CSIS 3290
WINTER 2021

Project Report:
E-commerce Shipping Analysis

TEAM MEMBERS:
DANIIL VOLOVIK (300313470)
DO MAN UYEN NGUYEN (300318626)
FABIO TURAZZI (300318010)

Contents

1. Introduction & Discovery

This project was developed by Daniil Volovik (300313470), Do Man Uyen Nguyen (300318626), and Fabio Turazzi (300318010) with the goal of applying Machine Learning concepts to generate useful insights from the dataset of our choice. Video presentation: https://drive.google.com/drive/folders/14GnFQncfOnB0NFb6_KUdA2TH2xQy-i0w?usp=sharing

1.1. Business and Dataset Introduction

We chose to analyze E-commerce shipping data from an electronics company obtained from Kaggle. The company aims to improve their shipping process by analyzing past data to identify issues in their orders that ultimately create delays. To address this matter, we intend to study the efficiency of the current shipping system and understand the features of orders which currently render delays, building a predictive model that interprets shipment quality based on its characteristics.

To accomplish this, we will analyze a dataset of past orders executed by the company, containing both the target variable (Reached.on.Time_Y.N.) and the additional features listed below:

- **Dataset Features:** Warehouse_block, Mode_of_Shipment, Customer_care_calls, Customer_rating, Cost_of_the_Product, Prior_purchases, Product_importance, Gender, Discount_offered, Weight_in_gms

1.2. Analysis Focal Questions

Our focus for this analysis is to determine which orders (and their characteristics) are currently delivered with delays. This direction will allow us to provide evidence of which orders should be targeted by the management of this company in their efforts to reduce shipping time, guiding their future actions to improve results. With this accomplished, the company will be able to efficiently direct efforts to improve shipment quality and, consequently, customer service.

We also intend to provide to the client a better understanding of their process, by describing variable patterns and relationships inside the dataset. This knowledge will be relevant for planning future market expansions, obtaining valuable insight about the intricacies of their shipping structure.

1.3. Initial Hypotheses

Our initial considerations include some hypotheses for how characteristics may impact shipping quality. These hypothetical scenarios will be tested in the modelling stage of this project. The following list describes all hypotheses, divided into categories of “Warehousing”, “Shipment”, and “Customer and Product Profile”:

- Warehousing
  - Warehouse_block: Blocks provide different delay rates, considering possible differences in personnel, shipped products, and equipment. Rejecting this hypothesis will indicate that the Warehouse processes are exhaustive and well implemented.
- Shipment
  - Mode_of_Shipment: Flight, road, and ship shipment impact delays differently, considering the different natures of each method.
  - Weight_in_gms: Orders of large weight are more likely to generate delays. Alternatively, orders of very small weight may receive less attention and may also render delays.
- Customer and Product Profile
  - Customer_care_calls: There are two aspects to consider here. While customers that performed more calls received attention to their order and reduced delays, calls may also be a response to already delayed orders.
  - Customer_rating & Prior_purchaces: Customers with higher rating or many prior purchases receive more attention in their orders, reducing delays.

- Cost_of_the_Product & Product_importance: Shipment of products with higher cost or importance receive more attention, reducing delays.
- Gender: We do not estimate an impact on shipping delays.
- Discount_offered: Orders with high discount imply more important customers for the company, indirectly generating less delays.

2. Data Preparation

2.1. Data inventory

The selected dataset describes E-commerce shipping data from an electronics company and was obtained from Kaggle in the following link: https://www.kaggle.com/prachi13/customer-analytics

This dataset contains information regarding past orders executed by the company, including several features detailing customers, order, product, and shipping, as described below. Before processing, there are a total 10999 rows with no null values.

Features contained in the dataset:

- Reached.on.Time_Y.N.: Our target variable, describes if an order was delivered on time (1) or delayed (0)
  - This variable was renamed to "Delivery_status" for simplification purposes.
- Warehouse_block: Categorical variable describing the warehouse block responsible for this shipping (A, B, C, D, E, F).
- Mode_of_Shipment: Describes how shipping was performed (Flight, Road, Ship).
- Customer_care_calls: Number of inquire calls made by the customer for this shipping.
- Customer_rating: Rating given by the company to its customers (1-5).
- Cost_of_the_Product: Product cost in US Dollars.
- Prior_purchases: Number of prior purchases performed by the customer.
- Product_importance: Importance categorized by the company for each product (high, medium, low).
- Gender: Gender of the customer.
- Discount_offered: Discount offered for that order (%).
- Weight_in_gms: Total weight of the order in grams.

2.2. Data processing

2.2.1. Summary Statistics

We will focus on the relationship between our dependent variable with the rest of the independent variables to get a better understand of our dataset.

The bar chart on the left describes the *Delivery_status* distribution, with approximately 40% of total orders being delivered late, and 60% being delivered on time.

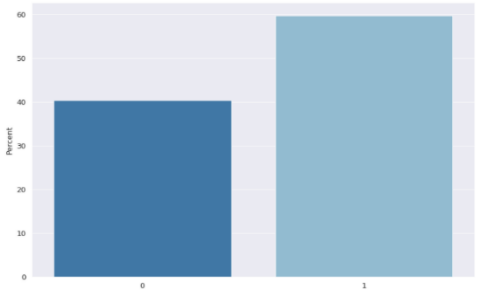
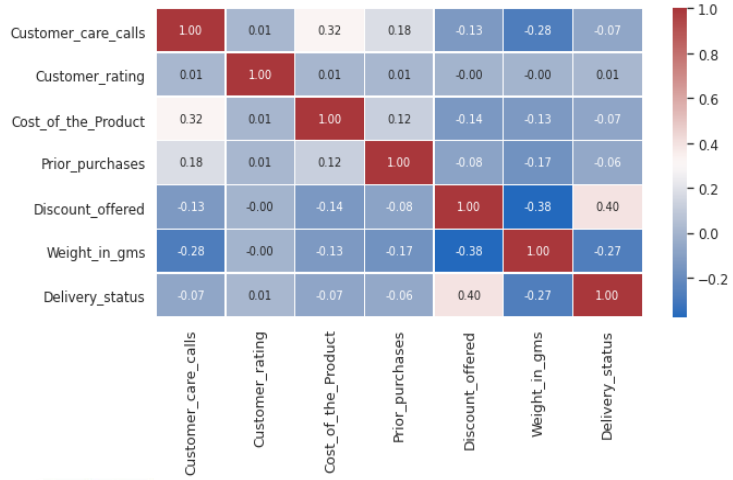


Figure 1: Delivery_status distribution

Figure 2: Correlation Heatmap



The heatmap shows the correlation between the numeric variables.

As for our interest variable, we can see that *Discount_offered* and *Weight_in_gms* have the stronger correlations with *Delivery_status* when compared to the rest of the numeric variables, with *Discount_offered* having positive correlation of 0.4 and *Weight_in_gms* having negative correlation of -0.27. Other numeric variables seem to have weak or no correlation with our target, with their absolute correlation values being less than 0.1.

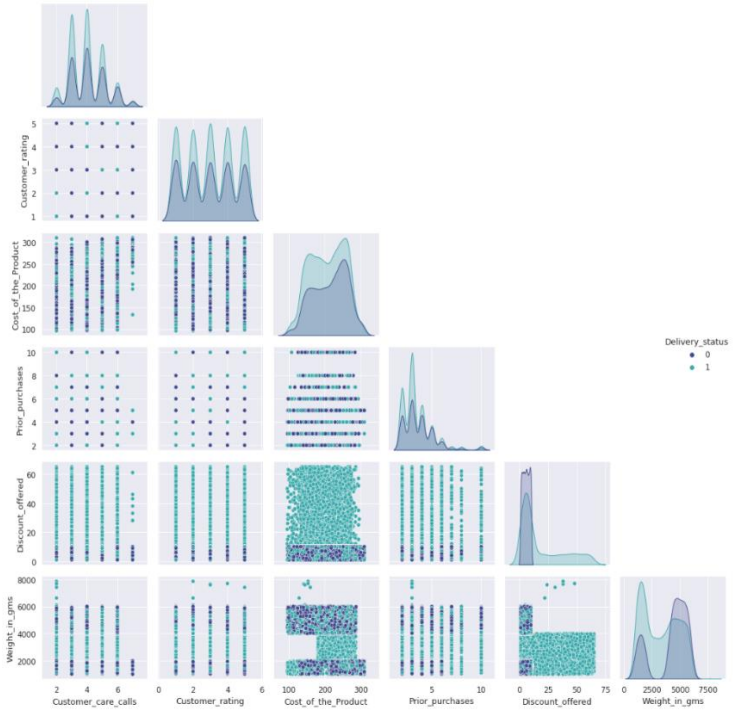


Figure 3: Pairplot of Studied Variables, Delivery_status as Color

The pairplot also corroborates the previous observation, by showing the feature relationships with the dependent *Delivery_status* detailed on different colors. An important conclusion drawn is that all delayed deliveries contained products with *Discount_offered* equal to or below 10%. Additionally, those tardy products either weighed less than 2000grams or more than 4000 grams.

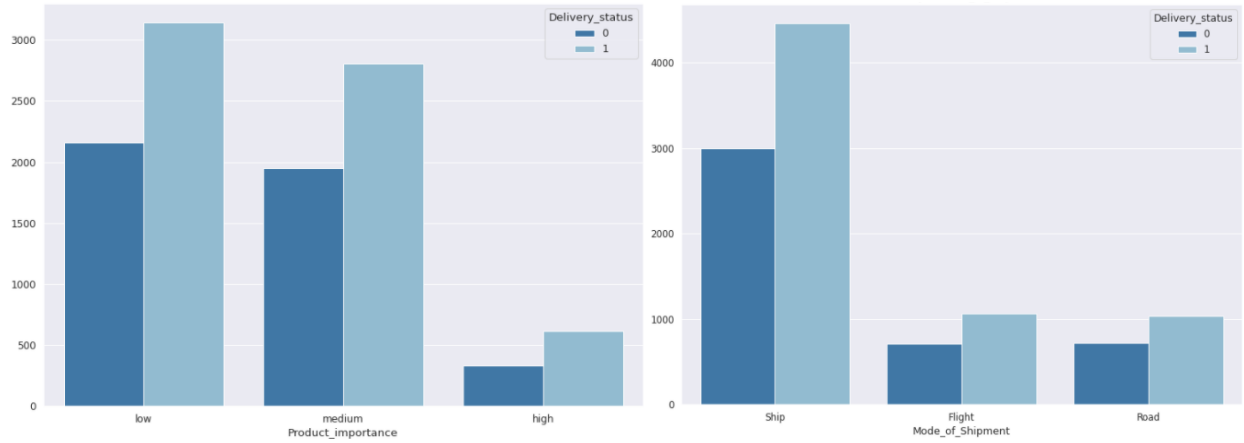
Another notable observation here is that most of the heavier products (greater than 4000 grams) had a discount of 10% or less.

From these observations, it is possible that these two variables might have predominant influence in our model.

Considering this, we also chose to exclude outliers presenting weight above the 6,1000 threshold, which appear to differ from the pattern seen in the rest of the data.

For the categorical variables, most of the distributions are equivalent between their corresponding classes. The two variables that presented an uneven distribution, Mode_of_Shipment and Product_importance, are detailed below:

Figure 4: Categorical Variables Distribution



From the bar plots above, we can see that the most frequently used *Mode_of_Shipment* has been Ship, while Flight and Road were significantly less used. Products of high importance also accounted the least amount in sold products. Nevertheless, regardless of modes of shipment of product importance, the proportion between late deliveries and on-time delivers are roughly similar.

2.2.2. Feature Selection:

Before proceeding to features selection and modelling, we generate dummies and drop our categorical variables, without using drop_first so that the next step: feature selection can have all features to select from and avoid loss of information by dropping additional dummies from the same feature.

As we have seen earlier in Figure 1, the distribution of *Delivery_status* is more skewed towards on time delivers, with the proportion of 6 to 4. As we want to focus on improving late deliveries, we try to balance out the training data by applying RandomOverSampler to the original dataset. This step has achieved our goal by giving us a sample of 13112 data points of equal *Delivery_status* distribution.

To pick the optimal set of features for our model, we also experiment with 4 feature scaling methods: *LogisticsRegression*, *LinearSVM*, *RandomForest*, and *SelectKBest*, resulting in 4 different feature sets. We then apply *RobustScaler* to rescale the data.

To test the feature sets' performance, we apply 3 simple classification models on each feature set and calculate average accuracy of the 3 models. The result is recorded in the table below.

Figure 5: Feature Selection Evaluation

Feature_Selection	Accuracy
SelectKBest	0.702599
RandomForest	0.694735
SelectKBest2Features	0.694735
LogisticRegression	0.683176
LinearSVM	0.682736

From this table, we see that the best performing feature set is SelectKBest with default number of features (10). However, the second-best performing model, RandomForest with only 2 features, renders only 0.8% less accurate result. To confirm the significance of those 2 features, we ran SelectKBest again setting k to 2. We verified that the 2 features chosen by RandomForest are the same as those of SelectKBest (k=2), which are *Discount_offered* and *Weight_in_gms*. This also confirms our previous observation from EDA that these two variables have the most influence on the *Delivery_status*.

As the result, we choose RandomForest feature selection set to build prediction models.

3. Model Planning and Implementation

3.1. Modelling Process Overview

To test our models' performances, we divide the sample data into training and testing data, with the proportion of 75:25. Our analysis was organized on three following steps:

- Initial tuning of hyperparameters for the models we intend to use.
- Compare performance of different model combinations for ensemble methods.
- Train and compare classification models to choose the best performing one.
- Tune an ANN and add to comparison of other models.

Project Workflow and Efficiency

Considering the trade-off between processing time and level of detail in the models, our workflow for this project focuses on maximizing accuracy. Performing the three steps above in addition to the initial feature selection analysis was very time-consuming, but we believe that these steps ensured the following qualities for our analysis:

- Feature Selection sets were tested on three different models to ensure their relevance.
- All models were tuned with an efficient set of parameters for better results.
- Ensemble methods applied previously tuned models to maximize performance.
- Ensemble methods applied a tried and tested combination of models.
- Lastly, the final model comparison considered all previously listed benefits to ensure each model could perform as well as possible.

Modelling Process and Hypothesis Testing

This modelling process was developed considering the listed hypotheses for this project, which described hypothetical impacts of features on the dependent variable. This was done by two main actions, involving identification of features that significantly impact shipping delays and how that impact occurs. The following considerations can be made about this:

- Testing our Feature Selection sets allowed us to efficiently identify the most relevant features to predict delays.
- Ensuring all models and ensemble algorithms were tuned for maximum accuracy allowed us to correctly explain how those features affect delivery time. This insight will serve as a basis for our recommendations for the company's management.

3.2. Modelling Process Steps

Hyperparameter Tuning

The first step in our analysis consisted in performing hyperparameter tuning to obtain the optimal models with the best performing configuration. For this step we still have not included Ensemble methods, since the voting algorithms will require the tuned parameters information to perform better.

To avoid overfitting, we used cross validation with 5 folds to validate our tuning results for all models except for *XGBoost*, which was tuned using *early_stopping_rounds*. The models chosen for our analysis are as follows: *LogisticRegression, LinearSVC, RbfSVC, NaiveBayes, NearestNeighbors, DecisionTree, RandomForest, AdaBOOST, XGBoost.* The following figures describe the list of hyperparameters tuned, as well as the best set of parameters established for each model:

Figure 6: Hyperparameter Tuning.

```
#list of classifiers
classifiers = [
    LogisticRegression(),
    SVC(kernel="linear", probability=True),
    SVC(probability=True),
    GaussianNB(),
    KNeighborsClassifier(),
    DecisionTreeClassifier(max_depth=5),
    RandomForestClassifier( max_leaf_nodes=16, random_state=42),
    AdaBoostClassifier(),
    XGBClassifier(use_label_encoder=False, eval_metric="error")]
```

```
#hyperparameters for cv
param_grid = [
    {'max_iter': [1000,100]},
    {'C': [0.001, 0.01, 0.1, 1, 10]},
    {'C': [0.001, 0.01, 0.1, 1, 10], 'gamma': [0.5, 1]},
    {'var_smoothing': np.logspace(0,-9,num=30)},
    {'n_neighbors': list(range(1,10)), 'p': [1,2]},
    {'min_samples_leaf': [1, 5]},
    {'n_estimators': list(range(10, 1000, 200))},
    {'learning_rate': [0.01, 0.1, 1]},
    {'early_stopping_rounds': list(range(1,20))}
]
```

Figure 7: Best Model Hyperparameters

```
{'LogisticRegression': {'max_iter': 1000},
 'LinearSVC': {'C': 0.01},
 'RbfSVC': {'C': 10, 'gamma': 1},
 'NaiveBayes': {'var_smoothing': 0.006723357536499335},
 'NearestNeighbors': {'n_neighbors': 2, 'p': 1},
 'DecisionTree': {'min_samples_leaf': 1},
 'RandomForest': {'n_estimators': 10},
 'AdaBOOST': {'learning_rate': 1},
 'XGBOOST': {'early_stopping_rounds': 14}}
```

Selecting the Best Ensemble Configurations

The next step in our analysis considered the different model combinations for ensemble methods, in order to choose the voting configuration rendering best results. To accomplish that, we experimented with 5 different sets of models using hard voting and soft voting. We ran the following voter models using 5-fold cross validation and compared the results in the end:

- All classifier models with tuned hyperparameters
- 5 and 3 Classifier models with tuned hyperparameters of highest individual accuracy
- 5 and 3 Simplest classifier models with tuned hyperparameters

Figure 8 lists the configurations tried for the voter models and Figure 9 display the ones rendering best results:

Figure 8: Tuning Ensemble Configuration

```
#hyperparameters for cv
voters = [
       [('lr', lr), ('linear_svc', linear_svc), ('rbf_svc', rbf_svc),('gnbs', gnbs), ('knn', knn), ('dtc', dtc),
                    ('rf', rf), ('ada', ada), ('xgb', xgb)],
       [('rbf_svc', rbf_svc), ('rf', rf), ('ada', ada), ('xgb', xgb), ('dtc', dtc)],
       [('rbf_svc', rbf_svc), ('rf', rf), ('ada', ada)],
       [('lr', lr), ('linear_svc', linear_svc), ('gnbs', gnbs), ('knn', knn), ('dtc', dtc)],
       [('lr', lr), ('linear_svc', linear_svc), ('gnbs', gnbs)],
                  ]
```

Figure 9: Best Ensemble Configuration

```
Hard Voting:
Best Voters: [('rbf_svc', SVC(C=10, gamma=1, probability=True)), ('rf', RandomForestClassifier(max_leaf_nodes=16, n_estimators=
10, random_state=42)), ('ada', AdaBoostClassifier(learning_rate=1))]
Best Accuracy: 0.7177164309789037

Soft Voting:
Best Voters: [('rbf_svc', SVC(C=10, gamma=1, probability=True)), ('rf', RandomForestClassifier(max_leaf_nodes=16, n_estimators=
10, random_state=42)), ('ada', AdaBoostClassifier(learning_rate=1))]
Best Accuracy: 0.7177164309789037
```

Comparing the Tuned Models and Ensemble Methods

After determining the best sets of parameters for models and voting configurations, to compare their performances. For this step, we also included a Multilayer Perceptron tuned using 5-fold cv on the following set of parameters:

Figure 10: MLP Classifier Tuning

```
parameters={
'learning_rate_init': [0.1, 0.01, 0.001],
'hidden_layer_sizes': [(250,),(100,),(30,),
                       (250,125), (100,50), (30,15)],
'alpha': 10.0 ** -np.arange(1, 3),
'activation': ["relu", "Tanh"],
'max_iter': [500, 1000]
}
```

4. Result Interpretation and Implications

4.1. Feature Selection Results

To recap, we have tried four different feature selection methods and evaluated the sets' performance by running simple classification models and comparing the average accuracy scores. From this experiment, we found that the features *Discount_offered* and *Weight_in_gms* are the most significant to predict the dependent variable, with the marginal benefit of adding new features being too low to justify their contribution. This result is supported by previous evidence found on EDA, which showed a high correlation of these two features with the target variable.

4.2. Model Results

We experimented with 11 models on the 2 *RandomForest* selected features scaled with *RobustScaler*. The results obtained are listed in the following table, containing information about the classifier, accuracy on the testing set, used parameters, and feature selection set:

Figure 11: Model Results Comparison

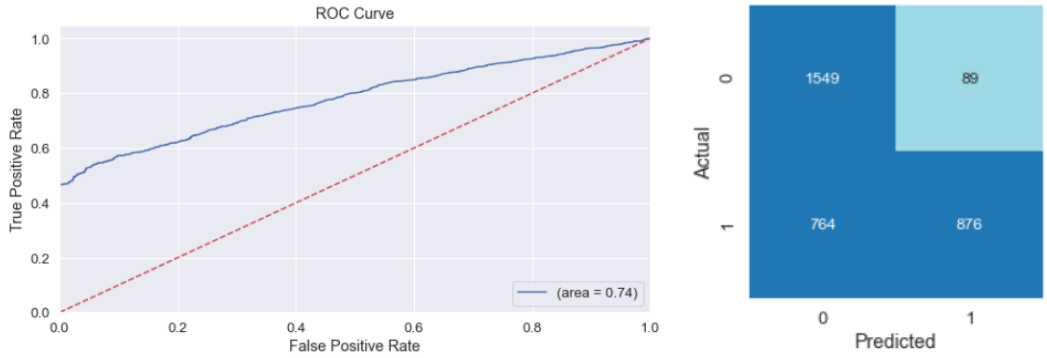
Classifier	Accuracy	Params	Feature_Selection
XGBOOST	0.739780	{'early_stopping_rounds': 14}	RandomForest
NearestNeighbors	0.735815	{'n_neighbors': 2, 'p': 1}	RandomForest
RandomForest	0.732459	{'n_estimators': 10}	RandomForest
HardVoting	0.731544	Voters: RandomForest, RBF SVC, AdaBOOST	RandomForest
MLPClassifier	0.730018	{'activation': 'relu', 'alpha': 0.01, 'hidden_...	RandomForest
SoftVoting	0.729713	Voters: RandomForest, RBF SVC, AdaBOOST	RandomForest
AdaBOOST	0.729103	{'learning_rate': 1}	RandomForest
DecisionTree	0.724527	{'min_samples_leaf': 1}	RandomForest
RbfSVC	0.709274	{'C': 10, 'gamma': 1}	RandomForest
NaiveBayes	0.706528	{'var_smoothing': 0.006723357536499335}	RandomForest
LinearSVC	0.702868	{'C': 0.01}	RandomForest
LogisticRegression	0.685479	{'max_iter': 1000}	RandomForest

We can see from the table that *XGBoost* presented the highest accuracy between the models (73.97%), with early_stopping_rounds set to 14.

4.3. Result Assessment

XGBoost Model Evaluation

Figure 12: Confusion Matrix and ROC Curve



The Confusion Matrix shows that our model evaluates actual delays accurately but renders a large amount of false negative predictions of deliveries on time. Since our focus for the analysis is correctly identifying delayed deliveries, there results do not pose a large issue for us, since the model will still accurately describe actual delayed deliveries. This effect can also be seen in steep initial elevation of the ROC curve. Lastly, the Area Under Curve for the ROC has a reasonable value of 0.74.

Figure 13: Classification Report

	precision	recall	f1-score	support
0	0.67	0.95	0.78	1638
1	0.91	0.53	0.67	1640
accuracy			0.74	3278
macro avg	0.79	0.74	0.73	3278
weighted avg	0.79	0.74	0.73	3278

The classification report shows a high recall score (0.95) and relatively high f1-score (0.78) for predicting actual delays, indicating that the model is efficient to solve the company's problem of identifying this type of delivery issue.

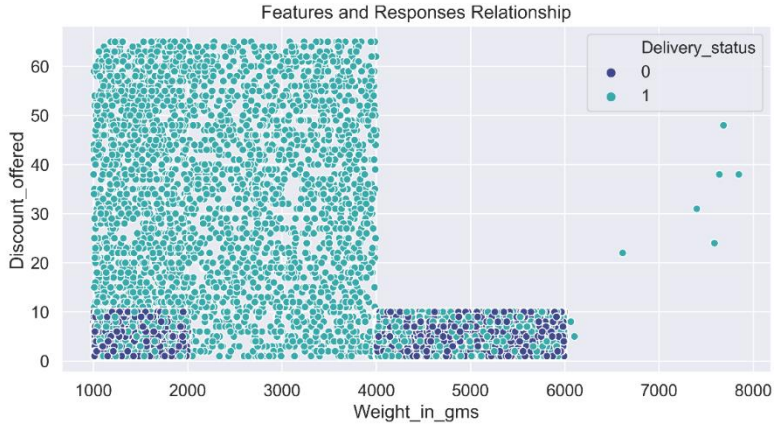
4.3.1. Model Validity Considerations

We have established that the model can accurately identify delayed deliveries, which is the core issue targeted by the company in this project. However, the fact that this is performed using only the features *Weight_in_gms* and *Discount_offered* presents some issues when considering domain logic. This section will explore the issues we found and later discuss if they impact the overall validity of the model for the goals of this project.

Model Features and Domain Knowledge

Considering the validity of the model to our domain, the main issue that arises is the sole use of *Weight_in_gms* and *Discount_offered* to predict *Delivery_status*. Although the modelling supports this configuration, it may not be consisted with domain knowledge regarding shipment. Additionally, the fact that only two variables are being used may indicate a simpler problem that would not require machine learning. To address both concerns, we will present the relationship of those 2 variables with *Delivery_status* and attempt to explain the rationale behind it.

Figure 14: Relationship Between Discount_offered, Weight_in_gms, and Delivery_status



As shown in the chart below, all orders with delayed delivered presented a *Discount_offered* value below 10 percent. Additionally, weights below 2000 grams or above 4000 grams were also the main precursors for delayed deliveries.

Another interesting observation is that orders within those thresholds were not predominantly delays, with some of them being delivered on time. With this observation, we can start to address the concern of the simplicity of a problem with just two independent variables.

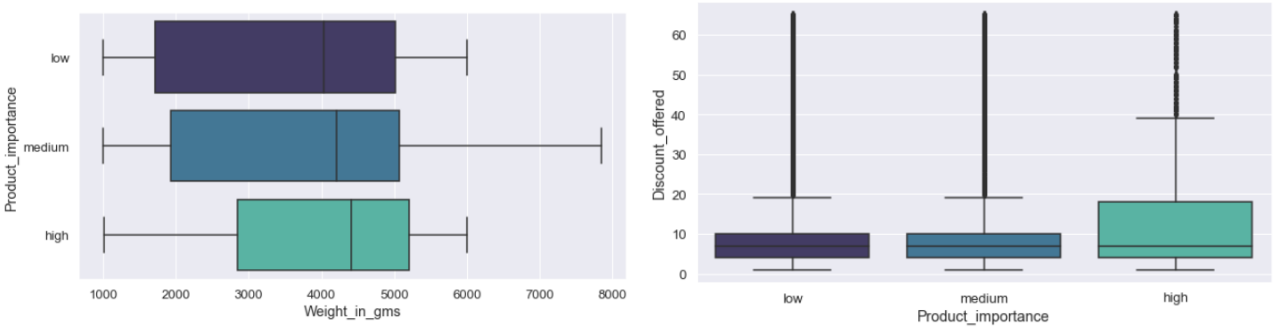
Problem Simplicity with Two Independent Variables

We can infer that, although the classification task with two independent variables is simpler, it still presents a probabilistic problem that benefits from using Machine Learning. While the simple thresholds of these two variables can be used to flag any potentially delayed order, the fact that we are providing an accurate model that can analyze each case by their specific value and render probabilistic outputs makes this process more efficient and precise.

Meaning of Variables on Domain Knowledge

To address the second issue of relating those variables with domain knowledge, we should first consider how those two aspects may affect delayed deliveries. We believe that low discount rates will typically relate to less deliveries that receive less priority than others, considering that high rates indicate a particular relevancy of a given order for the client company. Additionally, we believe that the deliveries with weight below 2000 or above 4000 present delays due to intricacies of the delivery process for that type of merchandise, while heavier products may also indicate more relevant orders for the company. To illustrate those two points, the following relationships were identified between the selected features and product importance:

Figure 15: Relationship Between Selected Variables and Product Importance



The charts above seem to corroborate our hypothesis, indicating that orders with higher discount rates and heavier weights are more predominantly linked to higher importance products, specially in the case of the discounts. Although this still indicates an indirect effect to our model, it aids us in making sense of our findings when considering domain knowledge.

Potential Improvements and Additional Data Benefits

A final consideration to be made here is that additional inputs to the model would be beneficial, particularly by aggregating data related to the shipment contracts and specifications with the supplier for each order. We believe that this would aid the models in the interpretation of why those features ultimately impact delivery. We hypothesize that orders with low discount and "problematic" weight ranges translate to particular shipping contract specifications that reflect a lower priority by the shipment providers. Therefore, including this sort of data would further help improve the

results. Lastly, we believe the variety models implemented is already exhaustive, and what is missing to improve results is additional information/data.

Consistency with Project Goals

Despite the previously mentioned problems, when we consider the prediction accuracy for actual negatives (delayed deliveries), we can infer that the model correctly addresses the problem posed by the company. This indicates that we can correctly direct managers of the company to target potential delivery delays. Additionally, since the number of false positives is relatively small, this model will not fail to identify delayed deliveries, which would pose an issue where troublesome orders would not be targeted by management.

Considering this, we do believe that the model is sufficient for the goal of the current project. The issues identified could be addressed and improved by aggregating new data, but we believe that our model can be used to correctly flag problematic deliveries for managers to act on.

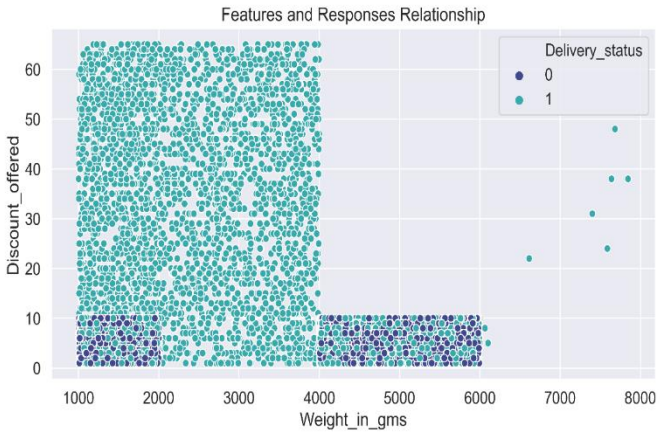
5. Out of Sample Predictions

We generate 4 new data points to emulate data with the purpose of demonstrating our model prediction capability. To perform that, we established new data illustrating a combination of scenarios for weight and discount. The table below shows the values for each feature and the result outputted by our *XGBoost* on scaled the data.

Figure 16: Model Predictions

	Discount_offered	Weight_in_gms	Predicted_Delivery_status
0	0	1100	0
1	70	1100	1
2	0	5000	0
3	70	5000	1

We can see that orders 1 and 3 are predicted to be delivered on time, while order 0 and 2 have estimated late deliveries and should receive extra attention. This is consistent with the previously analyzed scatterplot to the right, indicating that the orders with low discount and with weight located in the extremes of the spectrum have potential to be delayed.



These findings are in line with the observations made when assessing model results, indicating that our model can successfully be used in the future to flag potentially problematic deliveries and trigger management action to mitigate problems.

6. Concluding remarks

Using the provided dataset of shipping information, our team has successfully built a predictive model to predict the delivery status with a high accuracy, especially in identifying potential order delays. By using statistical techniques and machine learning algorithms, we identify two main features that have the most impact on delivery status, which are *Discount_offered* and *Weight_in_gms*. Although these features may pose some questions about model simplicity and domain knowledge, which we addressed on the item 4 of this report, the fact remains that our model still correctly and efficiently accomplishes the goal of identifying potential delays for the company.

We have manifested our hypothesis that including more data related to delivery specifications and contract with the shipping suppliers would help us to further improve the models. However, we strongly believe that the current findings already suffice to help direct managerial efforts from our client to improve their shipping process.