Mid Term Project Report
INFO 6105 Fall 2023

Rishabh Kaushick
NU ID: 002808996
College of Engineering,
Northeastern University
Toronto, ON, Canada
kaushick.r@northeastern.edu

Abstract

This is the abstract which is indented on both left and right.

Dataset

Dataset	Name	Dataset Characteristics	Attribute Characteristics	Associated Task	Number of Instances	Number of Attributes
1	DataCo Supply Chain Management for Big Data Analysis	Multivariate	Real	Classification	180,519	53

Table 1: Understanding the type of dataset.

The Kaggle Data set contains two csv files. Figure 1 below shows all the columns present in the structured csv and the un-structured csv files. The structured csv document contains a total of 53 columns, and 180,519 rows which results in 9,567,507 total data points. On the other hand, the un-structured CSV document contains 8 columns and 469,977 rows therefore resulting in 3,759,816 total data points. Although the unstructured csv has a greater number of rows, it still has lesser number of data points compared to the structured csv data. Since we cannot incorporate the data from the unstructured csv data to the structured csv data, we will not be using it for the rest of the project.

Why I Find the Dataset Interesting

It is very interesting to know that many of the products which we purchase with a single click on Amazon (or other e-commerce platforms) started their journey a few months before we place the order. Nowadays most products are manufactured in Southeast Asia in countries like Vietnam and are usually placed on a ship for a month-long journey across the Pacific Ocean to reach the North American continent.

The supply chain industry has been brought to the limelight after two major disruptions – the COVID-19 pandemic and the Evergreen ship getting stuck in the Suez Canal.

Data Characteristics

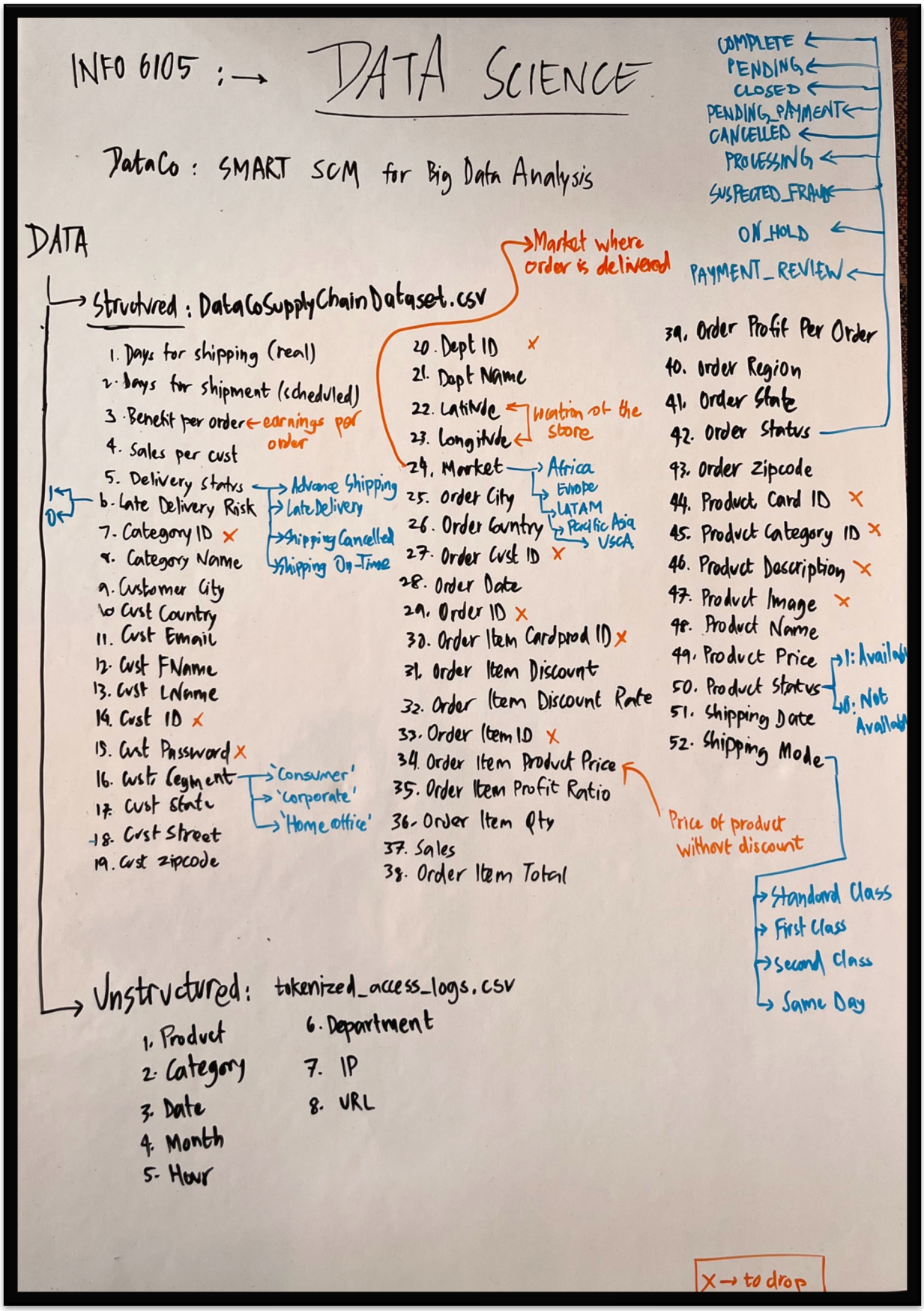

Figure 1: List of the columns present in the structured and unstructured dataset provided in Kaggle.

Figure 1 above lists all the columns in the dataset. The explanation of some columns are mentioned in orange and each of the columns containing categorical data have been mentioned in blue. The columns marked with '×' are columns which do not hold any importance while training the model. Some of these columns include primary keys like 'Customer ID', 'Order Item ID' and 'Product Card ID'. Other columns like 'Password' or 'Product Image' are also do not necessarily add any value for the machine learning classifier. Therefore, we can safely drop these columns before the model.

Understanding & Visualizing the Data

Types of Data

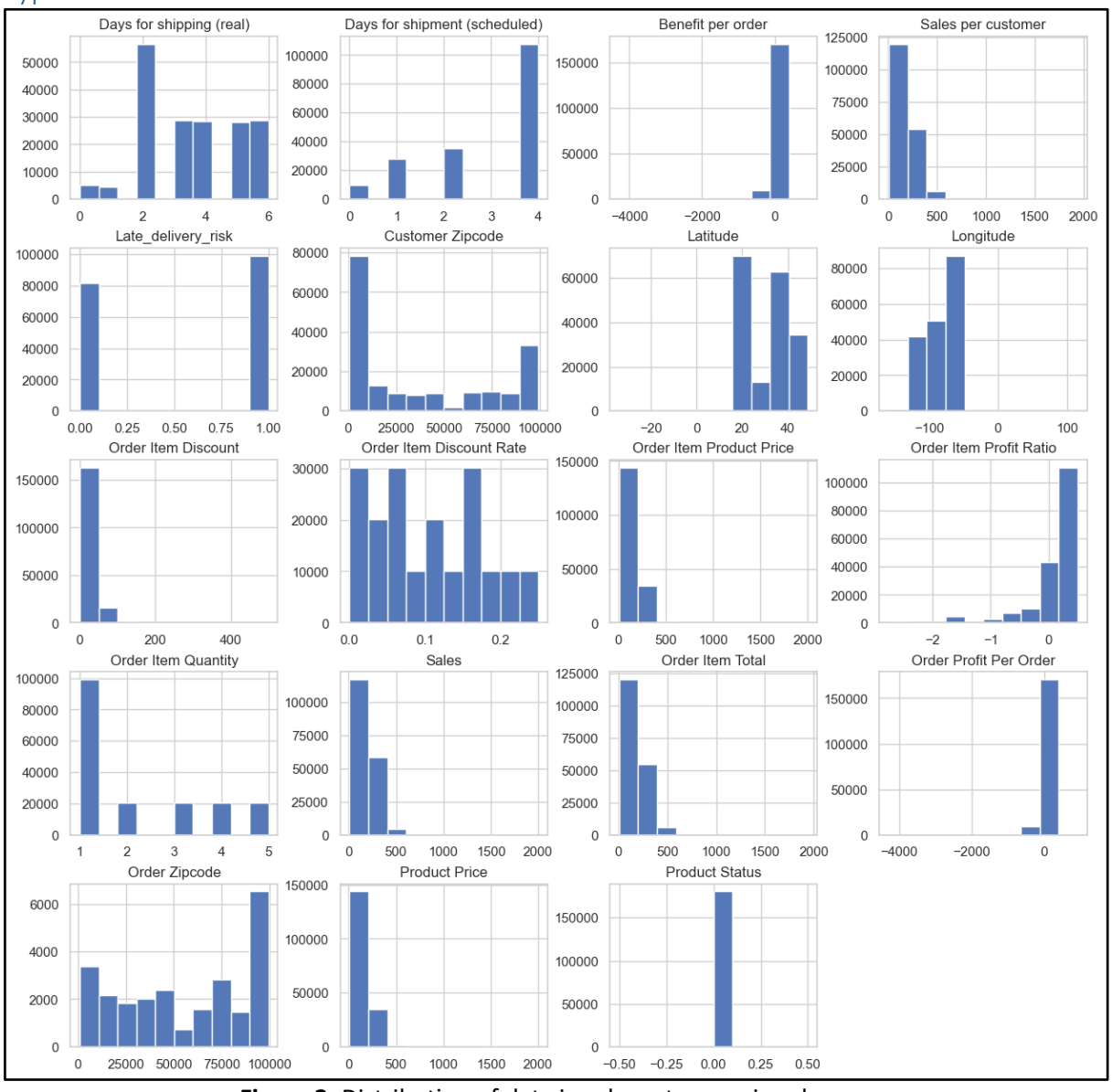

Figure 2: Distribution of data in relevant numeric columns

Something here

Delivery Status & Late Delivery Risk

Based on Figure 3 below, we can see that most of the orders have the delivery status as 'Late delivery'.

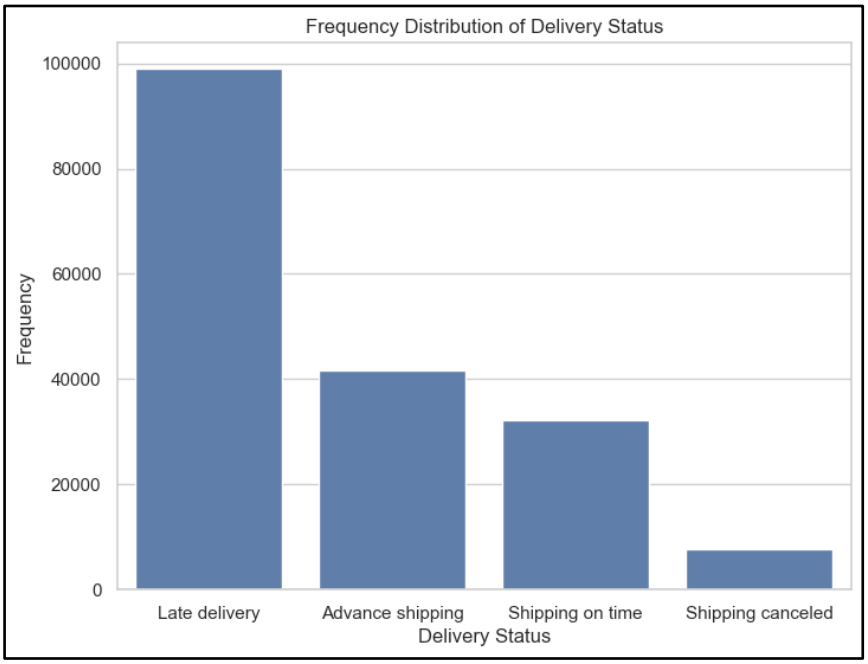

Figure 3: Types of Delivery Status

There is another column which has late delivery risk as shown below in Figure 4. There are 98,977 values with late delivery risk, and 81,542 values that do not have a late delivery risk.

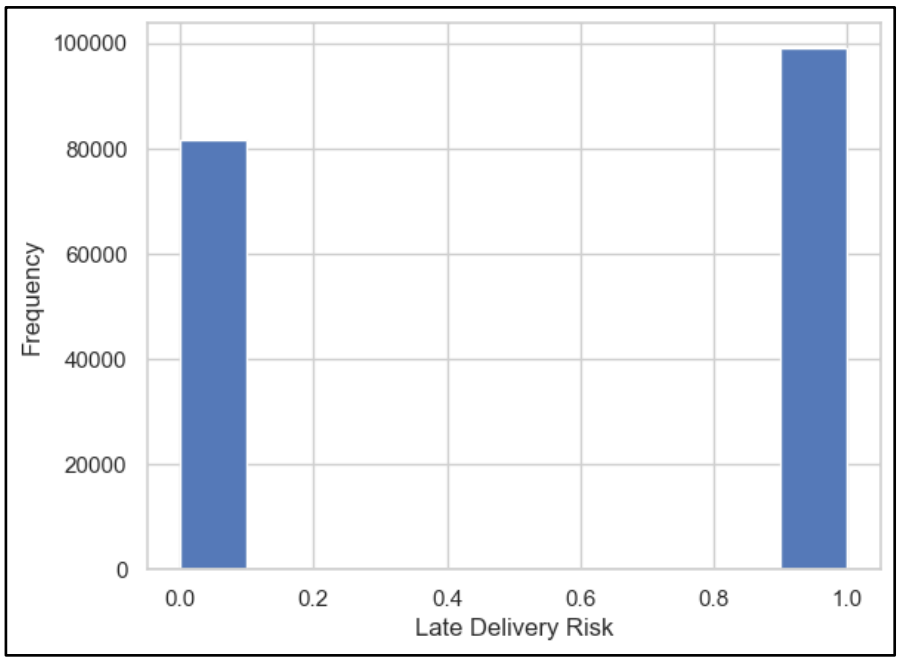

Figure 4: Late delivery risk (0 – no risk, 1 – high risk)

I tried to look deeper into whether the delivery status has any impact on the late delivery risk, and therefore plotted them together in one graph (Figure 5 below) – with x axis as the

delivery status, y axis as the frequency and a color denoting if there is a high or low delivery risk.

Figure 5: Delivery status and late delivery risk plotted together.

The above graph is counterintuitive. We can see that there is a late delivery risk only on those data which have delivery status as 'Late delivery', and all other delivery statuses do not have a late delivery risk. We can infer from this figure that the delivery status and the late delivery risk are two columns which are talking about the same data. Hence, we can drop one of these columns before training the model.

Another interesting visualization is the frequency of Late Delivery Risk color-coded based on each different region. The graph in Figure 6 shows that the late delivery risk does not seem to be region specific.

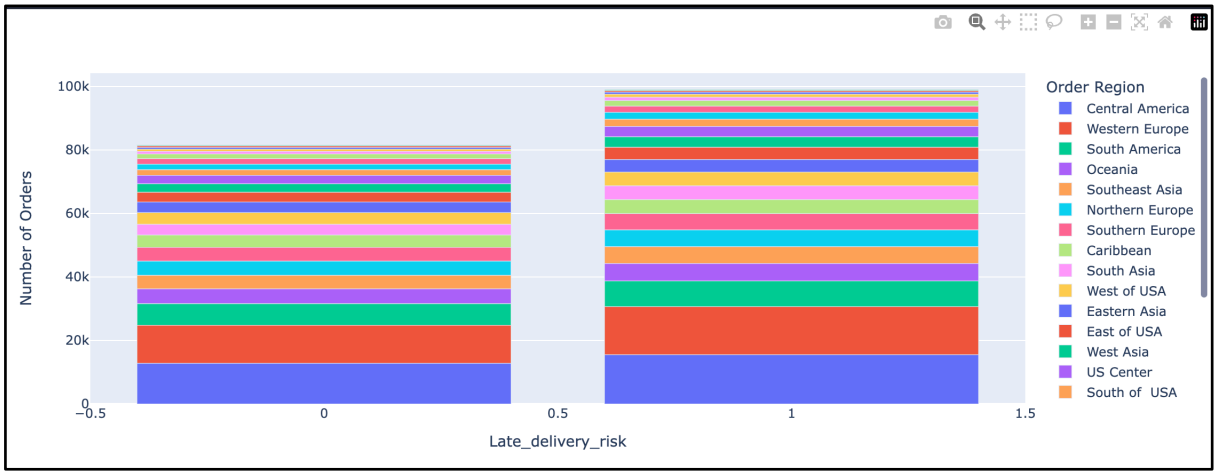

Figure 6: Late Delivery Risk vs Order Region

Late Delivery Risk &

Store Locations on World Map

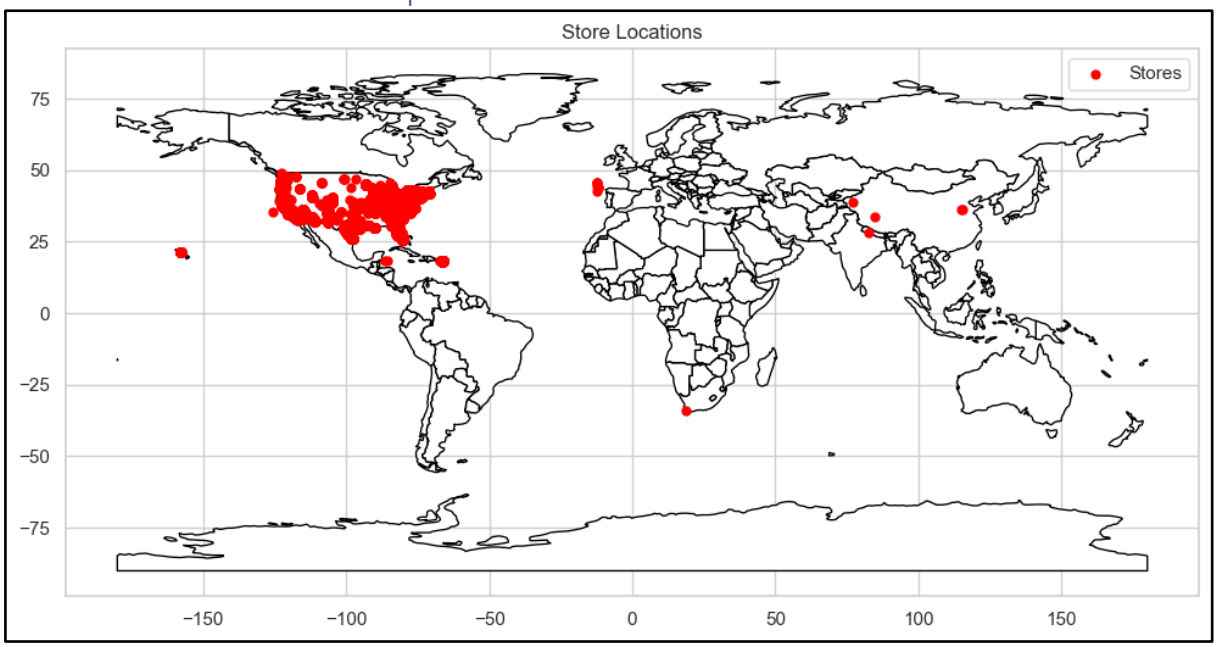

Figure 7: Store locations on the world map.

Based on the latitude and longitude values provided for the store locations we can use geopandas library to visualize the location on the world map as shown in Figure 7. From this visualization, we can see that a vast majority of the stores are present in United States of America. There are a few stores South Africa, China, India and perhaps Portugal/ Spain, and no stores in Canada.

Feature Engineering

Customer Location – Latitude & Longitude

Similar to Figure 7, it is important for us to understand the customer delivery locations. One major issue for the late deliveries could be because the customer delivery locations are far away from the stores. Further information is discussed in the Discussions section.

Defining The Problem

Based on this dataset, I have analyzed that there are multiple problems for which we can devise a machine learning algorithm:

1. Predicting the risk of late delivery: Classification problem
2. Predicting the Sales: Regression problem
3. Order suspected fraud: Classification problem.

Out of the above three issues, I decided to work on the first issue of predicting late delivery risk because of the following reasons:

- Unique issue – No one in the Kaggle has worked on predicting the risk of late delivery. There have been many people who have worked to analyze the suspected fraud orders.
- Enough data – there are enough data points for late delivery and other delivery statuses. This will ensure that the model has enough ways to learn for both the positive and negative scenarios. For suspected fraud orders, there is much lesser data for positive and negative scenarios.
- Business Use case – Predicting the risk of late delivery can help the business to understand which products usually are delayed and understand the factors behind why certain products are delivered late.

Splitting & Training the Dataset

```
# Next creating the train and test set

# creating a function which splits the data randomly
def split_train_test(data, test_ratio):
    np.random.seed(23) # setting the random number generator's seed will make sure that each time the same test and train set will be considered.
    shuffled_indices = np.random.permutation(len(data))
    test_set_size = int(len(data) * test_ratio)
    test_indices = shuffled_indices[:test_set_size]
    train_indices = shuffled_indices[test_set_size:]
    return data.iloc[train_indices], data.iloc[test_indices]
```

[125] ✓ 0.0s Python

```
# here we are splitting the data into 80% training and 20% testing sets.
scm_trainset, scm_testset = split_train_test(scm_clean_df, 0.2)
```

[126] ✓ 0.0s Python

```
len(scm_trainset)
```

[127] ✓ 0.0s Python

```
144416
```

```
len(scm_testset)
```

[128] ✓ 0.0s Python

```
36103
```

We have split the dataset into 80% training and 20% testing sets. There are a total of 144,416 records in the train-set and 36,103 records in the test-set.

Preparing the Model

Selection Of the Model – Decision Trees, Random Forest & XG-Boost

In this dataset, I explored how decision trees specifically and Random Forest can be useful to make predictions. I chose decision trees because of their white box nature. Once the model has been trained on the data, we can visualize the decision tree which was created to understand why the machine would predict a certain value.

Furthermore, once we have a random forest model, we can go one step further to understand which of the features present in the dataset are important for improving the model to make better predictions. For this reason, we can use the XG-Boosting algorithm to generate the feature importance map. This is very useful for:

1. Developers:
    a. Improve model with only important features and remove unimportant unnecessary features.
2. Business Users:
    a. To understand which of the attributes play an important role to make sure that the products are not delivered late.

Evaluation of Model

Decision Tree Model

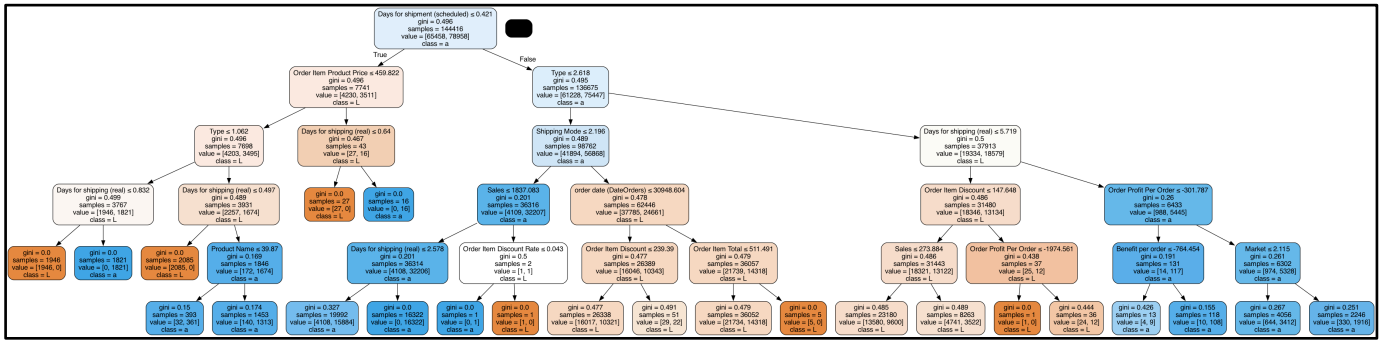

Figure above shows the decision tree model generated by the machine.

To evaluate the predictions, I have used the mean squared error method. When I ran the predictions for the first time, the percentage of error was 54.6%, therefore training accuracy was 46.4%.

Based on this information, I reckon that the model did not learn much from the training dataset. One reason for this could be that 5 levels of the decision tree is not enough for the model to make inferences from the data. Therefore, another model was created with a maximum depth of 10 levels, and a different random state – which yielded better training and test accuracy of 52%.

Conclusion

In this mid-term exam, we have worked on a large dataset using Decision Trees, to try to solve a real-world problem in the Supply Chain industry. We have been able to visualize the data, train the model and received a train and test accuracy of 52%.

Discussion

For the scope of this mid-term project, feature engineering was not completed due to complexities and time restrictions. However, post the mid-term project the same feature engineering will be explored. This way we will be able to feed the model the store latitude and longitude, while also having order city and country in terms of latitude and longitude.

I plan to improve the decision tree model and implement Random Forest model as well to see how it fairs while comparing it with the Decision Tree model. Furthermore, to understand the important features, I plan to also use XG-Boost algorithm.

Acknowledgements

The dataset for this project has been obtained from the following websites:

- https://www.kaggle.com/datasets/shashwatwork/dataco-smart-supply-chain-for-big-data-analysis/data
- https://data.mendeley.com/datasets/8gx2fvg2k6/5

References

1. *Why Global Supply Chains May Never Be the Same: WSJ Documentary*. YouTube. (2022, March 23). https://youtu.be/1KtTAb9Tl6E?si=11--rHEKcscN7Gs5
2. *Pandas.dataframe.plot.pie*. pandas.DataFrame.plot.pie - pandas 2.1.1 documentation. (n.d.). https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.plot.pie.html
3. *Seaborn.barplot*. seaborn.barplot - seaborn 0.13.0 documentation. (n.d.). https://seaborn.pydata.org/generated/seaborn.barplot.html
4. *Seaborn.countplot#*. seaborn.countplot - seaborn 0.13.0 documentation. (n.d.). https://seaborn.pydata.org/generated/seaborn.countplot.html
5. *Pandas.dataframe.hist#*. pandas.DataFrame.hist - pandas 2.1.2 documentation. (n.d.). https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.hist.html
6. Lucaswolff. (2023, September 10). *Inferential statistical analysis on Supply Chain*. Kaggle. https://www.kaggle.com/code/lucaswolff/inferential-statistical-analysis-on-supply-chain
7. Navlani, A. (2023, February 23). *Python decision tree classification tutorial: Scikit-Learn Decisiontreeclassifier*. DataCamp. https://www.datacamp.com/tutorial/decision-tree-classification-python
8. *Sklearn.tree.decisiontreeclassifier*. scikit. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html