**PR-Milestone 2 Report**

**Table of Contents**

[**Preprocessing 2**](#_Toc74312057)

[**Data Analysis 3**](#_Toc74312058)

[**Classification Techniques 6**](#_Toc74312059)

[**Model Acquired Results 7**](#_Toc74312060)

[**Features Used/Discarded 8**](#_Toc74312061)

[**Data Size 9**](#_Toc74312062)

[**Results Accuracy 10**](#_Toc74312063)

[**Conclusion 14**](#_Toc74312064)

# **Preprocessing**

1. Renaming columns names from X1,X2,… to a better naming convention ( Item ID, Store ID, etc..) by using function rename in pandas.

Renaming in detail:

* X1: Item\_ID
* X2: Item\_Weight
* X3: Item\_Fat\_Amount
* X4: Item\_Store\_Allocation
* X5: Item\_Category
* X6: Item\_Price
* X7: Store\_ID
* X8: Store\_Establishment\_Year
* X9: Store\_Size
* X10: Store\_Location\_Type

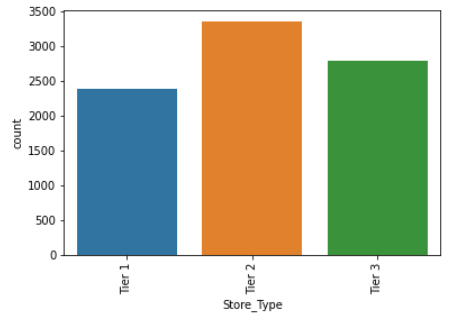
1. Replacing the following in item fat amount column using pandas rename:
   1. LF to Low Fat
   2. Low fat to Low Fat
   3. Reg Regular
2. Filling the NaN’s in item weight using with backward fill– using fillna function found in pandas with a method parameter=’bfill’
3. Filling the 0’s in item store allocation with backward fill -using fillna function found in pandas with a method parameter=’bfill’
4. Filling the NaN’s in store size with backward fill - using fillna function found in pandas with a method parameter=’bfill’
5. We performed label encoding on Store Size , Store Location Type, Item category , Item fat amount, and store ID

# **Data Analysis**

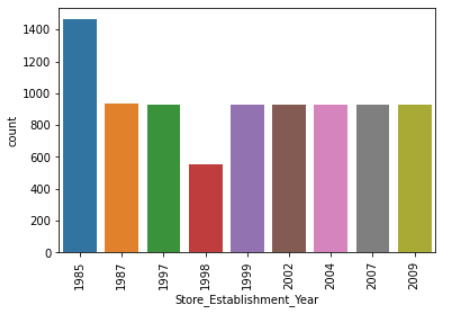


Visually, as we can see above. This is the correlation between each feature with the other. In our case, we will mainly focus on the relation between all features with our target which is “Y”.

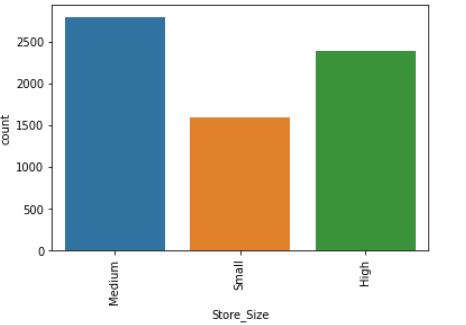
Store Type count



Store Establishment Years



Store Size count



Here is a summary table that shows the relation of all features with our target.

|  |  |
| --- | --- |
| Feature | Correlation with target “Y” |
| Item\_Weight | -0.02 |
| Item\_Fat\_Amount | 0.00 |
| Item\_Store\_Allocation | 0.02 |
| Item\_Category | 0.00 |
| Item\_Price | 0.01 |
| Store\_ID | -0.21 |
| Store\_Establishment\_Year | -0.29 |
| Store\_Size | -0.11 |
| Store\_Location\_Type | 0.56 |

Based on the table above, our top features are:

1. Store\_Location\_Type
2. Store\_ID
3. Store\_Establishment\_Year
4. Store\_Size

# **Classification Techniques**

1. SVC Model
2. KNN
3. Decision Tree
4. Naïve Bayes
5. XGbooster
6. RandomForest
7. Logistic Regression

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Advantages | Disadvantages | Accuracy | Error |
| SVC Model  (Kernel:Poly) | high accuracy compared to other classifiers such as logistic regression, and decision trees. It is known for its kernel trick to handle nonlinear input spaces. | Choosing a “good” kernel function is not easy. | 100% | 0% |
| KNN | * No training period * New data can be added without effecting the accuracy | * Does not work well with large dataset. * Does not work well with high dimensions | 100% | 0% |
| Decision Tree | Does not require normalization or scaling | Long training time | 100% | 0% |
| XGBooster | It is designed to handle missing data with its in-build features | you must label encoding for categorical features before feeding them into the models | 100% | 0% |
| Random Forest | It works well with both categorical and continuous values | It also requires much time for training as it combines a lot of decision trees to determine the class. | 100% | 0% |
| Naïve Bayes | his algorithm works quickly and can save a lot of time | Its estimations can be wrong in some cases, so you shouldn’t take its probability outputs very seriously | 95% | 5% |
| Logistic Regression | Good accuracy for many simple data sets and it performs well when the dataset is linearly separable | The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables | 98.83% | 1.17% |

# **Model Acquired Results**

# **Features Used/Discarded**

* Features Used:

1. Store ID
2. Store\_Establishment\_Year
3. Store\_Size
4. Store\_Location\_Type

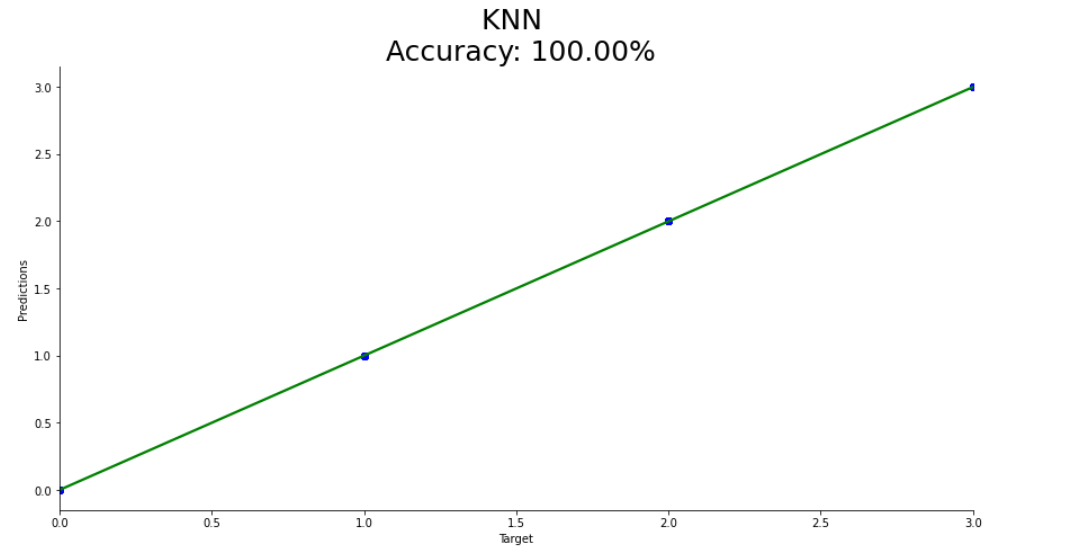
* Features Discarded:

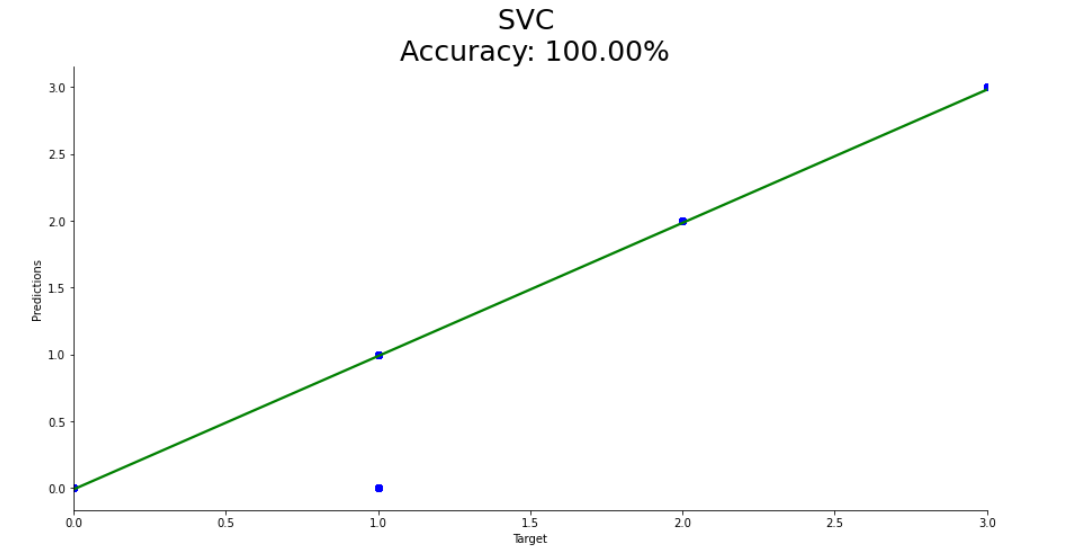
1. Item ID
2. Item Weight
3. Item fat amount
4. Item store allocation
5. Item category
6. Item price

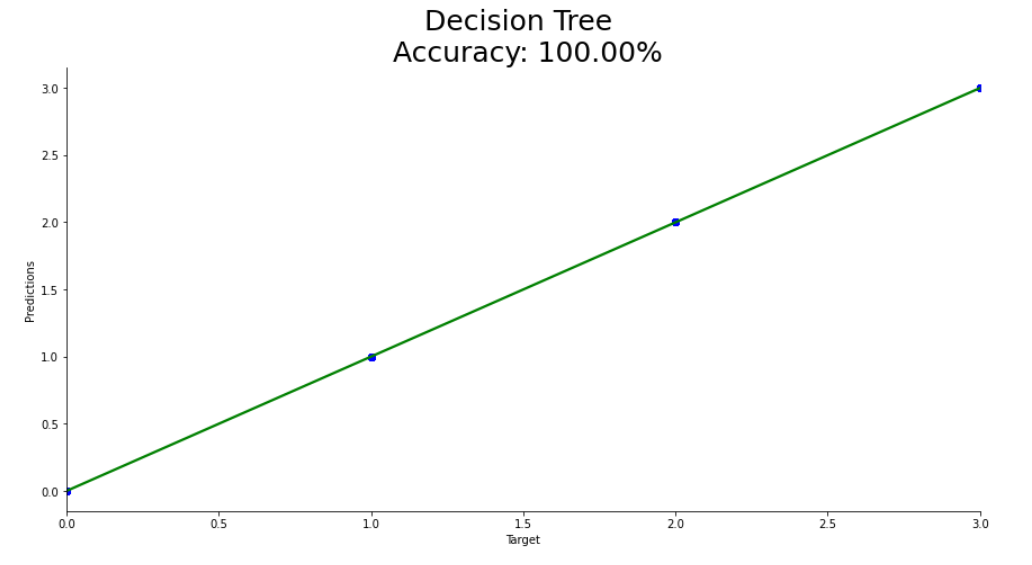
# Data Size

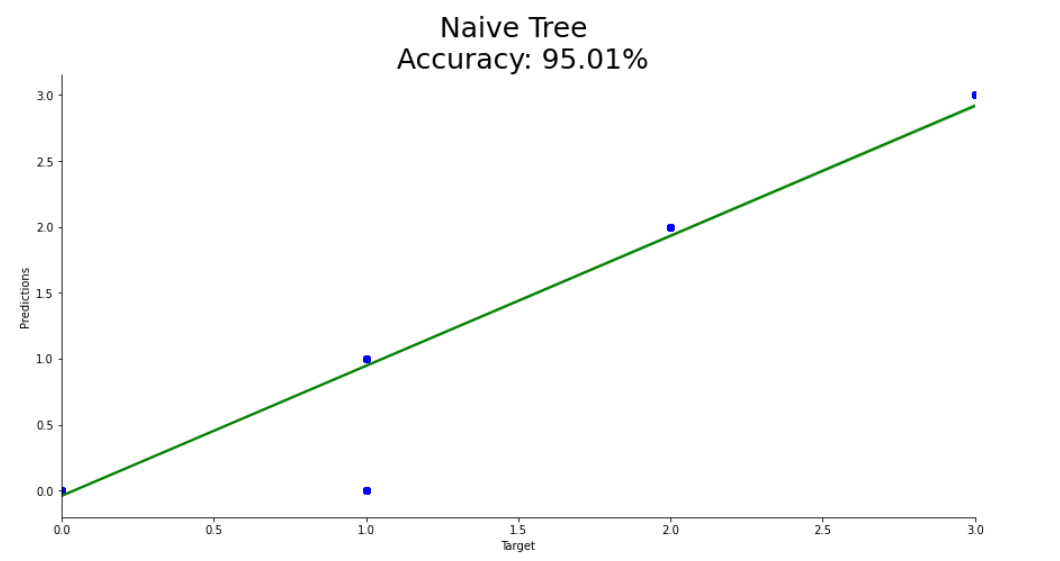
The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model. Where training takes 80% from the dataset and testing takes 20% from the dataset.

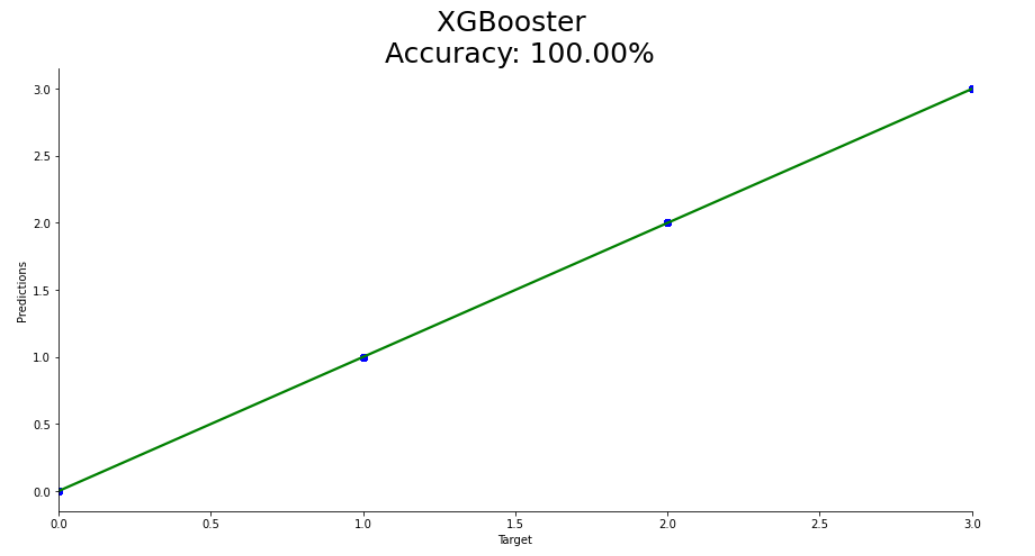
# **Results Accuracy**

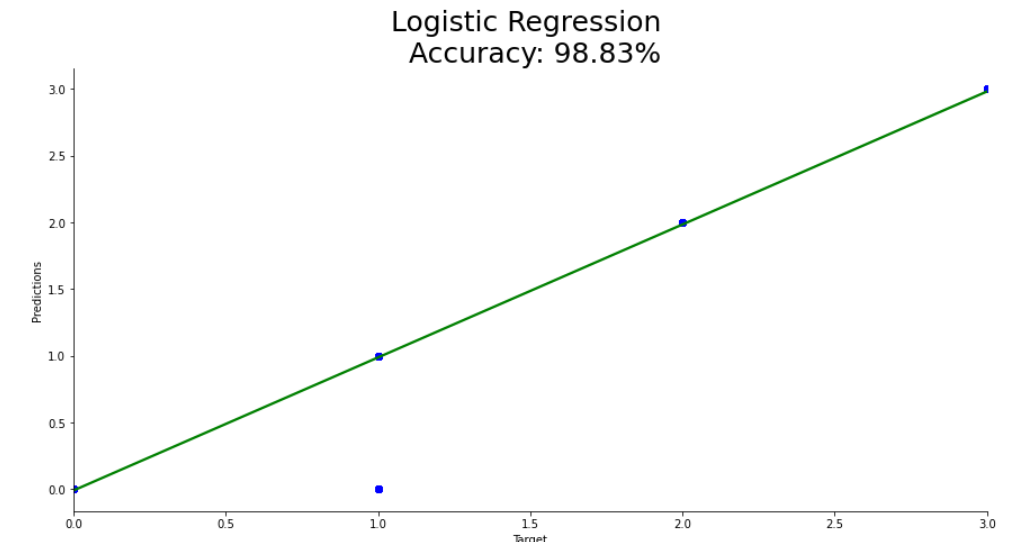


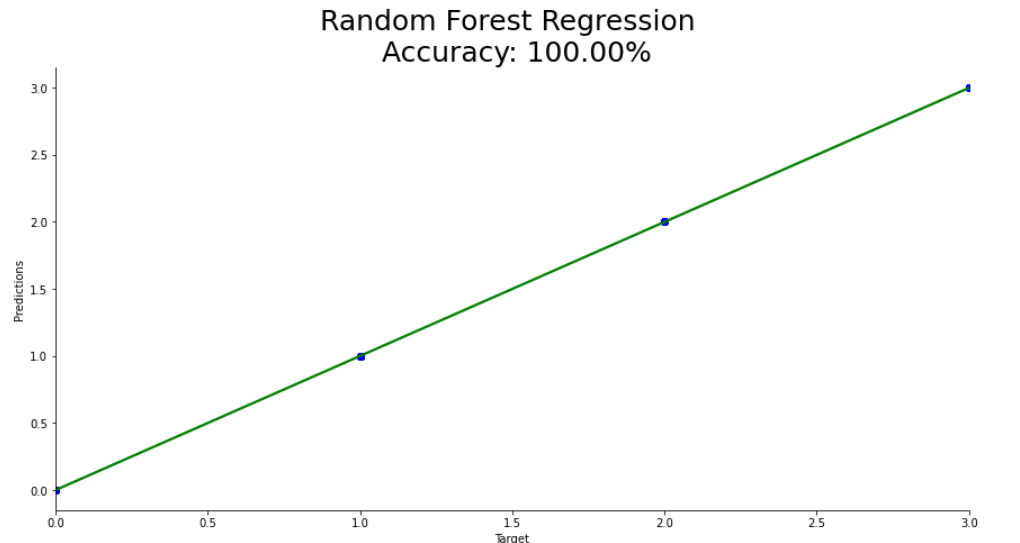












# **Conclusion**

In conclusion, all the store related features had the highest correlation with our target “Y” (Store Type). My Intuition was all features regarding the item’s wouldn’t be considered as our target was the store type. So, after checking the correlation. It was clearly shown that all stores’ features will be disregarded.