ÖPTICYCLE

Technical report

S. Abdian[1], N. Chakrabarty[1], E. Honarvar[2], J. Mwende[1], R. Sadeghimehr[1], G. Walravens[2], and M. Zelina[2]

[1]Jheronimus Academy of Data Science
[2]Eindhoven University of Technology

TU/e x IKEA x JADS

July 17, 2023

Contents

1 Problem Background **3**

2 Goals and Objectives **3**

3 Data **3**

4 Methodology **5**
- 4.1 Data Extraction 5
- 4.2 Exploratory Data Analysis 6
- 4.3 Order-based Univariate & Bivariate Analysis 6
  - 4.3.1 Univariate Analysis 6
  - 4.3.2 Bivariate Analysis 9
- 4.4 Investigative Analysis of the activities within orders 11
- 4.5 Non-activity-based Approach 13
  - 4.5.1 Feature Engineering 14
  - 4.5.2 Modelling 15
- 4.6 Activity-based Approach 16
  - 4.6.1 Feature Engineering 16
  - 4.6.2 Modelling 18

5 Results **18**
- 5.1 Non-activity-based Predictive Model 18
  - 5.1.1 Model Evaluation 18
  - 5.1.2 Model Interpretation: Top reasons of Imperfection 19
- 5.2 Activity-based Predictive Model 21
  - 5.2.1 Model Evaluation 21
  - 5.2.2 Model Interpretation: Top reasons of Imperfection 22

6 Conclusions **29**

7 Recommendations **29**

8 Future Work **29**

1 Problem Background

IKEA, a renowned multinational retailer, operates a complex sequence of processes and activities within their Order Life Cycle (OLC), encompassing various stages from order placement to fulfillment and delivery to the customer. However, the orders within this complicated system are prone to imperfections which could result in dissatisfied customers, lost sales, and high costs for IKEA.

Order imperfections, as defined by the company, encompass any deviations, errors, or issues occurring within the order processing system that result in disparities between customer expectations and the actual delivery of products. Order imperfections can stem from different activities related to multiple sources within the OLC. These include errors in internal processes such as packaging and delivery, supplier-related issues like supply chain delays and product defects, and customer-related factors such as incorrect order specifications and changes in preferences. These imperfections lead to customer dissatisfaction, additional costs for IKEA, and the need for extra time and resources to rectify the situation.

To mitigate the negative impact of order imperfections, there is an urgent need for IKEA to estimate the likelihood of imperfections occurring at various stages of the OLC. More precisely, the company plans to identify and analyze the most significant reasons behind these imperfections, thereby enabling them to gain insights into the key contributing factors. By detecting the crucial attributes or patterns typical of imperfection orders, they hope to analyze the process and take preventative action. Specifically, the solution should provide the company with the ability to recognize and predict possible order imperfections based on the available order lifecycle logs.

2 Goals and Objectives

Our project's primary goal is to design a method/tool to estimate the likelihood of order imperfections based on the available order life-cycle logs and detect the most important reasons for the imperfections in the Finland market. Our objectives to reaching the goal are:

1. Perform Exploratory Data Analysis (EDA) on the order lifecycle data.
2. Develop and evaluate a predictive process mining model
3. Design an explanation method for the prediction model

3 Data

The IKEA-INGKA Team maintains the Order Database following an OLAP Architecture. The schema of the OLAP Order Database is shown in Figure 1.

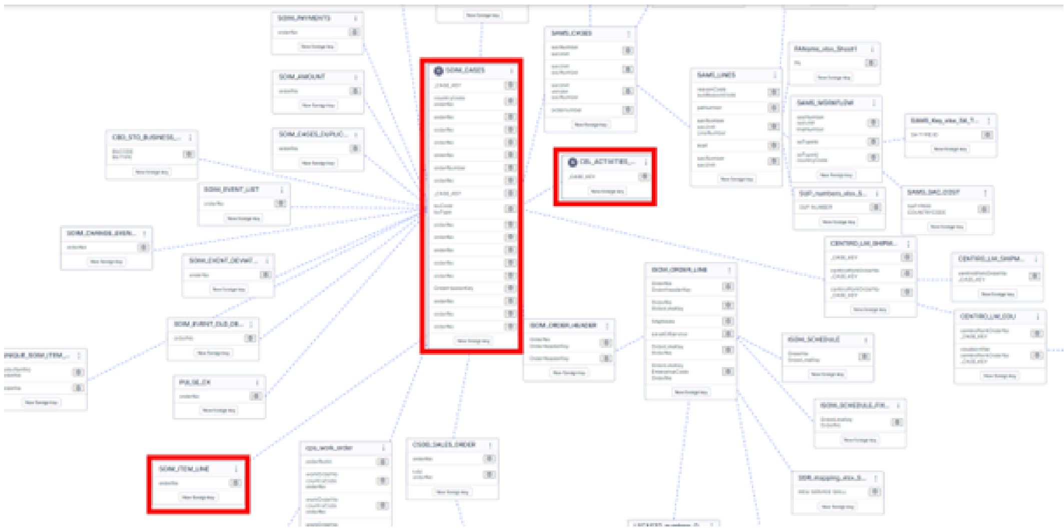


Figure 1: Order Database in OLAP Architecture

In Figure 1, the following tables (in bounding boxes) and table attributes were considered:

1. **SOIM_CASES**: Each instance of this table represents a unique order with the following considered order attributes with corresponding data-types shown in Table 1.

Attribute Name	Feature Type	Description
buCode	Categorical	Business Unit Code (Store Number)
buType	Categorical	Business Unit Type (STO as in Store)
countryCode	Categorical	Country processing the order
createDateTime	Date & Time	Creation date and time of the order
orderType	Categorical	Specific type of order
salesSetId	Categorical	Product group identifier
_CASE_KEY	Categorical	Unique Order identifier

Table 1: Attribute details of SOIM_CASES

2. **SOIM_ITEM_LINE**: Each instance of this table represents the quantities of the products ordered in every order. Table 2 summarizes all the attributes considered in SOIM_ITEM_LINE.

Attribute Name	Feature Type	Description
_CASE_KEY	Categorical	Unique Order identifier
Quantity	Continuous	Number/units of products ordered

Table 2: Attribute details of SOIM_ITEM_LINE

3. **CEL_ACTIVITIES_SOM**: Each instance of this table represents an activity in the lifecycle process of an order. The attributes considered are tabulated in Table 3.

Attribute Name	Feature Type	Description
_CASE_KEY	Categorical	Unique Order identifier
ACTIVITY_EN	Categorical	Activity Name
SYSTEM	Categorical	System of origin of the activity
EVENTTIME	Date & Time	Date and time of the activity occurrence

Table 3: Attribute details of CEL_ACTIVITIES_SOM

Apart from the OLAP Order Database, we got access to a separate a table containing the list of imperfect activities in the presence of which in the Order Lifecycle, the entire order becomes imperfect.
This table is called the **IMPERFECT_ACTIVITIES** table and the attributes considered are tabulated in Table 4:

Attribute Name	Feature Type	Description
Activity	Categorical	Activity Name
Imperfect Group	Categorical	Imperfect Activity Group Name

Table 4: Attribute details of IMPERFECT_ACTIVITIES

4 Methodology

4.1 Data Extraction

The order data for Finland (countryCode: FI) from the three tables mentioned in Section 3: SOIM_CASES, SOIM_ITEM_LINE, CEL_ACTIVITIES_SOM were extracted using Process Query Language (PQL) as the Data Pool inside the Celonis ML Workbench. In the data extraction process, **_CASE_KEY** served as the Primary Key for the SOIM_CASES table, and Foreign Keys for SOIM_ITEM_LINE and

CEL_ACTIVITIES_SOM and countryCode in the SOIM_CASES table was set to FI i.e., Finland.
The total quantity for each order is calculated by summing up the quantities by each unique _CASE_KEY.
Approximately 1.8 million unique orders were extracted in the duration from June 1, 2021 to June 4, 2023.

4.2 Exploratory Data Analysis

The EDA performed on the order lifecycle data is bifurcated into two broad approaches:

1. Order-based Univariate & Bivariate Analysis

2. Investigative Analysis of the activities within orders

4.3 Order-based Univariate & Bivariate Analysis

4.3.1 Univariate Analysis

The Univariate Analysis involves the study and computation of the number of unique orders in every category of the categorical attributes.

- buCode: Number of orders processed by each IKEA store in Finland was computed and visualized in Figure 2.

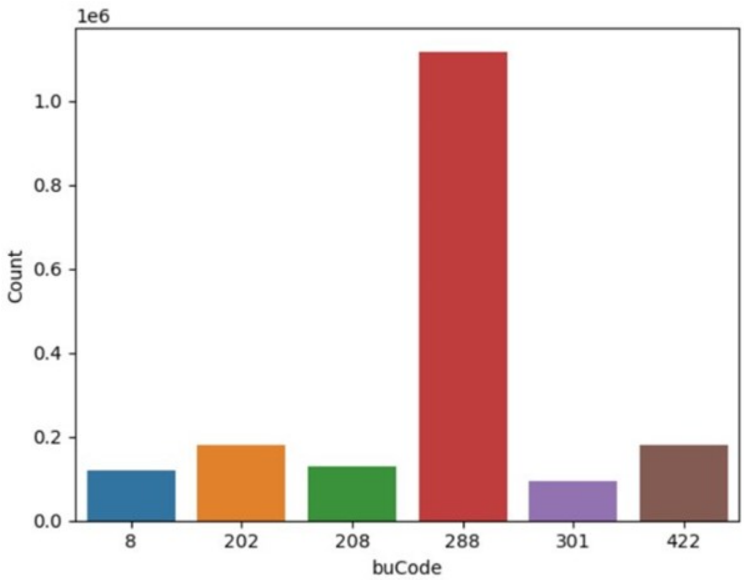
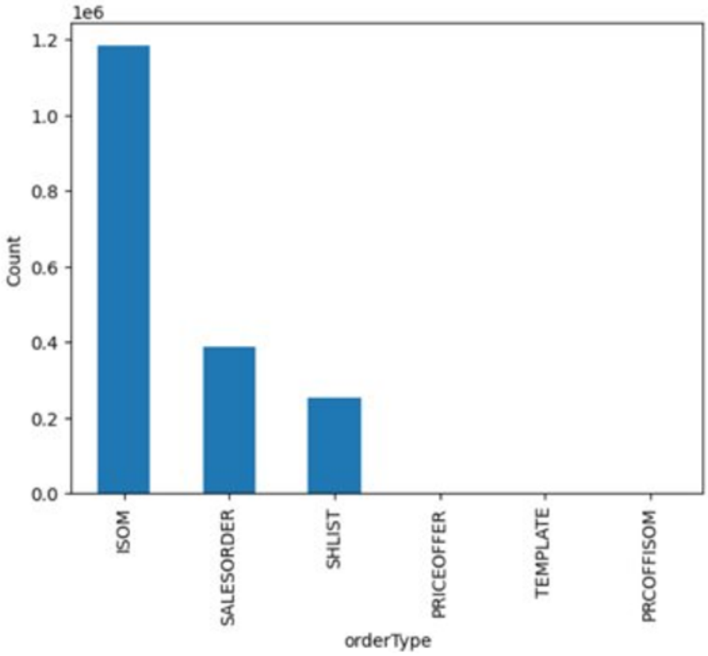


Figure 2: Number of orders processed in each store

From the analysis in Figure 2, store 288 has the highest number of orders processed and store 301 has the least. But, after the meeting with the stakeholders, we figured out that store 288 is an online store and the rest of the stores, 8, 202, 208, 301, and 422 are physical stores.

- buType: It was found that the order data are only for stores i.e., STO. So, buType was dropped from the analysis.
- orderType: Number of orders of each type was computed and visualized in Figure 3.




```
ISOM          1183874
SALESORDER     385902
SHLIST         251052
PRICEOFFER        214
TEMPLATE           19
PRCOFFISOM          2
Name: orderType, dtype: int64
```

Figure 3: Number of orders processed of each type

Based on the IMPERFECT_ACTIVITIES table, the orders are labelled as Perfect or Imperfect based on the logic that if an imperfect activity occurs at least once in the lifecycle of a order, then the order is labelled as imperfect and perfect otherwise. So, a class attribute is created and named as **IMPERFECT** that can assume values 0 (if the order is perfect) and 1 (if the order is imperfect).

- IMPERFECT: The class distribution i.e., the % distribution of perfect and imperfect orders is computed and visualized in Figure 4.

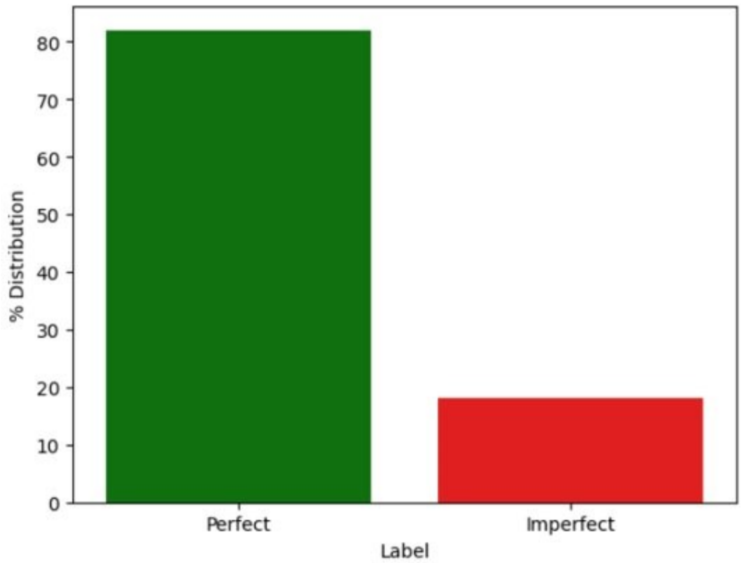


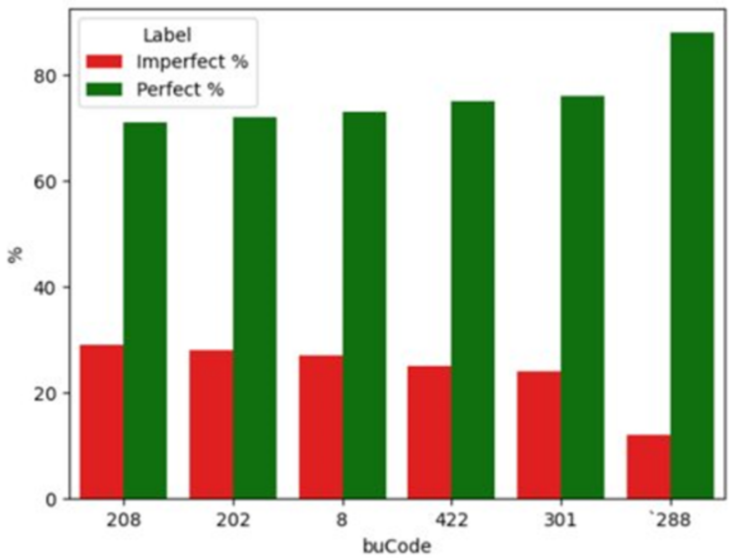
Figure 4: Class Distribution (in %) of Perfect and Imperfect orders

From the analysis, we concluded that 18% of the total number of orders are imperfect, and the remaining 82% are perfect orders.

4.3.2 Bivariate Analysis

The Bivariate Analysis involves the study of the relationship between the independent attributes (buCode and orderType) with the dependent attribute, IMPERFECT. This analysis highlights the impact of the variables, buCode and orderType on order imperfection.

- buCode with IMPERFECT: % of imperfect and perfect orders for each store number are computed and visualized in Figure 5.

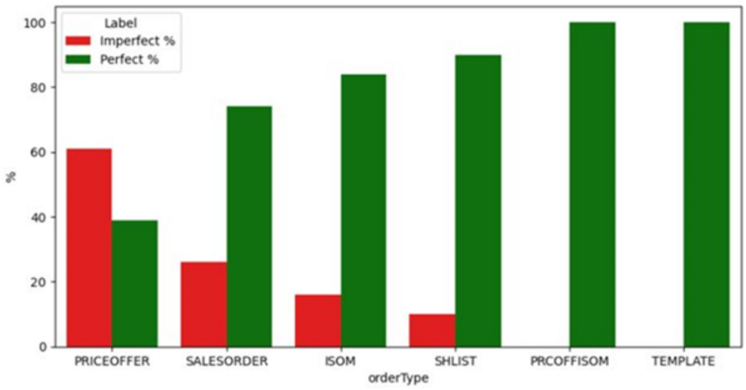



```
buCode
208    28.969512
202    28.254680
8      27.438673
422    25.683018
301    24.655980
288    12.180369
Name: IMPERFECT, dtype: float64
```

Figure 5: Percentage of imperfect and perfect orders in every processing store

From the analysis, it can be concluded that Store 208 has the highest % of imperfect orders i.e., 29% (approx.) followed by 202, 8, 422, and 301, each higher than 18% which is the percentage of imperfect orders overall.
The % of imperfect orders for the online store, 288 is the least i.e., 12%. This infers that orders processed via online stores are mostly perfect.

- orderType with IMPERFECT: % of imperfect and perfect orders for each order type are computed and visualized in Figure 6.




```
orderType
PRICEOFFER     61.214953
SALESORDER     26.662728
ISOM           16.607004
SHLIST         10.824451
PRCOFFISOM      0.000000
TEMPLATE        0.000000
Name: IMPERFECT, dtype: float64
```

Figure 6: % of imperfect and perfect orders in every order type

From the analysis, PRICEOFFER has an alarming % of imperfect orders i.e., 61% whilst order types like PRCOFFISOM and TEMPLATE have no imperfect orders at all.
Following this, we got a recommendation from the stakeholder to focus on the order types of ISOM and SALESORDER only as part of their requirements post this analysis.

4.4 Investigative Analysis of the activities within orders

It is important to have event-time to discover the sequence of the activities based on time. In this regard, we have explored the sequence of activities, identified the first position of imperfect orders, and thoroughly investigated the initial and most imperfect activities.

- Sequence of Activity: The sequence of activities based on event-time for a specific Case Key is visualized in Figure 8. As shown in this Figure, different activities are created at the same time, so we chose a sequence based on indexes to handle this. Figure 8 demonstrates the analysis in a manner that highlights instances where a single activity has been created for a particular order but with different event times.

	_CASE_KEY	ACTIVITY_EN	SYSTEM	SORTING	EVENTTIME
0	1015424538A01202	SALES ORDER CREATED	iSELL	1	2022-12-12 12:23:38
1	1015424538A01202	SALES ORDER CONVERTED	iSELL	10	2022-12-12 12:23:38
2	1015424538A01202	CC STATUS CHANGED	iSELL	12	2022-12-12 22:30:12
3	1015424538A01202	SALES ORDER CANCELED	iSELL	14	2022-12-12 22:30:12
4	1015424538A01202	CC STATUS CHANGED	iSELL	18	2022-12-12 22:30:12

Figure 7: Different Sequence of activities for one Case Key with the Same Event-Time

	_CASE_KEY	ACTIVITY_EN	SYSTEM	SORTING	EVENTTIME
142	1058948332A01288	CREATED	ISOM	100	2022-09-30 10:39:47
143	1058948332A01288	CREATED	ISOM	100	2022-09-30 10:40:07

Figure 8: Same Sequence of activities for one Case Key with Different Event-Time

- First Position of Imperfect Order: The first position of imperfect orders refers to determining the percentage of orders with imperfections and identifying the position of the initial imperfect activity within those orders. Based on the analysis presented in Figure 9, we have concluded that the majority of imperfect activities tend to occur at the second position within an order.

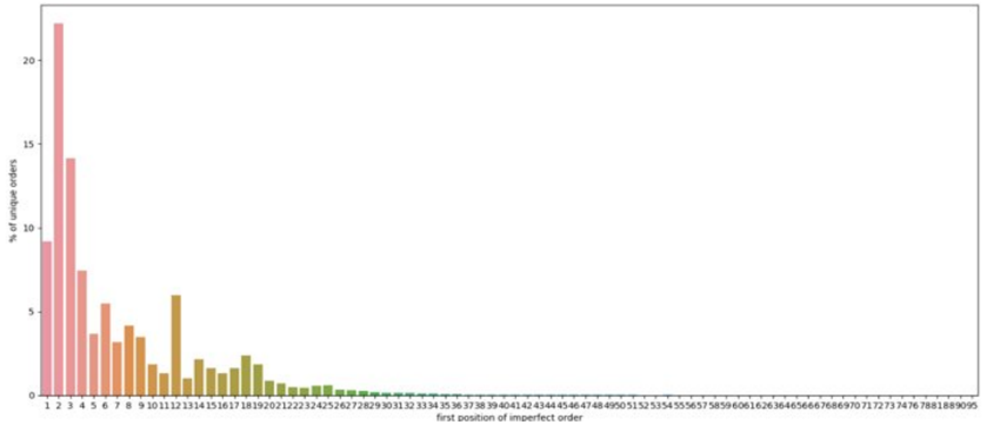


Figure 9: Percentage count of unique IMPERFECT orders with different positions of occurrence of the first imperfect activity

- First Most Imperfect Activity: The first most occurred imperfect activity in an order visualized in Figure 10. This Figure shows the count of the first most frequent imperfect activity that occurs in an Order Lifecycle of an Imperfect Order. On the right-hand side is the graphical presentation of the table and the count of the first most frequent imperfect activity.

	Imperfect Activity	# First Imperfect Position 1
0	SAC CREATED	18790
1	LCD STATUS CHANGED	248
5	CC STATUS CHANGED	11
3	DELIVERY PRICE CHANGED	4
2	PRICE CHANGED	1
4	BACKORDERED	1
6	FREIGHT AMOUNT CHANGED	1
7	SALES ORDER BLOCKED	1
8	SERVICE PRODUCT DATETIME CHANGED	1

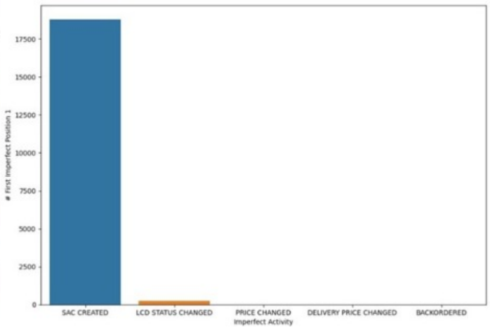


Figure 10: The first most frequent Imperfect Activity that occurs in an Order Lifecycle

4.5 Non-activity-based Approach

In the non-activity-based approach, we utilized order-based features to predict the likelihood of order imperfection. After conducting exploratory data analysis (EDA), we identified two categorical attributes, namely "Bucode" and "OrderType," as the only usable features for training our model. Figure 11 shows the overall view of the non-activity-based approach.

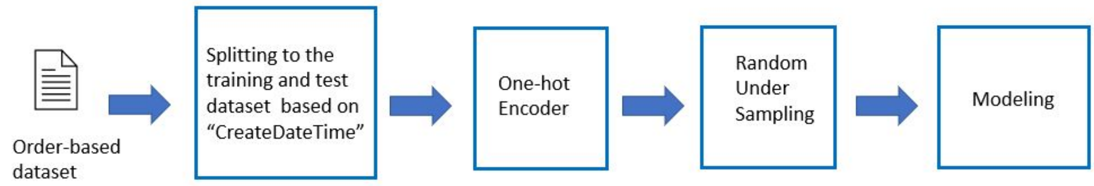


Figure 11: The Overall View of the Non-activity-based Approach

4.5.1 Feature Engineering

Non-activity-based feature engineering involves the following three main steps.

1. Splitting into the training and test dataset: We divided our dataset into training and test sets based on the "CreateDateTime" column to ensure reliable model evaluation. The training dataset comprised orders created between June 1, 2021 and April 30, 2023. The test dataset consisted of orders created from May 1, 2023 onwards.

2. Categorical Feature Encoding: Given that our chosen features were categorical in nature, we utilized the one-hot encoding technique to convert them into numerical values, ensuring compatibility with our training and test datasets. Notably, the "buCode" feature encompassed six distinct store numbers. However, based on stakeholder feedback, we refined the feature selection and retained only the "ISOM" and "SALES ORDER" types from the "OrderType" attribute. After applying the one-hot encoder, our feature set expanded to include a total of eight features. This encoding process enabled us to represent categorical information in a numerical format, facilitating accurate analysis and modeling of the selected attributes.

3. Addressing Class Imbalance: Applying Random Undersampling Technique
   While conducting exploratory data analysis (EDA), we uncovered a class imbalance concern between perfect and imperfect orders within the training dataset. This imbalance posed a risk of introducing bias into our model predictions. To mitigate this issue, we employed the Random Undersampling technique. Figure

12 illustrates the distribution of classes before and after undersampling, demonstrating the successful achievement of a balanced representation of perfect and imperfect orders in the training data.
By under-sampling the majority class, which, in this case consisted of perfect orders, we ensured an equal representation of both perfect and imperfect orders in the training data. This strategic reduction in the number of instances from the majority class helped alleviate the bias that may arise during model training.

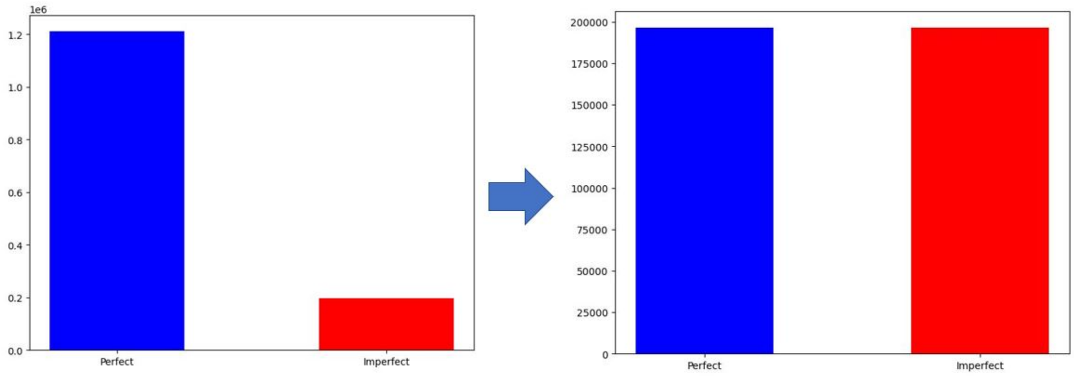


Figure 12: Class Distribution Before and After Random Undersampling

4.5.2 Modelling

To predict the likelihood of order imperfection considering only order attributes, we trained four distinct models using the pre-processed dataset. We initiated the model selection process with logistic regression which is a basic classification model to establish a baseline. After that, we experimented with a tree-based modelling approach using decision tree. Finally, we progressed to ensemble-based modelling approach involving several decision trees. In the ensemble modelling approach, we tried out two kinds of ensembles:

1. Bagging using Random Forest
2. Boosting using Extreme Gradient Boost (XGBoost)

4.6 Activity-based Approach

The second approach involves predicting order imperfection based on not only order-based data but also activity-based information. While we followed the same steps described in the previous section for order-based features, we merged the randomly Undersampled data with the activity-based dataset using the _CASE_KEY identifier. The figure 13 shows the overall view of the activity-based approach.

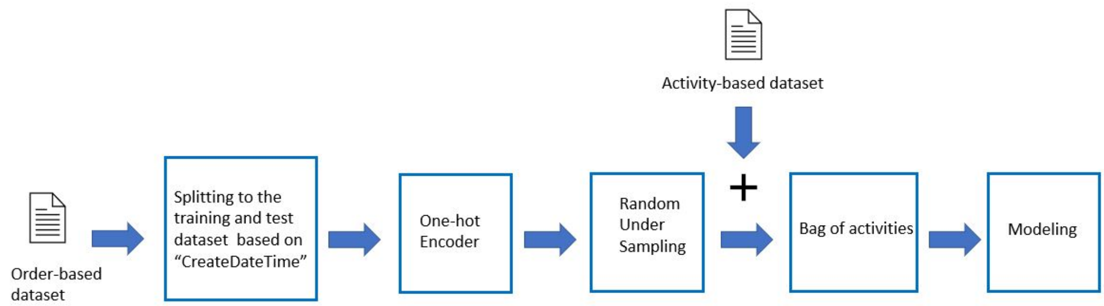


Figure 13: The Overall View of the Activity-based Approach

4.6.1 Feature Engineering

To build a model capable of predicting imperfections, we trained it solely on non-imperfect activities. To achieve this, we employed the following approach:

1. Creation of the New Training Dataset (Bag-of-Activities): For the purpose of training our model exclusively on non-imperfect activities, we adopted a bag-of-activity approach. This involved creating a new training dataset where the activities themselves serve as features. To obtain a comprehensive set of non-imperfect activities for modeling, we excluded all activities associated with imperfect orders from the complete set of activities. The resulting set of activities, devoid of imperfections, was then utilized as the new features for training our predictive model. The non-imperfect activities as features in the training set contribute to an order being perfect or imperfect.

2. Truncation of Activity Sequences: In our analysis, we identified the specific point within each order life-cycle where the first occurrence of an imperfect activity was observed. Subsequently, we performed truncation on the activity sequence, eliminating all activities that followed the first imperfect activity.

This process allowed us to retain only the non-imperfect activities that preceded the occurrence of imperfections in each order. This truncation ensured that the activities as features do not have any imperfect activity which can lead to data leakage.

3. Calculation of Activity Occurrence: In order to represent the non-imperfect activities as features in our model, we adopted a strategy where the number of occurrences of each non-imperfect activity was assigned as the corresponding feature value. This approach allowed us to quantify the frequency of each non-imperfect activity within the order sequence and utilize it as a numerical representation for training our model. By capturing the occurrence counts of these activities, we were able to encode valuable information about their relevance and contribution to order imperfection likelihood. This feature engineering technique enhanced the model's ability to capture patterns and relationships between non-imperfect activities, leading to a deeper understanding of the factors influencing order imperfection.

To provide a clearer explanation, consider an example with two different orders and their corresponding sequences of activities, where an asterisk (*) denotes imperfect activities.

Order 1: A1, A2, A3, A2, A4*, A6, A5*, A7

Order 2: A1, A7, A6, A2, A4*, A1

- Step 1: Calculation of New Features
  To obtain the new features, we exclude the set of imperfect activities from the complete set of activities:
  Features = All activities - Imperfect activities
  Features = (A1, A2, A3, A4, A5, A6, A7) - (A4, A5)
  Features = (A1, A2, A3, A6, A7)

- Step 2: Truncation of Activity Sequences
  In this step, we identify the point within each order life-cycle where the first imperfect activity occurs. Subsequently, we truncate the sequence of activities, retaining only the non-imperfect activities that precede the occurrence of imperfections. After truncation, the activity sequences are modified as follows:
  Order 1: A1, A2, A3, A2
  Order 2: A1, A7, A6, A2

- Step 3: Calculation of Activity Occurrence To represent the non-imperfect activities as features, we assign the number of occurrences of each activity as the corresponding feature value. This information is summarized in a resulting data frame, as shown in figure 14.

Bag-of-activities	A1	A2	A3	A6	A7
Order 1	1	2	1	0	0
Order 2	1	1	0	1	1

Figure 14: An Illustrative Dataframe Resulting from the Bag-of-Activities Approach

In the resulting data frame, each row represents an order, and the columns correspond to the non-imperfect activities. The values in each cell indicate the number of occurrences of the corresponding activity within the order life cycle. By capturing the occurrence counts of each activity, we can leverage this information as feature values in training our model.

4.6.2 Modelling

Following the same strategy as discussed in section 4.5.2, we trained four distinct models: Logistic Regression, Decision Tree, Random Forest, and XGBoost.

5 Results

5.1 Non-activity-based Predictive Model

5.1.1 Model Evaluation

The trained models discussed in Section 4, considering the features buCode and orderType were evaluated on the basis Accuracy, Precision, Recall, and F1-Score. The results of the model performance analysis are tabulated in Table 5. From Table 5, it

Non-activity-based Model Performance Analysis				
Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	**0.73**	**0.32**	**0.58**	**0.41**
Decision Tree	0.73	0.32	0.58	0.41
Random Forest	0.73	0.32	0.58	0.41
XGBOOST	0.73	0.32	0.58	0.41

Table 5: Performance Analysis of Non-activity-based Model

is evident that all the four models have the same performance. So, according to the Principle of Model Selection, Logistic Regression is chosen as the best model as it is the simplest model among the four, yielding the best performance.
The Logistic Regression Model has an accuracy of 73% which means out of 100 orders, it is correctly predicting the imperfection/perfection of 73 orders.
The Model has a Precision of 32% which means out of 100 predicted imperfect orders, 32 are correctly predicted as imperfect.
The Model has a Recall of 58% which means out of 100 actually imperfect orders, 58 are correctly predicted as imperfect.
As F1-Score is the Harmonic Mean of Precision and Recall, it is given the most emphasis among the four evaluation metrics.

5.1.2 Model Interpretation: Top reasons of Imperfection

After evaluating the model and selecting the best model (Logistic Regression), we brought out the most important order features responsible for the model to estimate a certain likelihood of imperfection. In order to accomplish this, we used SHapely Additive exPlanations (SHAP) for model interpretation. The SHAP Linear Explainer is used for explaining the Logistic Regression Model, to bring out the main features, and their contributions towards the model in estimating the likelihood of imperfection. We performed the model interpretation, taking two sample orders (_CASE_KEYS):

- **_CASE_KEY: 1358405196A01422 (it is actually an imperfect order)**
  The force plot of the SHAP Linear Explainer for this case key is shown in Figure 15.

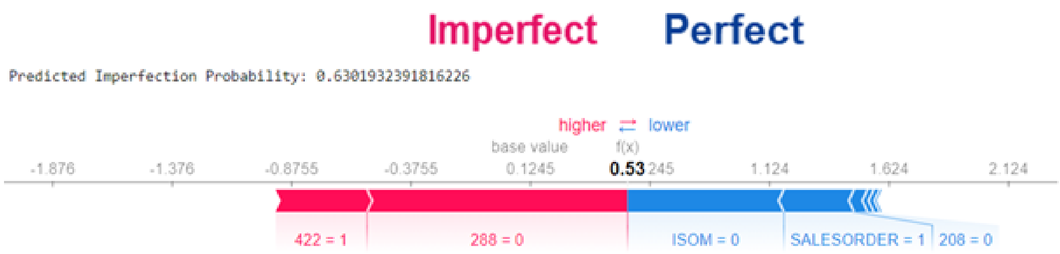


Figure 15: SHAP Linear Explainer for 1358405196A01422

From Figure 15, it can be interpreted that the model estimates that the order, 1358405196A01422 has a likelihood of imperfection to be 0.63 and the net contribution of features towards imperfection is more than that of perfection.
In order to simplify the interpretation of the force plot, we generated a tabular heatmap, shown in Figure Figure 16.

	FeatureValue	Contribution
2	288 = 0	1.090215
4	422 = 1	0.382215

Figure 16: Tabular Heatmap derived from SHAP Linear Explainer for 1358405196A01422

Hence, the top imperfection reasons (order attributes) responsible for the high likelihood of imperfection (0.63) are as follows:

1. The order was not processed at the online store, 288, which had a very high perfection rate, confirmed by the EDA discussed in Section 4.
2. The order was processed at the physical store, 422 which, indeed has a high imperfection rate, obtained in the EDA discussed in Section 4.

- **_CASE_KEY: 1345897825A01288 (life-cycle has no imperfection)**
  The force plot of the SHAP Linear Explainer for this case key is shown in Figure 16. From Fig, it can be interpreted that the model estimates that the

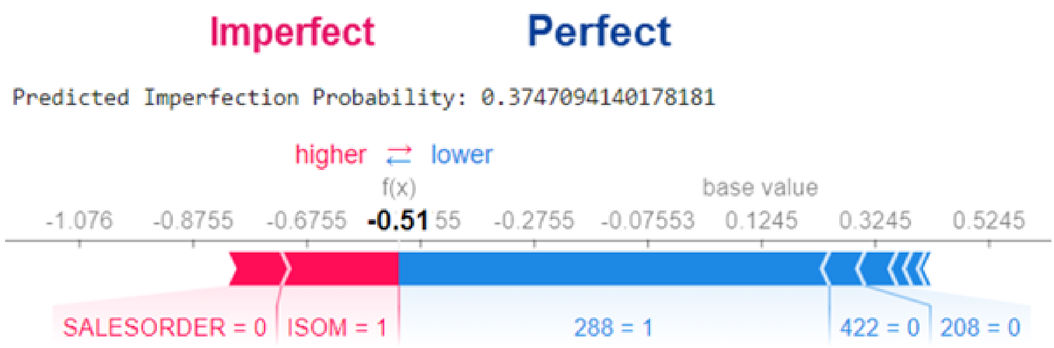


Figure 17: SHAP Linear Explainer for 1345897825A01288

order, 1345897825A01288 has a likelihood of imperfection to be 0.37 and the net contribution of features towards imperfection is less than that of perfection. In order to simplify the interpretation of the force plot, we generated a tabular heatmap, shown in Figure 18. Therefore, the top reasons (order attributes) resulting in a low likelihood of imperfection (0.37) are as follows:

	FeatureValue	Contribution
2	288 = 1	0.757607
4	422 = 0	0.062221
1	208 = 0	0.056825
0	202 = 0	0.025658
5	8 = 0	0.024283

Figure 18: Tabular Heatmap derived from SHAP Linear Explainer for 1345897825A01288

Activity-based Model Performance Analysis				
Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.90	0.63	0.87	0.73
Decision Tree	0.89	0.63	0.88	0.73
Random Forest	0.87	0.56	0.86	0.68
XGBOOST	**0.90**	**0.65**	**0.87**	**0.75**

Table 6: Performance Analysis of Activity-based Model

1. The order was processed at the online store, 288 having a very high fraction of perfect orders (as per the EDA).
2. The order was not processed at the physical stores, 422 and 208 that have very high fractions of imperfect orders (as per the EDA).

5.2 Activity-based Predictive Model

5.2.1 Model Evaluation

The trained models discussed in Section 4, considering the features buCode and orderType along with the activity features derived from the Bag-of-Activities were evaluated on the basis Accuracy, Precision, Recall, and F1-Score. The results of the model performance analysis are tabulated in Table 6. As discussed in Section 5.1.1, F1 Score is the metric which we put the most emphasis on. From Table 6, it is evident that XGBOOST has the highest F1 Score among the four models. Hence, it is selected to be the best model.

The XGBOOST Model has an accuracy of 90% which means out of 100 orders, it is correctly predicting the imperfection/perfection of 90 orders.
The Model has a Precision of 65% which means out of 100 predicted imperfect orders, 65 are correctly predicted as imperfect.
The Model has a Recall of 87% which means out of 100 actually imperfect orders, 87 are correctly predicted as imperfect.

5.2.2 Model Interpretation: Top reasons of Imperfection

After evaluating the model and selecting the best model (XGBOOST), we brought out the most important order features and life-cycle activities responsible for the model to estimate a certain likelihood of imperfection using SHAP for model interpretation. The SHAP Tree Explainer is used for explaining the XGBOOST Model in order to bring out the main features, and their contributions towards the model in estimating the likelihood of imperfection. We performed the model interpretation by taking three sample orders (_CASE_KEYS):

- **_CASE_KEY: 1358405196A01422 (it is actually an imperfect order)**
  The characteristics of the order are shown in Figure 19. The order life-cycle

	_CASE_KEY	buCode	orderType
4799	1358405196A01422	422	SALESORDER

  Figure 19: Order Characteristics for 1358405196A01422

  sequence of 1358405196A01422, along with the stage where it is cut before transforming it into the framework of Bag-of-Activities, is shown in Figure 20. The force plot of the SHAP Tree Explainer for this case key is shown in Figure 21. From Figure 21, it can be interpreted that the model estimates that the order, 1358405196A01422 has a likelihood of imperfection to be 0.99 and the net contribution of features towards imperfection is more than that of perfection.
  In order to simplify the interpretation of the force plot, we generated a tabular heatmap, shown in Figure 22. Hence, the top imperfection reasons responsible for the high likelihood of imperfection (0.99) are as follows:

  1. Sales order was NOT completed.
  2. Sales order was modified.
  3. Sales order was NOT created.

- **_CASE_KEY: 1345897825A01288 (life-cycle has no imperfection)**
  The characteristics of the order are shown in Figure 23. The order life-cycle

	_CASE_KEY	ACTIVITY_EN	SYSTEM	EVENTTIME
17212135	1358405196A01422	SALES ORDER MODIFIED	iSELL	2023-05-22 12:59:00.000
17212136	1358405196A01422	ORDER AUTHORIZED	iSELL	2023-05-22 12:59:00.000
17212137	1358405196A01422	SERVICE PRODUCT STATUS CHANGED	iSELL	2023-05-22 12:59:03.000
17212138	1358405196A01422	CPS_PLANNED	CPS	2023-05-22 12:59:03.599
17212139	1358405196A01422	CPS_ASSIGNED	CPS	2023-05-22 13:38:06.109
17212140	1358405196A01422	SERVICE PRODUCT STATUS CHANGED	iSELL	2023-05-22 13:40:55.000
17212141	1358405196A01422	CPS_PICKING	CPS	2023-05-22 13:40:55.339
17212142	1358405196A01422	CPS_PICKED	CPS	2023-05-22 13:45:47.528
17212143	1358405196A01422	SERVICE PRODUCT STATUS CHANGED	iSELL	2023-05-22 13:45:48.000
17212144	1358405196A01422	CPS_CHECKING	CPS	2023-05-22 14:42:21.102
17212145	1358405196A01422	SERVICE PRODUCT STATUS CHANGED	iSELL	2023-05-22 14:46:10.000
17212146	1358405196A01422	CPS_CHECKED	CPS	2023-05-22 14:46:10.482
17212147	1358405196A01422	LCD STATUS CHANGED	iSELL	2023-05-22 14:46:14.000
17212148	1358405196A01422	SERVICE PRODUCT STATUS CHANGED	iSELL	2023-05-22 14:46:14.000
17212149	1358405196A01422	CPS_COMPLETED	CPS	2023-05-22 14:46:14.430
17212150	1358405196A01422	LM_SHIPMENT CREATED	CENTIRO-LM	2023-05-22 14:46:18.000
17212151	1358405196A01422	LM_SP_CONTACTED SENT TO ISELL	CENTIRO-LM	2023-05-22 14:46:18.000
17212152	1358405196A01422	LM_SHIPMENT CREATED	CENTIRO-LM	2023-05-22 14:46:25.000
17212153	1358405196A01422	LM_SP_CONTACTED SENT TO ISELL	CENTIRO-LM	2023-05-22 14:46:25.000

Figure 20: Order Life-Cycle for 1358405196A01422

sequence of 1345897825A01288 is shown in Figure 24. The force plot of the SHAP Linear Explainer for this case key is shown in Figure 25.

From Figure 25, it can be interpreted that the model estimates that the order, 1345897825A01288 has a likelihood of imperfection to be 0.16 and the net contribution of features towards imperfection is less than that of perfection. In order to simplify the interpretation of the force plot, we generated a tabular heatmap, shown in Figure 26.

Therefore, the top reasons resulting in a low likelihood of imperfection (0.16) are as follows:

1. The Sales order was completed.
2. LM was received at hub.
3. Store pick-up was not required as the product was delivered to the exact location (from CUSTOMER DELIVERY COMPLETE activity in the sequence).

- **_CASE_KEY: 1360331646A01288 (it is actually an imperfect order**

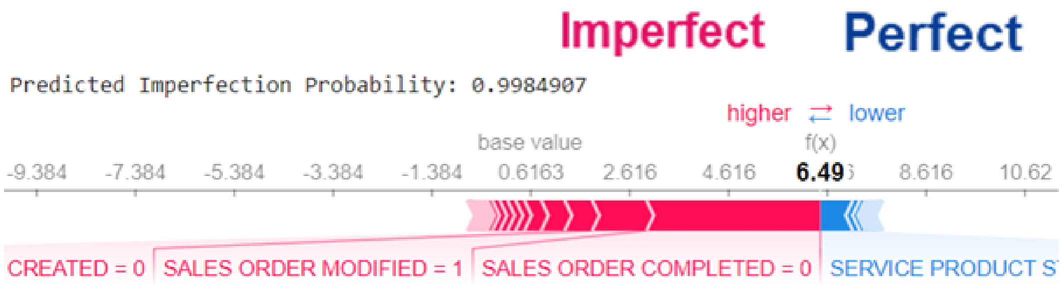


Figure 21: SHAP Tree Explainer for 1358405196A01422

	FeatureValue	Contribution
114	SALES ORDER COMPLETED = 0	3.519286
143	SALES ORDER MODIFIED = 1	1.099492
124	SALES ORDER CREATED = 0	0.556190
105	PAYMENT EXECUTED = 0	0.476054
25	SALES ORDER CANCELED = 0	0.340260

Figure 22: Tabular Heatmap derived from SHAP Tree Explainer for 1358405196A01422

despite PAYMENT EXECUTED and SALES ORDER COMPLETED being executed in the life-cycle)
The characteristics of the order shown in Figure 27. The order life-cycle sequence of 1360331646A01288 is shown in Figure 28. From Figure 28, it seems that 1360331646A01288 is quite a perfect order until SAC CREATED happened which is an imperfect activity. So, the sequence is cut just before SAC CREATED, transformed into the framework of Bag-of-Activities, and fed into the XGBOOST Model.
The force plot of the SHAP Tree Explainer for this case key is shown in Figure 29.

	_CASE_KEY	buCode	orderType
33836	1345897825A01288	288	ISOM

Figure 23: Order Characteristics for 1345897825A01288

	_CASE_KEY	ACTIVITY_EN	SYSTEM	EVENTTIME
14666452	1345897825A01288	SALES ORDER CREATED	iSELL	2023-05-03 12:37:18.000
14666453	1345897825A01288	SALES ORDER CONVERTED	iSELL	2023-05-03 12:37:18.000
14666454	1345897825A01288	LM_WORK ORDER CREATED	CENTIRO-LM	2023-05-03 12:37:22.000
14666455	1345897825A01288	CREATED	ISOM	2023-05-03 12:37:24.000
14666456	1345897825A01288	SENT FOR FULFILLMENT	ISOM	2023-05-03 12:37:25.000
14666457	1345897825A01288	CPS_PLANNED	CPS	2023-05-03 12:38:30.707
14666458	1345897825A01288	CPS_ASSIGNED	CPS	2023-05-05 11:04:27.683
14666459	1345897825A01288	CPS_PLANNED	CPS	2023-05-05 11:04:53.099
14666460	1345897825A01288	CPS_ASSIGNED	CPS	2023-05-05 11:18:56.789
14666461	1345897825A01288	CPS_PICKING	CPS	2023-05-05 11:25:31.307
14666462	1345897825A01288	RELEASED FOR PICKING	ISOM	2023-05-05 11:25:32.000
14666463	1345897825A01288	CPS_PICKED	CPS	2023-05-05 11:44:42.936
14666464	1345897825A01288	CPS_CHECKING	CPS	2023-05-05 12:04:34.722
14666465	1345897825A01288	PICKED	ISOM	2023-05-05 12:04:52.000
14666466	1345897825A01288	LM_DISPATCH COMPLETED	CENTIRO-LM	2023-05-05 12:04:52.000
14666467	1345897825A01288	CPS_CHECKED	CPS	2023-05-05 12:04:53.260
14666468	1345897825A01288	CPS_COMPLETED	CPS	2023-05-05 12:04:57.893
14666469	1345897825A01288	READY FOR DISPATCH	ISOM	2023-05-05 12:04:58.000
14666470	1345897825A01288	HANDED OVER TO TSP	ISOM	2023-05-05 13:05:34.000
14666471	1345897825A01288	LM_RECEIVED AT HUB	CENTIRO-LM	2023-05-05 17:21:00.000
14666472	1345897825A01288	RECEIVED AT LSC	ISOM	2023-05-05 19:41:04.000
14666473	1345897825A01288	LOADED ON DELIVERY TRUCK	ISOM	2023-05-09 12:29:26.000
14666474	1345897825A01288	DELIVERED	ISOM	2023-05-09 15:33:32.000
14666475	1345897825A01288	CUSTOMER DELIVERY COMPLETE	iSELL	2023-05-09 15:33:33.000
14666476	1345897825A01288	PAYMENT EXECUTED	iSELL	2023-05-09 15:33:33.000
14666477	1345897825A01288	SALES ORDER COMPLETED	iSELL	2023-05-09 15:33:33.000

Figure 24: Order Life-Cycle for 1345897825A01288

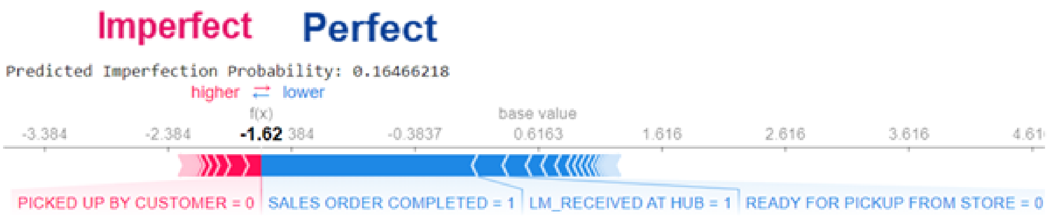


Figure 25: SHAP Tree Explainer for 1345897825A01288

	FeatureValue	Contribution
114	SALES ORDER COMPLETED = 1	1.750307
127	LM_RECEIVED AT HUB = 1	0.247706
129	READY FOR PICKUP FROM STORE = 0	0.189547
102	SALES ORDER CONVERTED = 1	0.124617
2	288 = 1	0.112690

Figure 26: Tabular Heatmap derived from SHAP Tree Explainer for 1345897825A01288

	_CASE_KEY	buCode	orderType
1468432	1360331646A01288	288	ISOM

Figure 27: Order Characteristics for 1360331646A01288

	_CASE_KEY	ACTIVITY_EN	SYSTEM	EVENTTIME
17593512	1360331646A01288	PAYMENT EXECUTED	iSELL	2023-05-24 18:45:00.000
17593513	1360331646A01288	SALES ORDER CREATED	iSELL	2023-05-24 18:45:02.000
17593514	1360331646A01288	LM_WORK ORDER CREATED	CENTIRO-LM	2023-05-24 18:45:08.000
17593515	1360331646A01288	CREATED	ISOM	2023-05-24 18:45:21.000
17593516	1360331646A01288	SENT FOR FULFILLMENT	ISOM	2023-05-24 18:45:22.000
17593517	1360331646A01288	CPS_PLANNED	CPS	2023-05-24 18:45:26.000
17593518	1360331646A01288	CPS_ASSIGNED	CPS	2023-05-25 04:05:08.000
17593519	1360331646A01288	CPS_PICKING	CPS	2023-05-25 04:09:39.000
17593520	1360331646A01288	RELEASED FOR PICKING	ISOM	2023-05-25 04:09:40.000
17593521	1360331646A01288	CPS_PICKED	CPS	2023-05-25 04:30:58.000
17593522	1360331646A01288	CPS_CHECKING	CPS	2023-05-25 04:43:13.000
17593523	1360331646A01288	LM_DISPATCH COMPLETED	CENTIRO-LM	2023-05-25 05:42:37.000
17593524	1360331646A01288	PICKED	ISOM	2023-05-25 05:42:38.000
17593525	1360331646A01288	CPS_CHECKED	CPS	2023-05-25 05:42:38.000
17593526	1360331646A01288	READY FOR DISPATCH	ISOM	2023-05-25 05:42:40.000
17593527	1360331646A01288	CPS_COMPLETED	CPS	2023-05-25 05:42:40.000
17593528	1360331646A01288	HANDED OVER TO TSP	ISOM	2023-05-25 07:04:56.000
17593529	1360331646A01288	LM_RECEIVED AT HUB	CENTIRO-LM	2023-05-25 12:25:00.000
17593530	1360331646A01288	RECEIVED AT LSC	ISOM	2023-05-25 13:11:45.000
17593531	1360331646A01288	LM_RECEIVED AT HUB	CENTIRO-LM	2023-05-26 04:37:00.000
17593532	1360331646A01288	RECEIVED AT PUP	ISOM	2023-05-26 10:26:48.000
17593533	1360331646A01288	PICKED UP BY CUSTOMER	ISOM	2023-05-26 16:02:52.000
17593534	1360331646A01288	CUSTOMER DELIVERY COMPLETE	iSELL	2023-05-26 16:02:53.000
17593535	1360331646A01288	SALES ORDER COMPLETED	iSELL	2023-05-26 16:02:53.000
17593536	1360331646A01288	RETURN MONEY SENT	SAMS	2023-05-27 07:27:36.000
17593537	1360331646A01288	SAC CREATED	SAMS	2023-05-27 08:26:02.558

Figure 28: Order Life-Cycle for 1360331646A01288

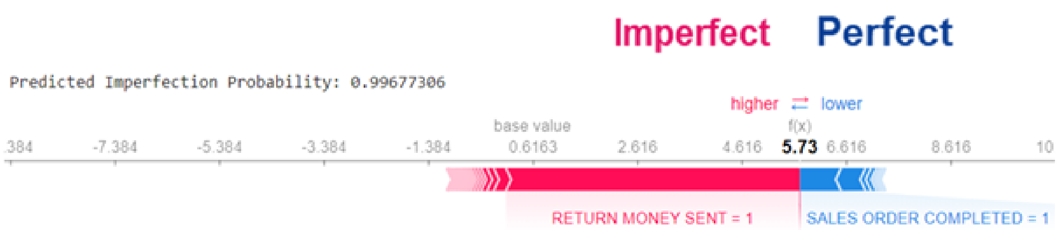


Figure 29: Fig. SHAP Tree Explainer for 1360331646A01288

From Figure 29, it can be interpreted that the model estimates that the order, 1360331646A01288 has a likelihood of imperfection to be 0.99 and the net contribution of features towards imperfection is more than that of perfection.
In order to simplify the interpretation of the force plot, we generated a tabular heatmap, shown in Figure 30. Therefore, the top imperfection reasons

	FeatureValue	Contribution
9	RETURN MONEY SENT = 1	5.625735
68	RECEIVED AT PUP = 1	0.235867
6	ISOM = 1	0.127187
67	CUSTOMER DELIVERY COMPLETE = 1	0.109331
123	CREATED = 1	0.066864

Figure 30: Tabular Heatmap derived from SHAP Tree Explainer for 1360331646A01288

(order attributes) responsible for the high likelihood of imperfection (0.99) are as follows:

1. Return money is sent.
2. Order was received at Pick Up Point (PUP).
3. 1360331646A01288 is a ISOM order.

6 Conclusions

In conclusion, our study has provided valuable insights into the probability of order imperfection in the Finland market through the application of two modelling approaches: non-activity-based and activity-based. Our findings highlight the limitations of relying solely on variables such as "Bucode" and "Ordertype" in accurately estimating imperfection likelihood. However, by incorporating the activities within the order life cycle, the activity-based modelling approach demonstrated a significant improvement in reliability. Furthermore, our analysis has successfully identified the most important reasons contributing to order imperfections. These reasons are closely tied to the unique characteristics of orders throughout their life cycle.

7 Recommendations

We recommend IKEA utilize our proposed method for estimating the likelihood of order imperfection and identifying the top reasons behind such imperfections specifically for the Finland market. By implementing this method, IKEA will be equipped with valuable insights into potential imperfections and their root causes, enabling the company to take proactive measures to prevent and mitigate order imperfections.

8 Future Work

Many adaption, tests and experiments have been left for the future to enhance the tool performance and capabilities. These works focus on customized training, improving prediction accuracy tailored to each stage of the OLC, and integrating the model into order systems to provide real-time likelihood of imperfection and suggest preventative actions. Furthermore, we recommend including additional order features through feature engineering and selection. Adapting the model to the OLC of different countries involves architectural adjustments, retraining, and validation. These modifications ensure that the model accommodates the specific details of the OLC in each country, optimizing its performance and accuracy.