

**Masters Programmes: Group Assignment Cover Sheet**

|  |  |
| --- | --- |
| **Student Numbers:** Please list numbers of all group members | **5665142, 5608651, 5660244, 5657386, 5619448, 5615136** |
| **Module Code:** | **IB9BW0** |
| **Module Title:** | **Analytics in Practice** |
| **Submission Deadline:** | **02/12/2024** |
| **Date Submitted:** |  |
| **Word Count:** | **1970** |
| **Number of Pages:** | **19** |
| **Question Attempted:**  *(question number/title, or description of assignment)* |  |
| **Have you used Artificial Intelligence (AI) in any part of this assignment?** | **Yes** |
| **Academic Integrity Declaration**  We're part of an academic community at Warwick. Whether studying, teaching, or researching, we’re all taking part in an expert conversation which must meet standards of academic integrity. When we all meet these standards, we can take pride in our own academic achievements, as individuals and as an academic community.  Academic integrity means committing to honesty in academic work, giving credit where we've used others' ideas and being proud of our own achievements.  In submitting my work, I confirm that:   * I have read the guidance on academic integrity provided in the Student Handbook and understand the University regulations in relation to Academic Integrity. I am aware of the potential consequences of Academic Misconduct. * I declare that this work is being submitted on behalf of my group and is all our own, , except where I have stated otherwise. * No substantial part(s) of the work submitted here has also been submitted by me in other credit bearing assessments courses of study (other than in certain cases of a resubmission of a piece of work), and I acknowledge that if this has been done this may lead to an appropriate sanction. * Where a generative Artificial Intelligence such as ChatGPT has been used I confirm I have abided by both the University guidance and specific requirements as set out in the Student Handbook and the Assessment brief. I have clearly acknowledged the use of any generative Artificial Intelligence in my submission, my reasoning for using it and which generative AI (or AIs) I have used. Except where indicated the work is otherwise entirely my own. * I understand that should this piece of work raise concerns requiring investigation in relation to any of points above, it is possible that other work I have submitted for assessment will be checked, even if marks (provisional or confirmed) have been published. * Where a proof-reader, paid or unpaid was used, I confirm that the proof-reader was made aware of and has complied with the University’s proofreading policy.   **Upon electronic submission of your assessment you will be required to agree to the statements above** | |

**Contents**

[Introduction 1](#_Toc183615433)

[Design & Architecture 2](#_Toc183615434)

[Code & Explanation 3](#_Toc183615435)

[1. Data Preparation 3](#_Toc183615436)

[2. Feature Selection & Engineering 4](#_Toc183615437)

[3. Modelling 12](#_Toc183615438)

[Recommendations 17](#_Toc183615439)

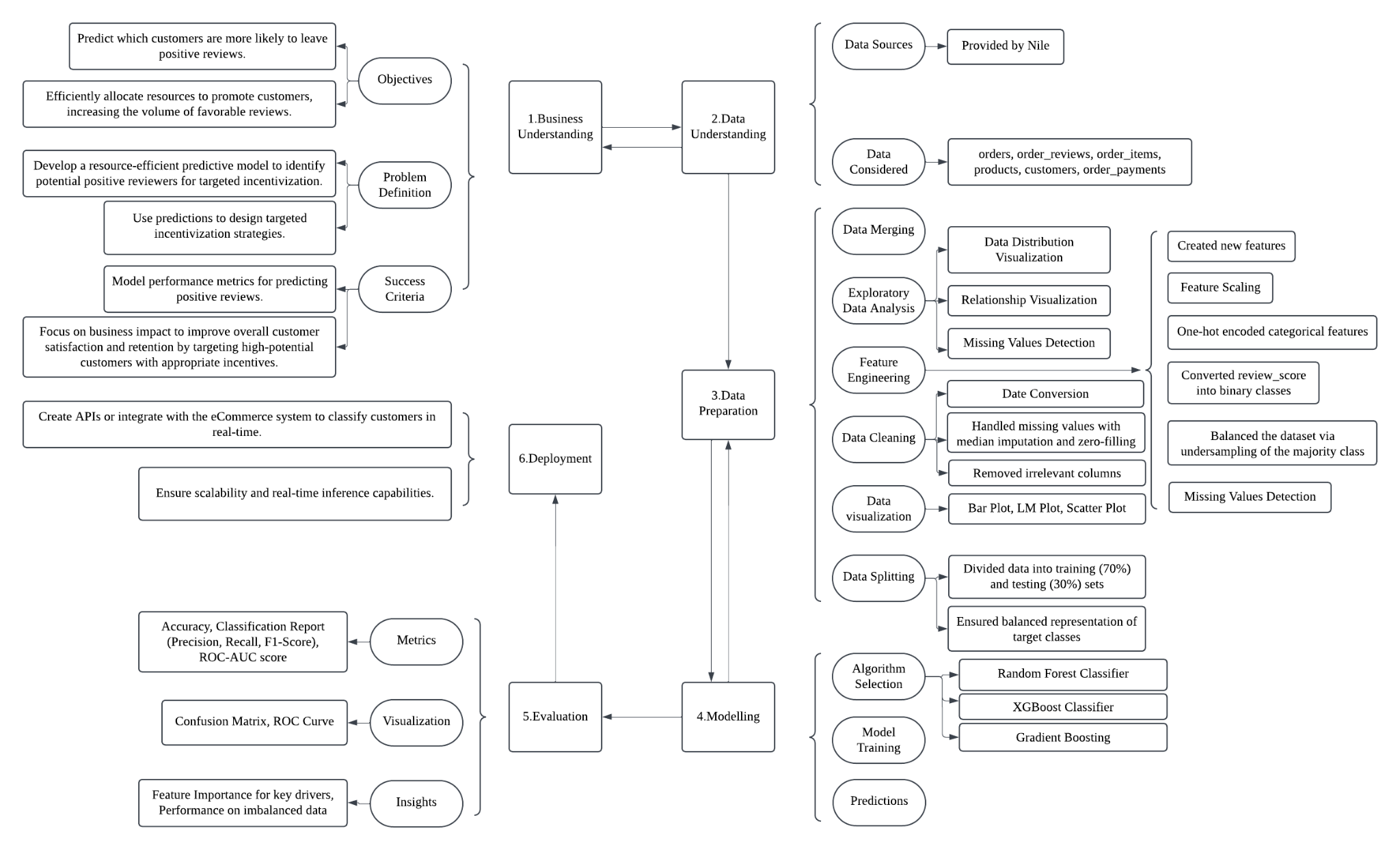
[Results & Conclusion 19](#_Toc183615440)

# Introduction

In the competitive landscape of eCommerce, maintaining a positive reputation is critical to attracting and retaining customers. Reviews play a pivotal role in shaping potential buyers' perceptions, making it essential for platforms to maximise positive customer feedback. This report focuses on developing a predictive model for a major South American eCommerce platform, "Nile," to identify which customers are more likely to leave positive reviews. By predicting positive reviews, the platform aims to efficiently allocate resources to incentivise such customers, thereby increasing the volume of favourable reviews in a cost-effective manner, a key objective of this project.

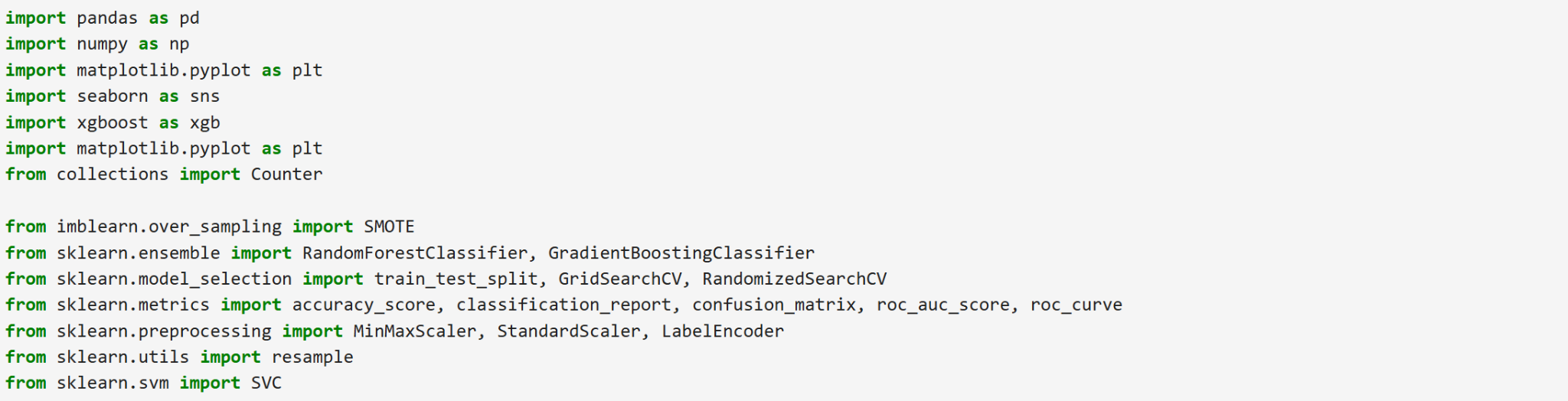
To achieve this goal, we were provided with a dataset consisting of various tables, including customer details, order information, product categories, and review scores. We employed a diverse range of models, including Gradient Booster, XGBoost and Random Forest to develop a classification model that can predict whether a review will be positive. Our approach includes handling class imbalance, optimising hyperparameters, engineering relevant features, and tuning decision thresholds to enhance the recall scores by using the Overfitting method.

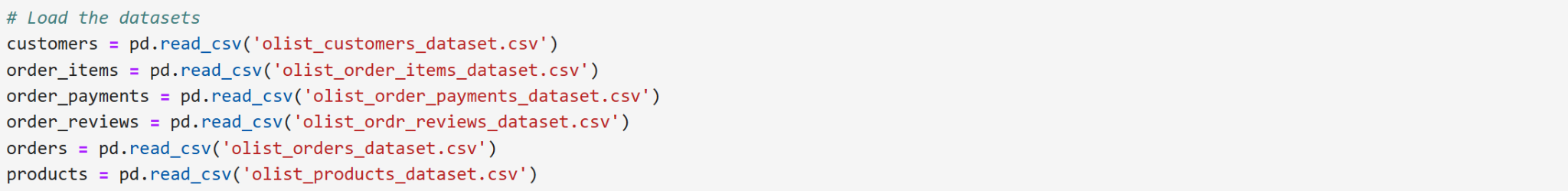
# Design & Architecture



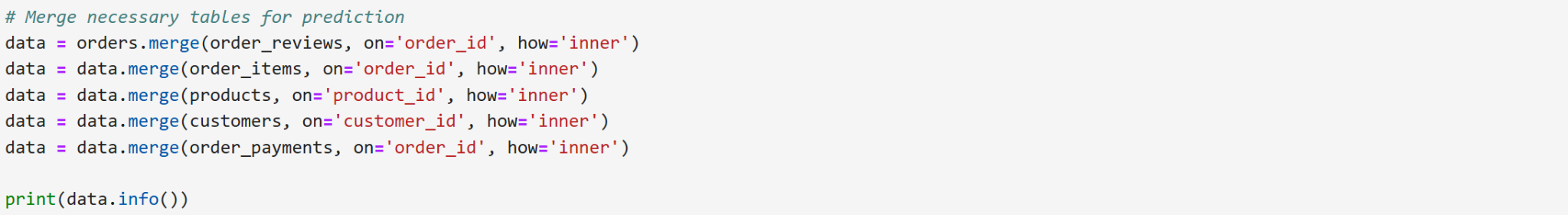
# Code & Explanation

## Data Preparation

****

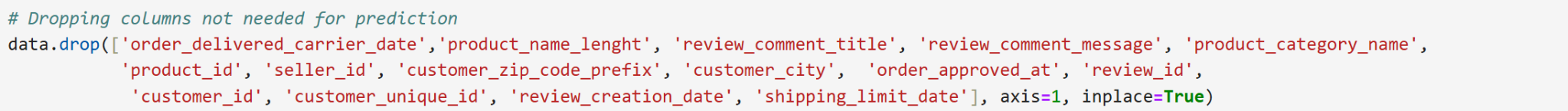
****

This code snippet is part of the data preparation process which imports all the csv files in the dataset for modelling.



For data modeling, we focused on **orders**, **order\_reviews**, **order\_items**, **products**, **customers**, and **order\_payments**, excluding **geolocation**, **sellers**, and **category\_translation** as their columns were deemed irrelevant.

Inner joins were used in all the tables, so that only records with matching keys across both tables will be retained, ensuring consistency in the dataset.

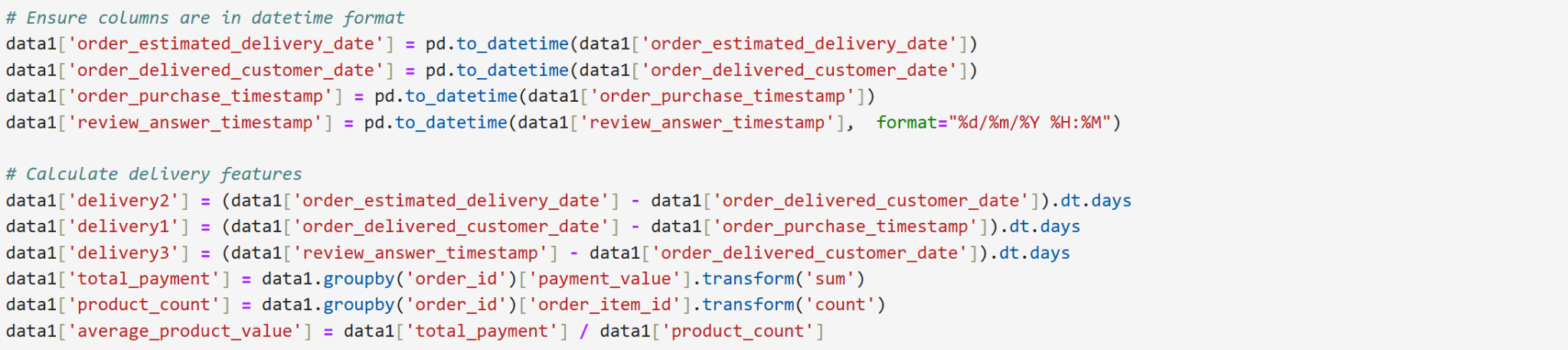
****

This step cleans the dataset by removing columns that are not relevant for prediction, which helps improve the efficiency of the model and reduces unnecessary complexity.



Creating a copy dataset to **preserve the original data**.

## Feature Selection & Engineering



**pd.to\_datetime()** is used to convert columns into **datetime format**, enabling date calculations.

Six new features have been created –

**'delivery2’**:

* Calculates the difference in **days between the estimated delivery date** (order\_estimated\_delivery\_date) and the **actual delivery date** (order\_delivered\_ customer\_date).
* .dt.days returns the difference in days.

**'delivery1’**:

* Calculates the difference in **days between the order purchase** (order\_purchase\_timestamp) and **actual delivery** (order\_delivered\_customer\_date).

**'delivery3'**:

* Calculates the difference in **days between the review answer** (review\_answer\_timestamp) and the **actual delivery** (order\_delivered\_customer\_date).

**'total\_payment'**:

* Uses groupby('order\_id') to calculate the **total payment** value for each order.

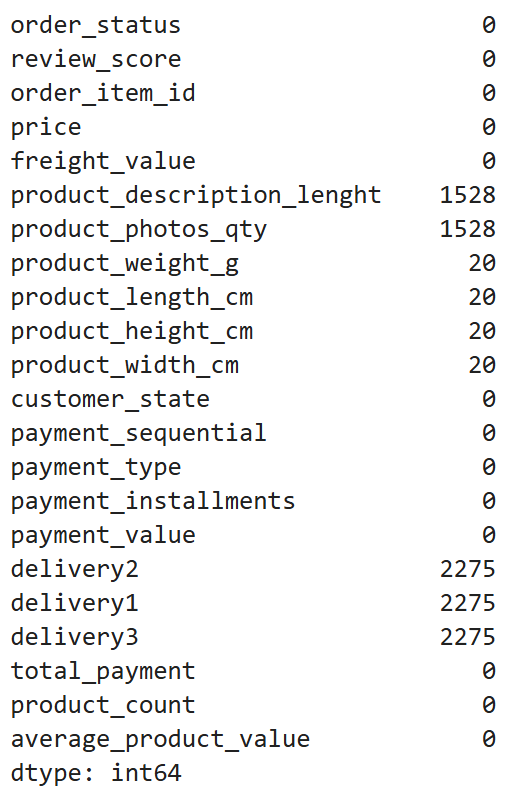
**'product\_count'**:

* Calculates the **number of items** in each order by counting the occurrences of order\_item\_id within each group (order\_id).

**'average\_product\_value'**:

* Computes the **average value per product** for each order by dividing the total\_payment by the product\_count.

These conversions are necessary to perform calculations involving dates (such as differences between delivery times).



This is a **basic data quality check** to understand if there are missing values in each column and how many.

It helps in deciding **data cleaning steps** (e.g., imputation or dropping columns/rows) before feeding the data into a model for analysis or prediction.



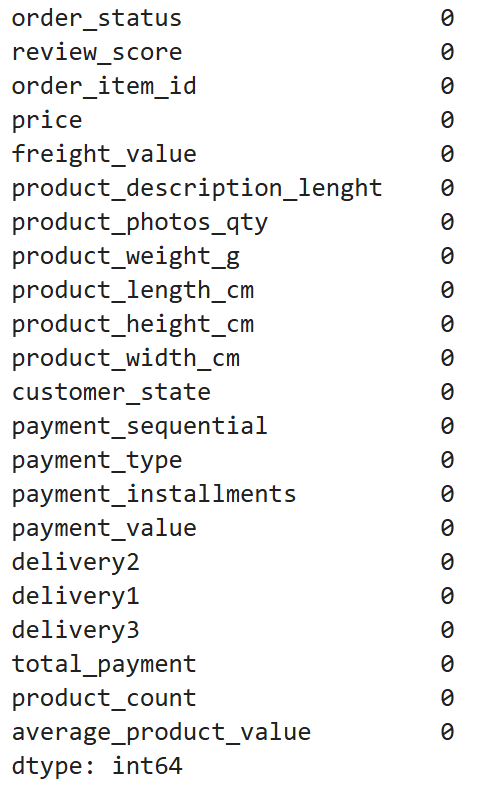
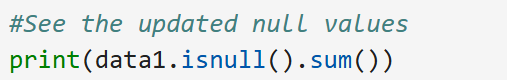
The **delivery1, delivery2 and delivery3** columns represent delivery-related features, and filling missing values with 0.0 suggests that these cases likely do not have meaningful delivery durations to calculate (perhaps items were never delivered), so 0.0 is used as a placeholder.

Filling missing values in the **product\_description\_lenght** and **product\_photos\_qty** column with the median value of that column.

The median is chosen as a replacement because it is less sensitive to **outliers** and provides a reasonable value for missing data without skewing the distribution significantly.

**data1 = data1.dropna(subset=['product\_height\_cm'])**:

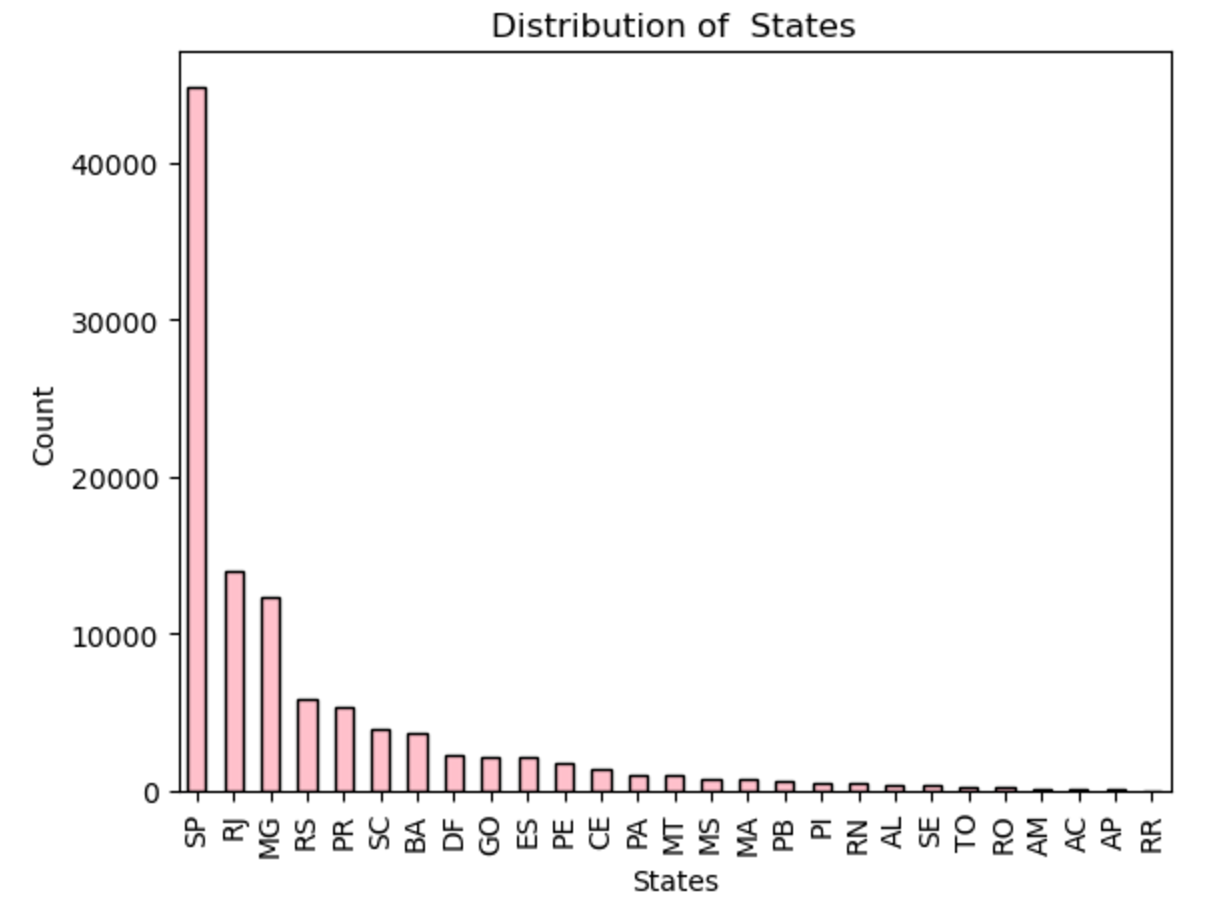
* Drops rows where the value in the product\_height\_cm column is missing (NaN).



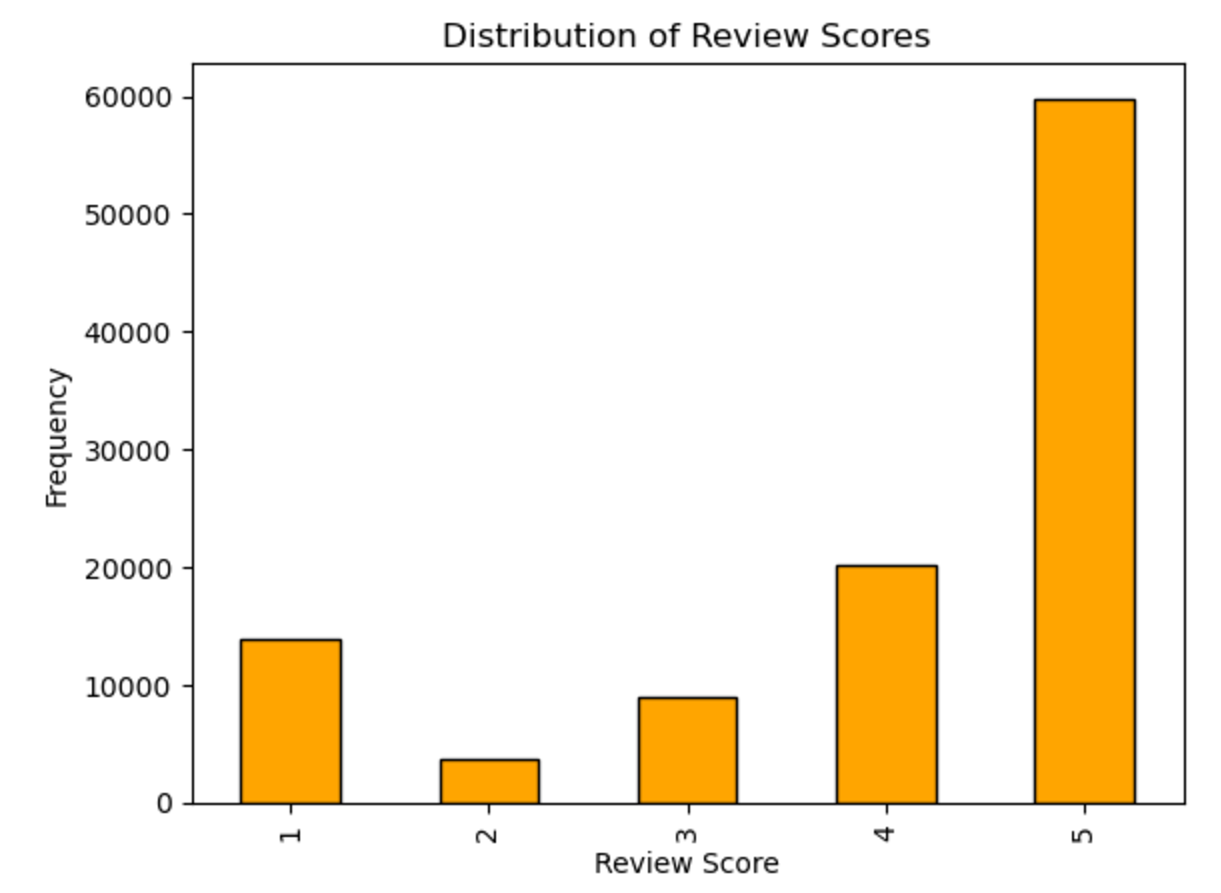
This is to check if all the missing values were handled.



The bar chart provides a **visual representation** of how often each order status occurs, which is useful for understanding the data balance and identifying potential issues.



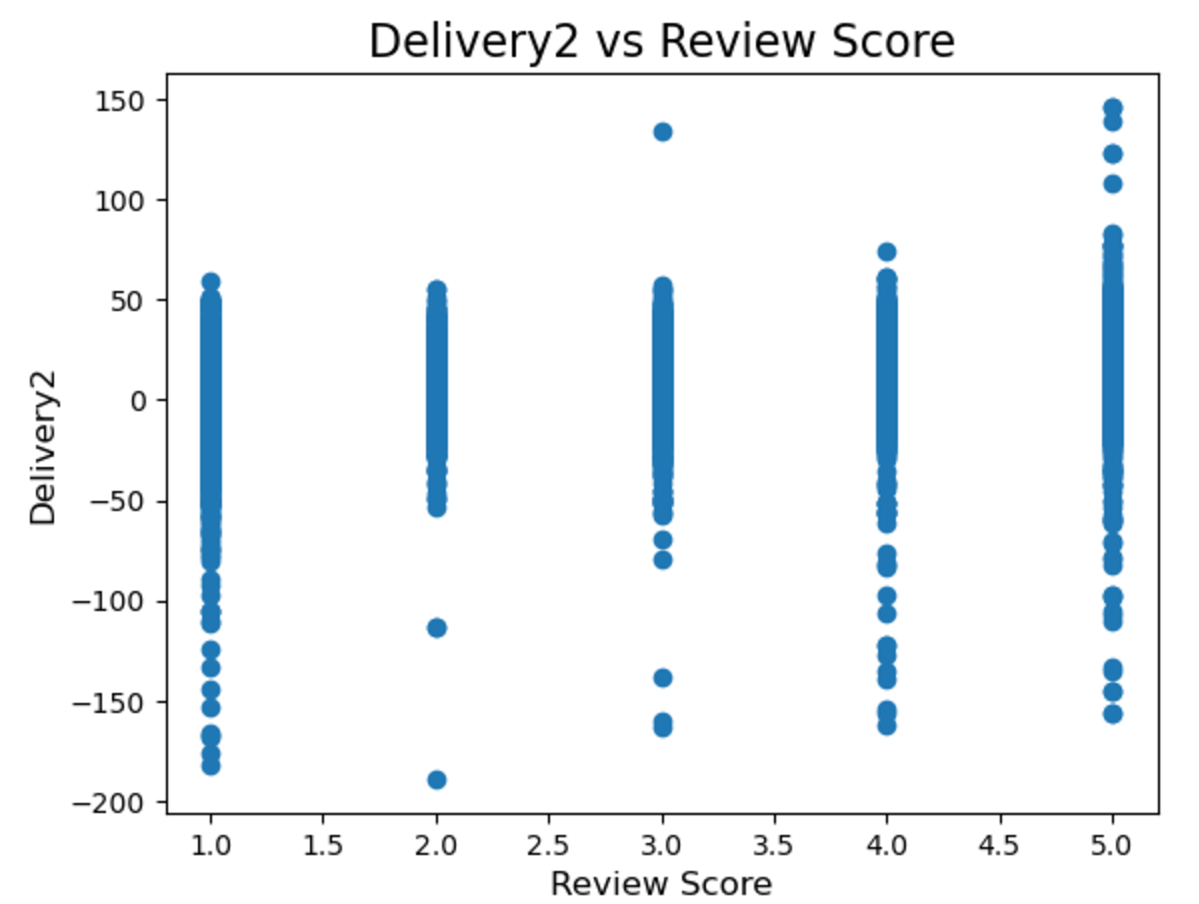
This allows us to quickly see which states have the **highest or lowest number of customers**.



It allows us to see which scores are most common and whether customers tend to give **positive** (e.g., scores of 4 or 5) or **negative** (e.g., scores of 1 or 2) feedback.



The plot provides valuable insight into how **delivery performance** impacts customer feedback. Faster delivery times are linked to **higher review scores**, while longer delivery times lead to customer dissatisfaction. This highlights the importance of **efficient logistics** to improve the overall customer experience.

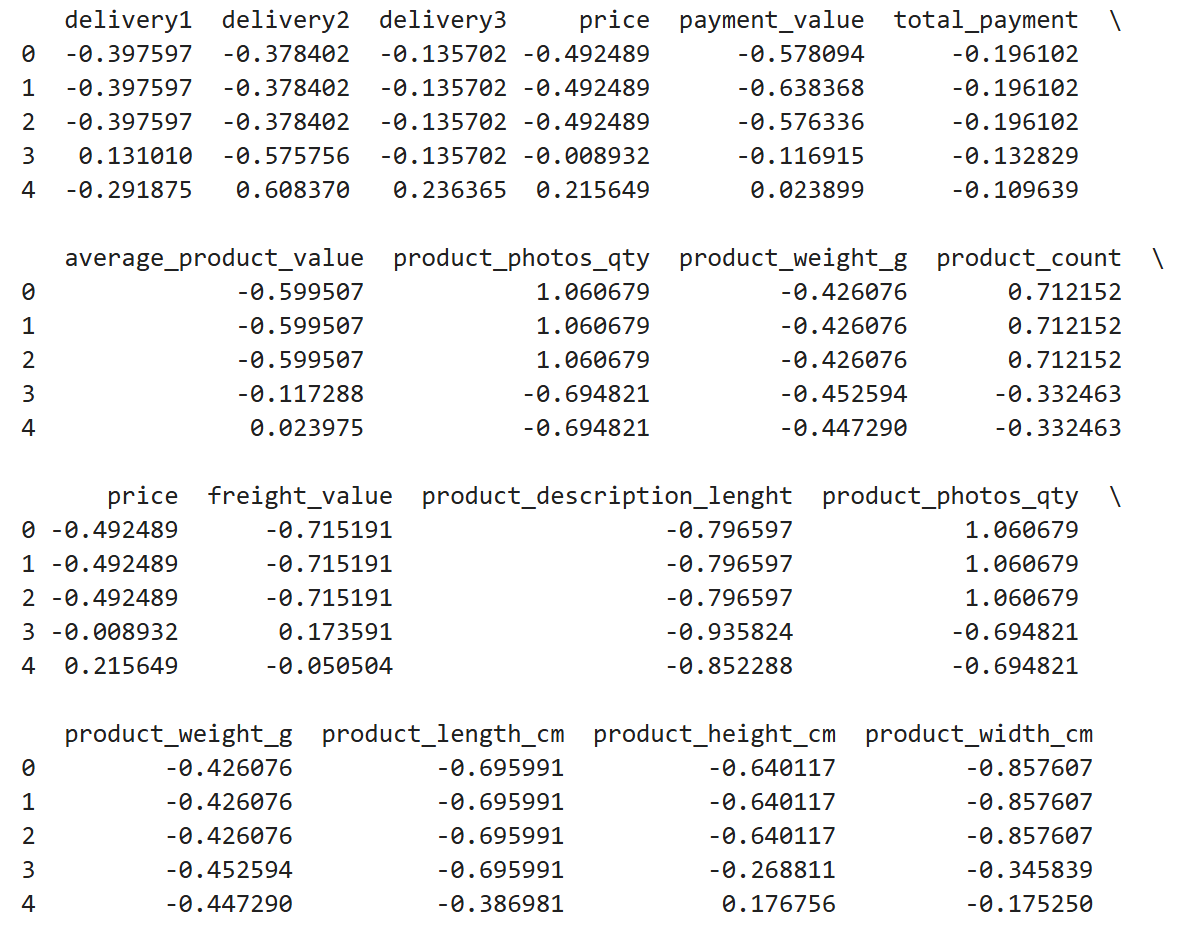


This scatter plot visualises the relationship between delivery2 and review\_score. Each point in the scatter plot represents a **review score** with its corresponding difference between **estimated delivery date** and the **actual delivery date**.



Each point in the scatter plot represents an individual product with its corresponding **review score** and **price**.



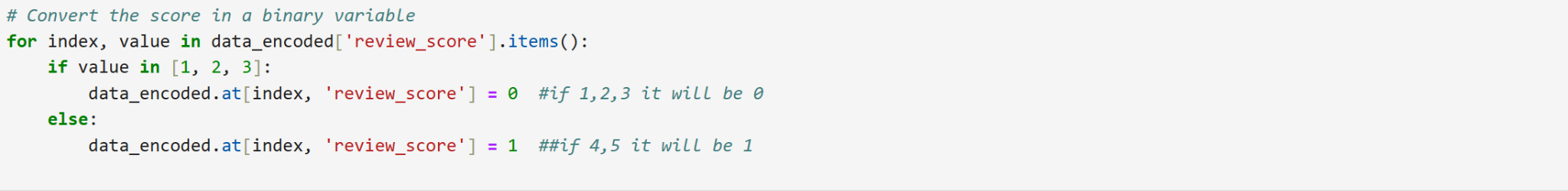


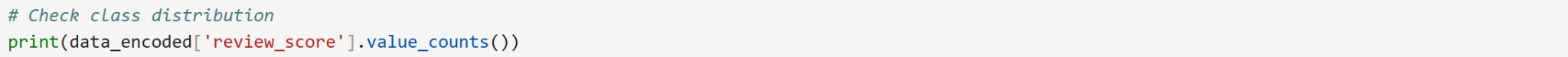
After standardisation:

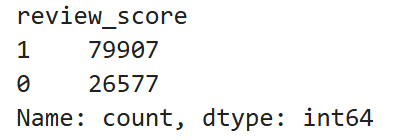
* The values are generally within a **smaller range** centred around **0**.
* Each value represents how far it deviates from the mean in units of standard deviation.



The data\_encoded DataFrame now has the original columns (customer\_state, order\_status, payment\_type) **replaced** with a set of **binary columns** for each category (except the one dropped).

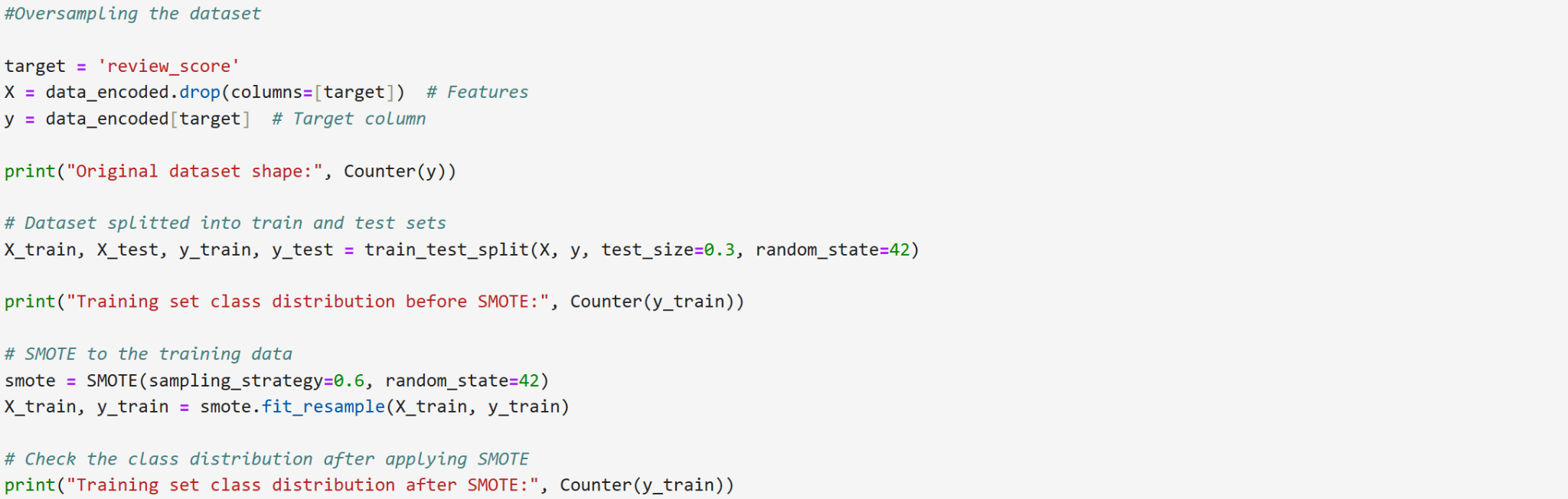


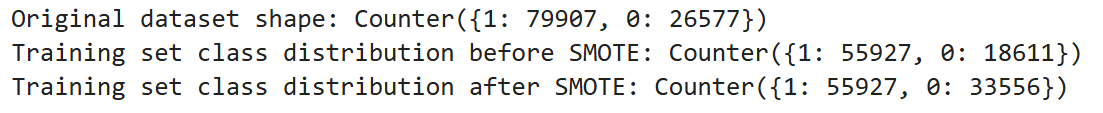




The code transforms the **review\_score** column into a **binary classification** column, where **0** represents scores 1, 2, 3 and **1** represents scores 4, 5.

The resulting **class distribution** shows that **positive reviews are the majority**,indicating the need to consider methods like **resampling** (e.g., oversampling negative reviews) or using **class-weighting** to address class imbalance for future modelling.





The original dataset was **imbalanced**, with **79,907 positive reviews** (review\_score = 1) and **26,577 negative reviews** (review\_score = 0).

This code used **SMOTE** to **oversample** the minority class in the training set, generating synthetic samples to balance the class distribution, ensuring the model can learn patterns from both majority and minority classes effectively.

## Modelling

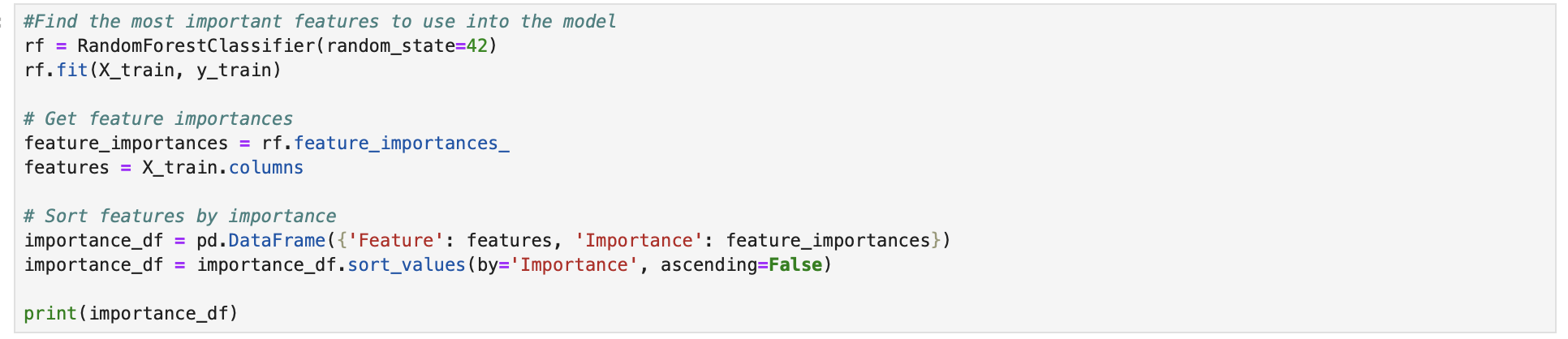
The code divides the dataset firstly in X and Y by removing the target variable, **review\_counts**, from the X dataset and adding the latter to the Y. Finally these two set have been divided into train and test with the respectively percentages 0.7 and 0.3

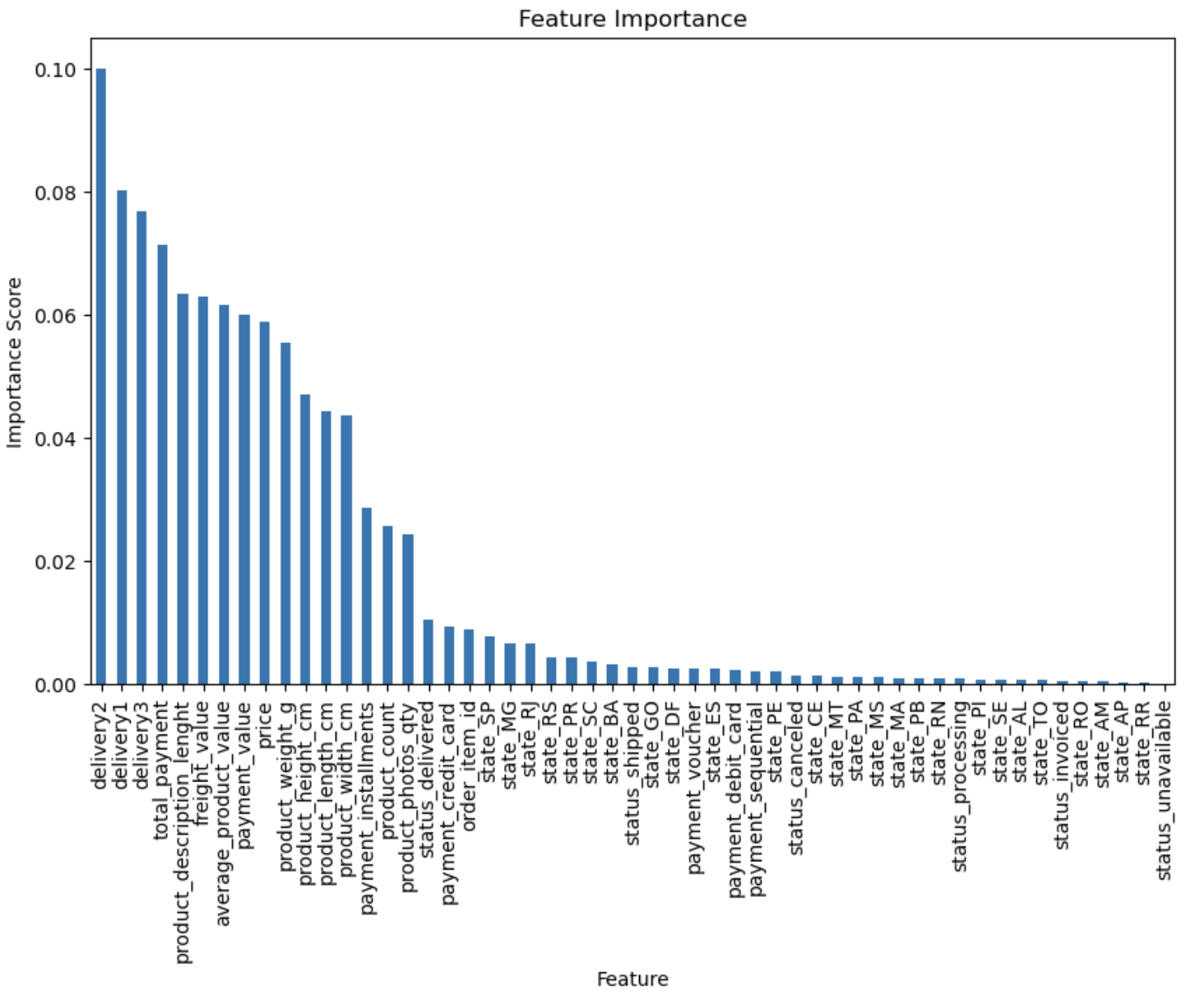
****

****

These are the dimensions of the different sets.

In order to compute the optimal result, a random forest model that calculates the features importance was created.





The above-graph gives a visual illustration of the importance score of the variables presented in the dataset, which is a wide range varying from 0.10 with ‘delivery2’, to almost 0.0 with ‘status\_unavailable’.

From the image a difference between the variables' importance in the model arose, in order to tackle this problem a threshold of 0.01 was set. The following code addresses this issue by removing in both the X train and test set the values under the chosen threshold.

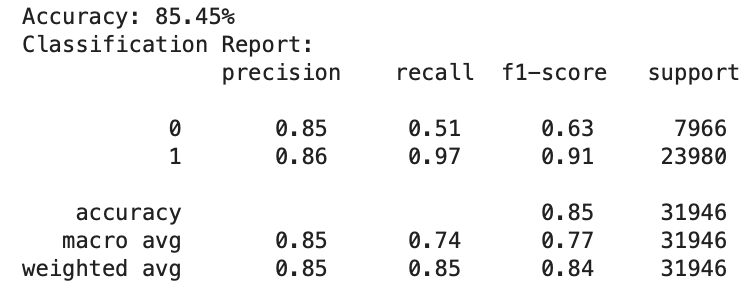




The fig(?) displays that from the 53 column our set has been reduced to 17.

During the analysis different algorithms have been used, in the report only the preferred one has been reported, the other two models used were GradientBoosting, and Xgboost. Even if they are similar the implementation adopted required firstly the use of GradientBoosting than Xgboost and finally Random Forest. The latter has resulted in outperforming the other two models.

The RandomForest model gave an **accuracy** of 85.45%, this result alone is not enough to understand how the algorithm performs. Therefore, it must be combined with a classification report, a ROC-AUC curve and a confusion matrix.



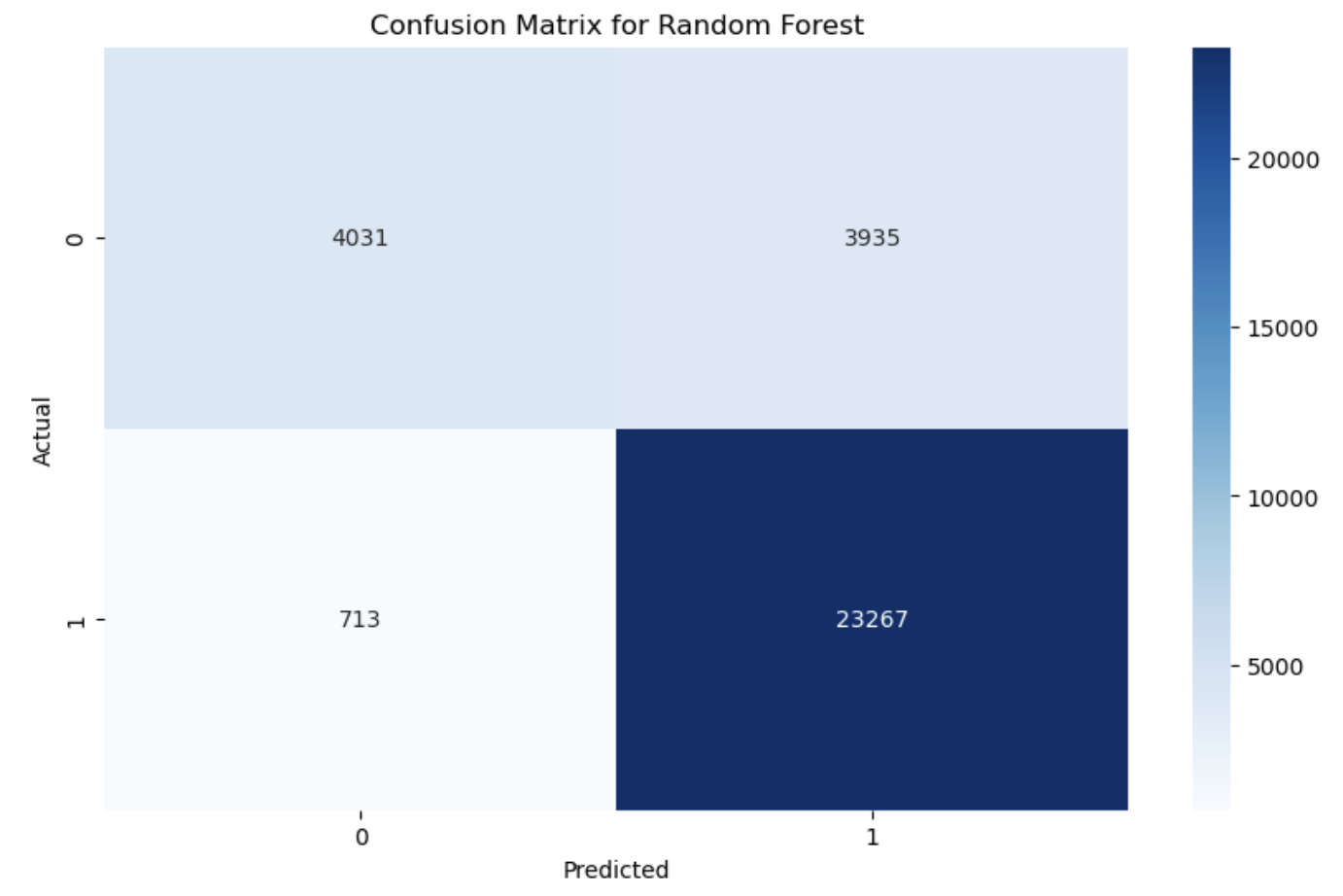
The classification report shows a **precision,** the proportion of correctly predicted positive instances out of all instances predicted as positive, of 0.85 for class 0 and 0.86 for class 1, therefore 85% of the predictions labeled as class 0 were correct as well as 86% of the predictions labeled as class 1 were correct.

The next score to analyse is the **recall,** the proportion of correctly predicted positive instances out of all actual positive instances, having for class 0, **0.51,** and class 1, 0.97.

Finally the **F1-Score**, the harmonic mean of precision and recall that balances the two metrics, the class 0 was **0.63** while class 1, **0.91**.

The Confusion Matrix displays 4031 **true negatives,** the instances where the model correctly predicted class 0, and 23267 **true positives,** the instances where the model correctly predicted class 1.

For the **false positives,** 3935 instances where predicted, this means that they were predicted in class 1 but were actually class 0. On the other hand, for the **false negatives,** the instances where the model predicted class 0, but the actual class was 1 was 713.



The ROC curve shows the trade-off between the True Positive Rate and the False Positive Rate. Its performance is represented by the blue line, with the curve being closer to the top-left corner the better the model is. For the Random Forest the ROC-AUC score is 0.81, which indicates that the model has a 81% chance in distinguishing between the two classes.



In conclusion Random Forest performs as the best algorithm given the metrics result and the most reliable given its accuracy.

# Recommendations

To ensure the predictive model is constructively implemented and its insights are actionable for all stakeholders, the following recommendations address both technical and non-technical needs in an integrated manner:

1. **Streamlining Customer Engagement:**

Using the model to identify customers who are most likely to leave positive reviews and prioritise incentivising these customers with targeted campaigns, such as discounts or personalised offers. This resource-efficient approach reduces costs while enhancing customer satisfaction.

1. **Improving Delivery and Service Quality:**

Insights from the analysis revealed that faster and more precise deliveries strongly influenced positive reviews. Stakeholders across logistics and technology should collaborate to enhance delivery timelines and adherence to estimated delivery dates. This can involve technical solutions for route optimization and operational process improvements driven by business teams.

1. **Model Deployment and Integration:**

Implementing the predictive model into Nile’s operational system using APIs to enable real-time decision-making. Technical teams should ensure scalability and robust performance to handle large datasets, whereas business teams should ensure the acumens are actively used in customer commitment strategies.

1. **Monitoring and Continuous Improvement:**

Establishing a feedback loop to evaluate the model's performance on a regular basis. Metrics such as accuracy, recall, and F1-score should be tracked by technical teams, while business teams should use these insights to improve engagement strategies. Retraining the model periodically will ensure it remains appropriate as customer behavior unfolds.

1. **Cross-Team Collaboration:**

Conducting regular cross-functional meetings to ensure all stakeholders understand the model's results and implications. Technical teams can provide simplified visual reports (e.g., dashboards, ROC curves) to demonstrate the model’s impact, and business teams can translate these insights into actionable strategies.

1. **Future Developments**:

Exploring additional features such as seller data or sentiment analysis may refine predictions further. Both technical and non-technical teams can collaborate on assessing new features to enhance the platform's overall review management process.

# Results & Conclusion

**Results:**

1. Data Preparation:
   * Relevant tables were joined and missing values were addressed. Features like delivery times, total payment and product count were created to enhance insights.
   * Review scores were transformed into a binary target: 0 for reviews 1–3 stars and class 1 for 4–5 stars.
2. Insights from EDA:
   * Delivery speed and accuracy strongly influenced positive reviews, highlighting the importance of efficient logistics.
   * São Paulo was identified as the largest customer base, making it a strategic focus area for customer engagement.
   * Product price showed very small impact on reviews, emphasising that service and delivery are the key operators of customer satisfaction.
3. Class Balancing:
   * To address the imbalance in review scores, under sampling was used on Class 1 to create a balanced dataset, ensuring fair predictions during model training.

**Conclusion:**

The dataset is well-prepared for modelling, with critical predictors like delivery performance and spending patterns identified. A binary classification model will allow Nile to predict customers likely to leave positive reviews, enabling targeted and resource-efficient review collection strategies. These insights also highlight the need to focus on logistics and customer satisfaction in São Paulo, ensuring sustained positive feedback and strengthened brand loyalty. Significant preprocessing, merging, and feature engineering to create a unified dataset suitable for building prediction models.