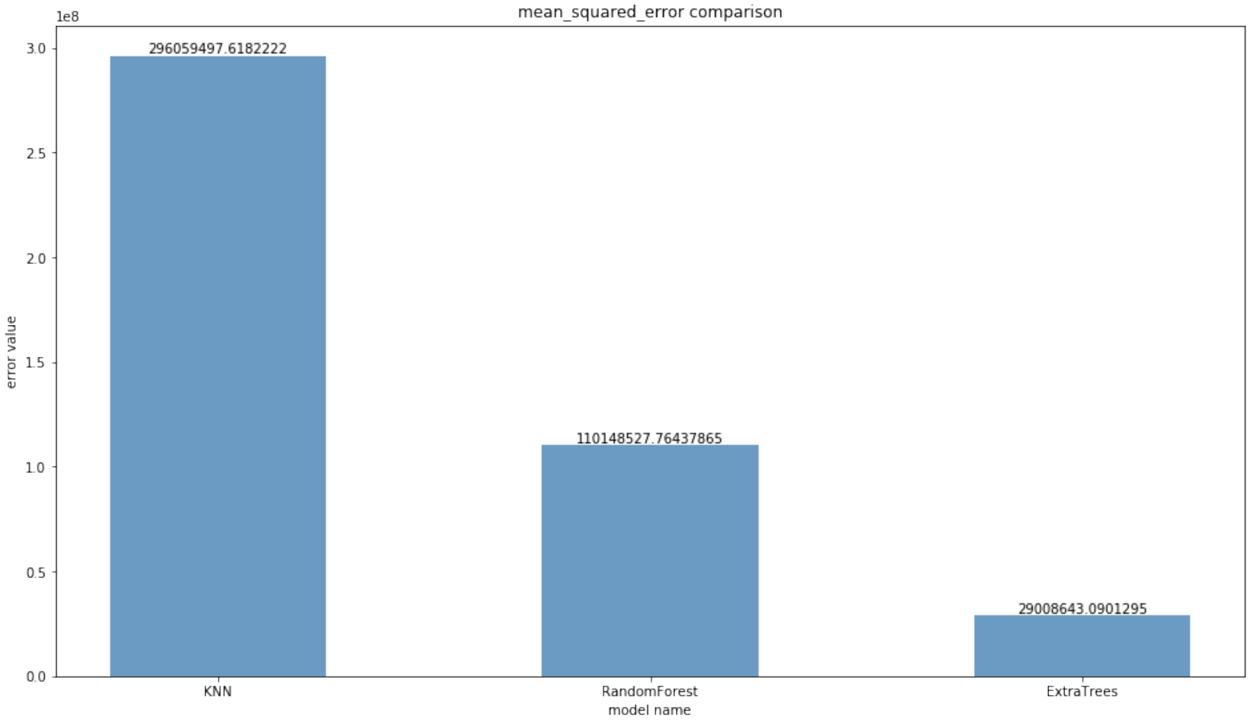


The mae of the three models are shown as above.

Mean Squared Error:

n statistics, the mean squared error (MSE) or mean squared deviation (MSD) of an estimator (of a procedure for estimating an unobserved quantity) measures the average of the squares of the errors—that is, the average squared difference between the estimated values and the actual value. MSE is a risk function, corresponding to the expected value of the squared error loss. The fact that MSE is almost always strictly positive (and not zero) is because of randomness or because the estimator does not account for information that could produce a more accurate estimate. The MSE is a measure of the quality of an estimator—it is always non-negative, and values closer to zero are better. The mean squared error is given by:

$$MSE = \frac{\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2}{n}$$



The mse of the three models are shown as above.

(*We only include significant findings in this paragraph, other explorations can be found in the notebook* `model-training.ipynb`)