**Project Report**

**CSIS 3290**

Student Name: Toan Diec

Student ID: 300320364

1. **Introduction and discovery**

The project is about applying data analysis pipeline from data preparation, modelling, model evaluation and tunning on an unclean dataset that contains information of more than 1 million rows of real data. The data file contains the historical data for the 8 weeks prior to the week we are trying to predict. The data were taken as weekly snapshots at the start of each week. The goal of this project is to develop a model that can predict went\_on\_backorder variable based on the given dataset. The project includes step by step of how to prepare the data to a fully running model using python as the main coding language.

Product back order is one of the common issues in supply chain management. Back order simply means that a customer order has not been fulfilled. The issue can raise some potential alert such as demand excess supply, bottle neck in supply chain or strong sale performance. Most of the companies want to avoid back order because it can upset the customer, cancelling order and lead to lower customer loyalty. However, companies also do not want to over stocking, which can add up the inventory cost. To solve this issue, this project will apply machine learning to seek and predict any pattern related to back order before it happens. Result of this predictive analytics approach can save organizations money and ensure customer will get their order with minimum waiting time.

For the initial hypothesis, I will assume that the pattern of back order is dependent on 22 variables in the dataset.

1. **Data Preparation**

The dataset can be found on data.world using this link: [*https://data.world/amitkishore/can-you-predict-products-back-order*](https://data.world/amitkishore/can-you-predict-products-back-order)

The dataset has more than 1 million row and 23 variables:

* sku – Random ID for the product
* national\_inv – Current inventory level for the part
* lead\_time – Transit time for product (if available)
* in\_transit\_qty – Amount of product in transit from source
* forecast\_3\_month – Forecast sales for the next 3 months
* forecast\_6\_month – Forecast sales for the next 6 months
* forecast\_9\_month – Forecast sales for the next 9 months
* sales\_1\_month – Sales quantity for the prior 1-month period
* sales\_3\_month – Sales quantity for the prior 3-month period
* sales\_6\_month – Sales quantity for the prior 6-month period
* sales\_9\_month – Sales quantity for the prior 9-month period
* min\_bank – Minimum recommend amount to stock
* potential\_issue – Source issue for part identified
* pieces\_past\_due – Parts overdue from source
* perf\_6\_month\_avg – Source performance for prior 6-month period
* perf\_12\_month\_avg – Source performance for prior 12-month period
* local\_bo\_qty – Amount of stock orders overdue
* deck\_risk – Part risk flag
* oe\_constraint – Part risk flag
* ppap\_risk – Part risk flag
* stop\_auto\_buy – Part risk flag
* rev\_stop – Part risk flag
* went\_on\_backorder – Product went on backorder. This is the target value.

The dataset has more than 1 million rows and 23 columns, as the picture below. It also contains a mixing of numeric variables and category variables.

Graphical user interface, text

Description automatically generated

There are a lot of nan values in the dataset, especially in lead\_time column. Graphical user interface

Description automatically generated with medium confidence

The dataset has imbalance problem. The majority variables have the value significantly outweighs the value of minority variables. To deal with unbalanced data, SMOTE (synthetic minority over-sampling technique) will be used in model implementation part. Calendar

Description automatically generated

SMOTE will create over-sampled synthetic examples of the minority classes. These new synthetic examples are based along the line segments joining a defined number of k nearest neighbor, which is set to five by defaut. To use SMOTE, the user first need to install the library from CMD prompt in ANACONDA Navigator. The code for installing is **pip install imblearn.** After that, I split the data in to trin and test dataset. Finally, I use the train dataset to perform SMOTE transformation.

Graphical user interface, text, application, email

Description automatically generated

1. **Model Planning**

Based on the structure of the dataset, including numeric and category variables, I found out that it is suitable to develop classification models. The target variable is went\_on\_backorder column which indicates the status of the order. It only has two values yes or no. Yes, means the order is listed as backorder. No, means there is no problem with the order, and it is delivered to the customer without any extra waiting time. To classify the target variable, multiple classification model will be used, such as Logistic Regression, Naïve Bayes, Support Vector Machine (SVM), k-Nearest Neighbors (KNN), Decision Trees combining with multiple techniques of feature selection and feature transformation to generate results. Then the results are compared with each other using confusion matrix, ROC curve, precision and recall table to select the best model for classification. Finally, the chosen model will be utilized to run on an out-of-sample dataset to simulate how the model would perform if deployed in the production environment in the real world.

1. **Model Implementation**

* **Feature Engineering**

In the original dataset, lead\_time column contains many null values, more than 100 thousand null values. So, I decided to fill the null values with the average value of lead\_time column.

Text

Description automatically generated with medium confidence

After that, I checked and dropped any null value left and duplicated rows in the dataset. Next, there are several category columns with string value, such as potential\_issue, deck\_risk, stop\_auto\_buy, oe\_constraint, ppap\_risk, rev\_stop, and went\_on\_backorder. For the best performance of the upcoming model, I transformed all the string value into category numeric values, with 0 and 1. Looking for any sign of issue, I used histogram plot for every column. In columns perf\_6\_month\_avg and perf\_12\_month\_avg, I saw some negative values, which is not correct. These columns should not have any negative values.

Chart, histogram

Description automatically generatedThe negative value (-99), represent for null value. Because of that, I drop all the negative values in these two columns.

To check for imbalance problem, which occurs very often in real life data, I used value\_counts function on the target column, went\_on\_backorder. A picture containing text

Description automatically generated

As the result above, we can see that the number of class 1 value in our target column is outweighed by the class 0 value with the ratio of more than 1 to 1000. The issue is dealt with using combination of taking fewer class 0 value and Synthetic Minority Oversampling Technique (SMOTE) technique. I randomly take only 1% of the rows in the dataset that has class 1 value in column went\_on\_backorder and concatenate it with the data frame that has the value of class 0 value.

Graphical user interface, text, application, email

Description automatically generated

The result is I have a data frame with more than 25 thousand rows with less imbalance problem.

1. **Feature Selection**

For this project, Variance Threshold feature selection is used to select only important enough variables. With the threshold of 0.5, only 13 variables are remaining, including national\_inv, lead\_time, in\_trainsit\_qty, forecast\_3\_month, forecast\_6\_month, forecast\_9\_month, sales\_1\_month, sales\_3\_month, sales\_6\_month, sales\_9\_month, min\_bank, pieces\_past\_due, and local\_bo\_qty. Planning to use some classification methods require predefined k neighbor, I use Elbow method to decide which k neighbor is the best. Chart, line chart

Description automatically generated

With the Elbow method, 3 k neighbor is chosen for suitable techniques such as KNeighborsClassifier, DecisionTreeClassifier.

Next step, the target and feature variables are divided and split into training and test set. To avoid any unwanted influence on the test set and the result, I only apply SMOTE on training set.

Text

Description automatically generated

Then, X\_train\_SMOTE and y\_train\_SMOTE are the two dataset that I will use to train my classification models.

1. **Train and Compare Various Classification Model**

There are several models will be utilized and compared to each other in the project, which are Logistic Regression, Random Forest Classifier, Bagging Classifier, AdaBoost Classifier, Soft Voting Classifier, XGB Classifier, and MLP Classifier. To achieve more efficient workflow, I implement Pipeline technique for the whole process of SMOTE, scaler using Standard Scaler, and the classifier methods listed above.

Text

Description automatically generated

After the training process, the accuracy score for each model is combined and sorted for a better understanding and comparation.

Table

Description automatically generated

Among the models, XGBoost, which is XGB Classifier method, generates the best accuracy outcome for this dataset. With the score of 0.89, the result is quite good. For deeper investigation, classification report, confusion matrix and ROC curve are some suitable performance indicators in this circumstance.

Calendar

Description automatically generated with medium confidence

With the classification report, we can see that the precision for both classes are above 0.85, which is a good sign.

Chart, treemap chart

Description automatically generatedThe confusion matrix tells us that among 3856 class 0 values in the test set the XGB classifier model can predict 3474 correct values and among 2733 class 1 values the model is able to predict 2103 correct values.

Chart, line chart

Description automatically generatedThe ROC curve illustrates the diagnostic ability of XGB classifier system, in this case the curve stands at 0.89.

With the good performance of XGB classifier, I’m confident to move on and apply the model for an out\_of\_sample data set. This will further prove the ability of generate accurate result of the model.

1. **Out-of-sample Predictions**

To test the model, a second separate dataset, with the exact same format with the original dataset is generated.

Text

Description automatically generated with medium confidence

With this dataset, I only keep the columns that is not filtered out by the variance threshold feature selection that was performed inn the feature engineering. It is done because the training model is based on 13 variables after feature selection.

Text

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

The target variable in out-of-sample dataset is also very imbalance. After that, I split the dataset into feature and target variables, then run the previous training model.

Graphical user interface, text, application, email

Description automatically generated

Chart, line chart

Description automatically generatedChart, treemap chart

Description automatically generated

The accuracy score is 0.9 which is very high. The model successfully classifies class 1 and class 0 with high precision. Among 239387 class 0 values the model can predict 216511 correct values (90% precission), and among 2688 class 1 values the model is able to predict 2103 correct values (78% precission).

To improve the performance of the model further, data of class 1 should be collected more. For now, because of the lack information of class 1, the predict precision of class 1 is lower in compare with class 0. Another combination of classification method with different hyperparameters or different scaler could improve the performance furthermore.