Model Development Life Cycle (MDLC)

Team Members:
Maria Alice Fagundes Vieira
Aritra Ray

Table of Contents

1. Problem Definition and Objective

Santander Bank's current product recommendation system is inefficient, leading to an uneven customer experience. Some customers receive an abundance of recommendations, while others receive very few or none at all. This disparity results in missed opportunities for both the bank and its customers. The challenge is to develop a more sophisticated and personalized recommendation model that can accurately predict which products each customer is most likely to need or want in the last month of the dataset, based on their historical behavior and transaction data. In other words, they need to know where the demand is so they are able to supply. We will create a machine learning model that will assist on which customers to target along with which products from the bank they are more likely to buy. The system aims to enhance customer satisfaction and drive product adoption while ensuring interpretability and operational efficiency.

---

2. Data Understanding and Exploratory Data Analysis (EDA)

2.1. Dataset Overview

The dataset contained 48 features, including:

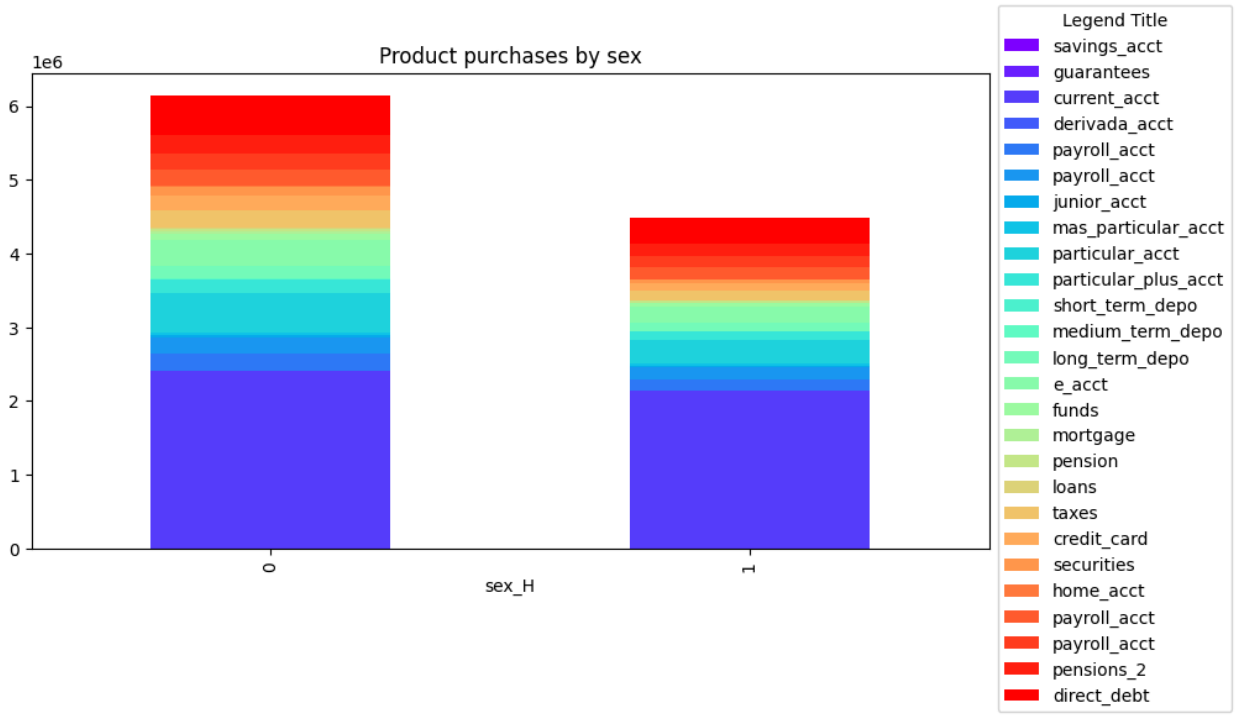
- **Demographics**:e `age, sex_H, income, income_to_product_ratio, income_to_age`.
- **Account Features**: `current_acct, savings_acct, credit_card, mortgage,` and more.
- **Categorical Variables**: `join_channel_encoded, province_name_encoded, employee_index_encoded.`
- **Target Variable**: Columns ending with `"_acct"` and other product-related features, representing products to be recommended.

2.2. EDA Findings

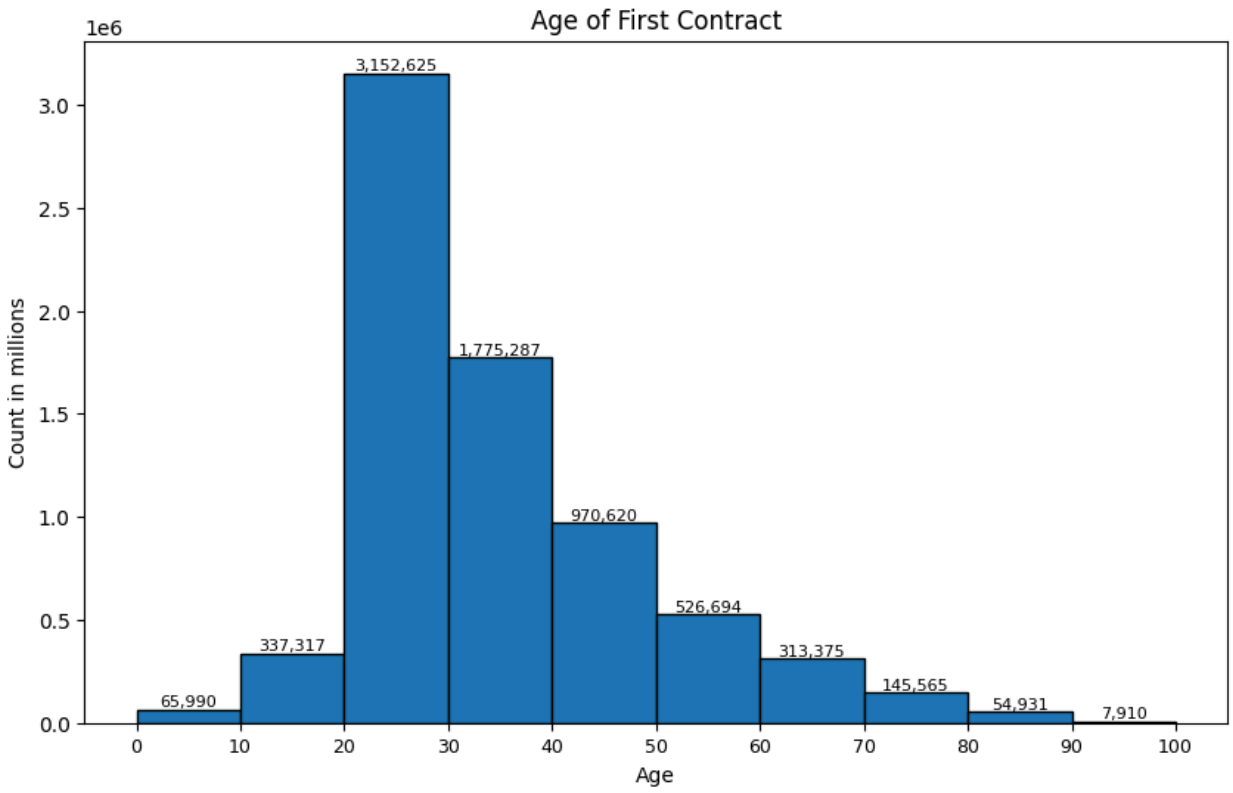
- **Gender Distribution**: The dataset has a higher proportion of male clients than female clients. Male customers have also bought more products, contributing to a skewed distribution in the total products purchased.
- **Age Distribution**: The average and median age of customers are 40 years old. The dataset has a broad range of ages, with notable concentrations in the 20-30 and 40-50 age groups. The 10-20 age range is dominant for junior accounts, while older clients (50-60 years) are associated with products like savings and home accounts.
- **Seniority**: Seniority in the bank varies widely, from 0 to 256 months. Newer clients typically buy products like short-term deposits and the `"Mas Particular"` account.
- **Product Preferences**: There is a significant variation in product ownership. Male clients generally buy more products than female clients. Certain products, such as the current account, junior account, and short-term deposits, have a higher representation of newer customers, while others like savings and loans are bought more by older customers.
- **Geographic Distribution**: The bank's client base is predominantly located in Madrid, followed by other cities like Barcelona, Valencia, and Sevilla. Product distribution generally mirrors the geographic spread, with Madrid clients dominating guarantees accounts and Barcelona clients favoring loans and savings accounts.
- **Customer Activity**: The dataset contains a large number of inactive customers, particularly those holding savings, current, and particular accounts. Active customers are more prevalent across most product categories, with current, junior, and short-term accounts being the most popular.

Visuals:

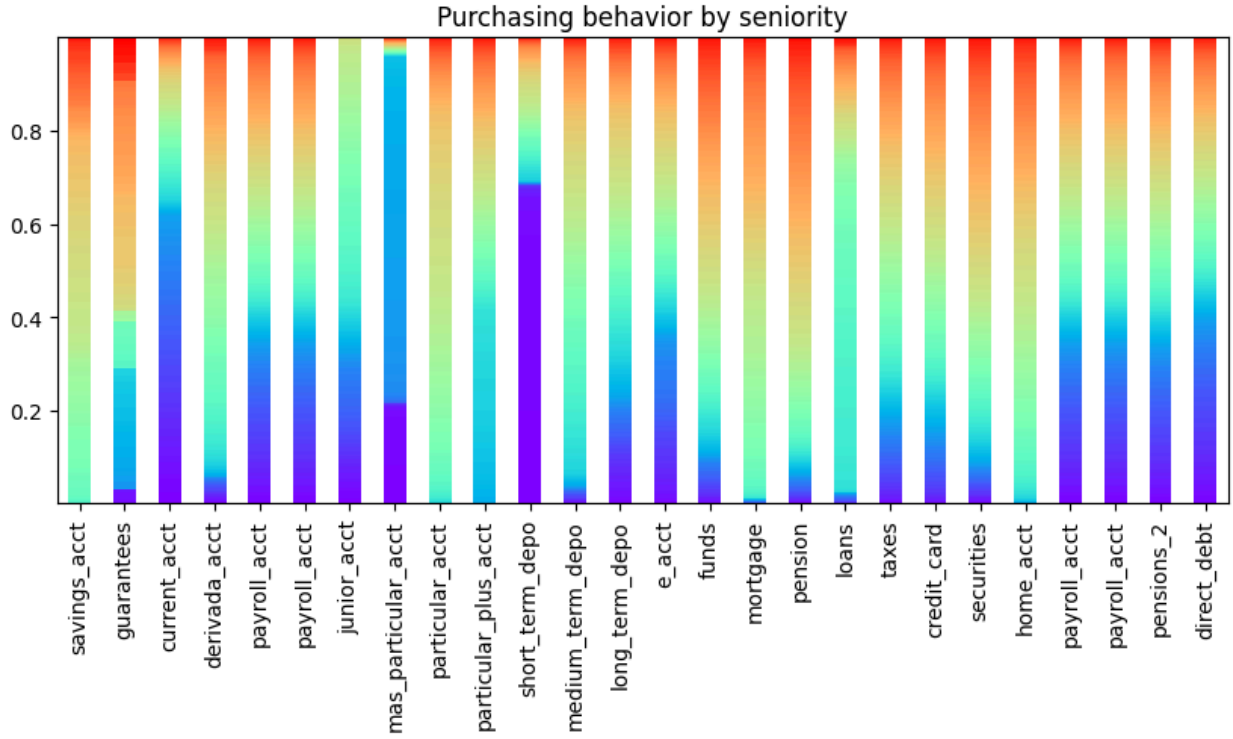
1. This visually highlights the gender imbalance in the dataset and its impact on product ownership.



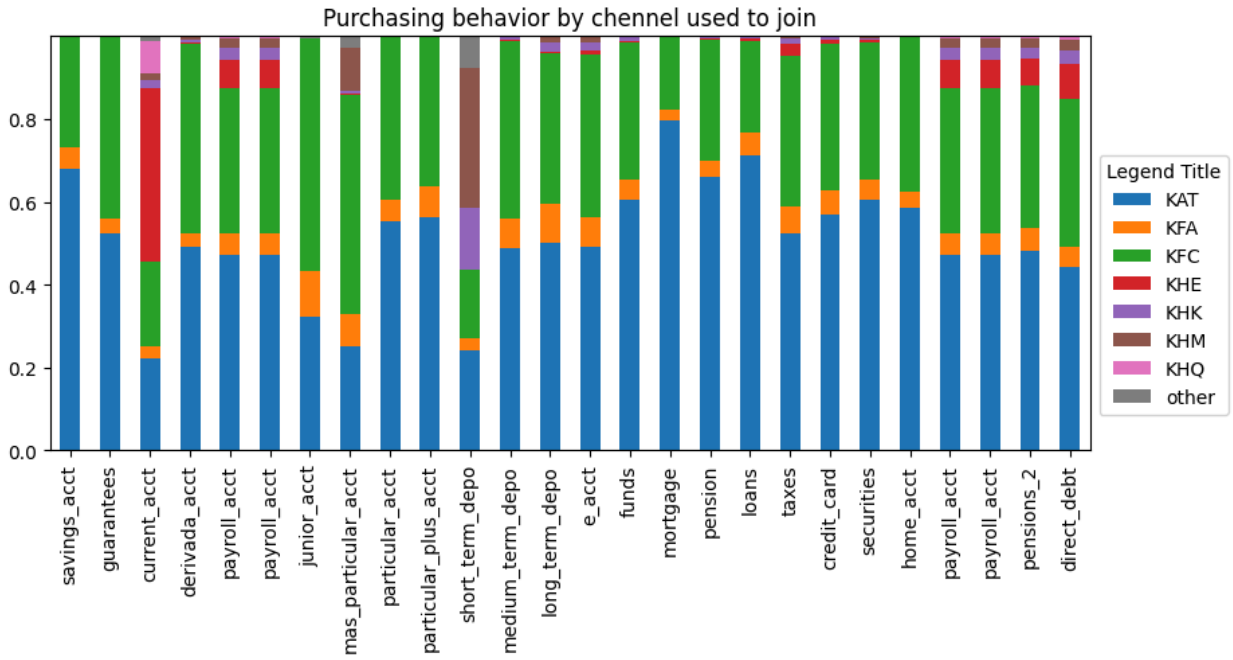
2. Focuses on key age groups such as 20-30 and 40-50, will help demonstrate the diversity in customer demographics.



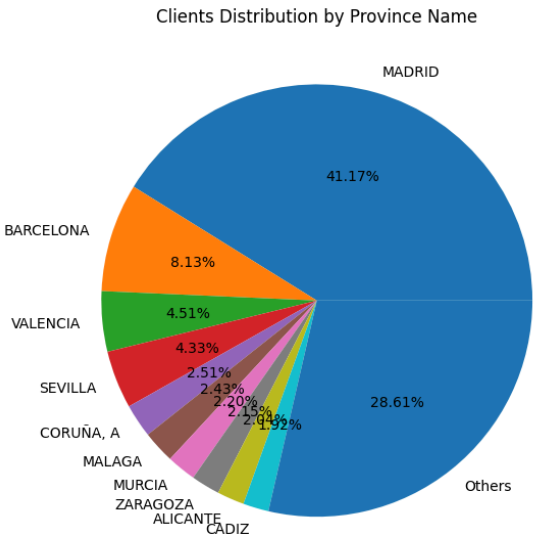
3. This provides a clear visual of how seniority relates to product ownership.

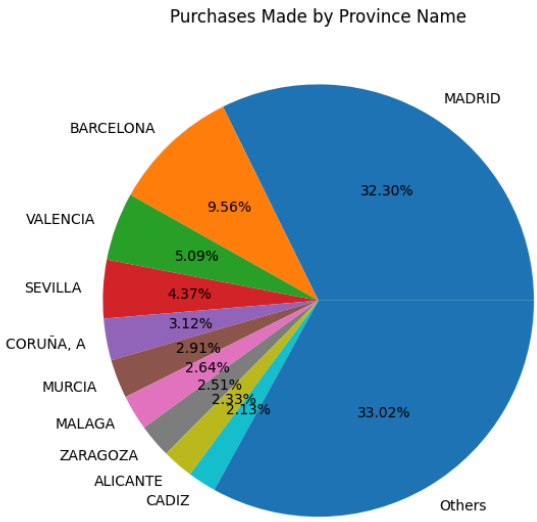


4. Purchasing behaviour by channels can help identify most preferred channel. Customers who joined through KAT have most purchases and customers who joined through KFC have more purchased than KAT on mas particular account and junior account.

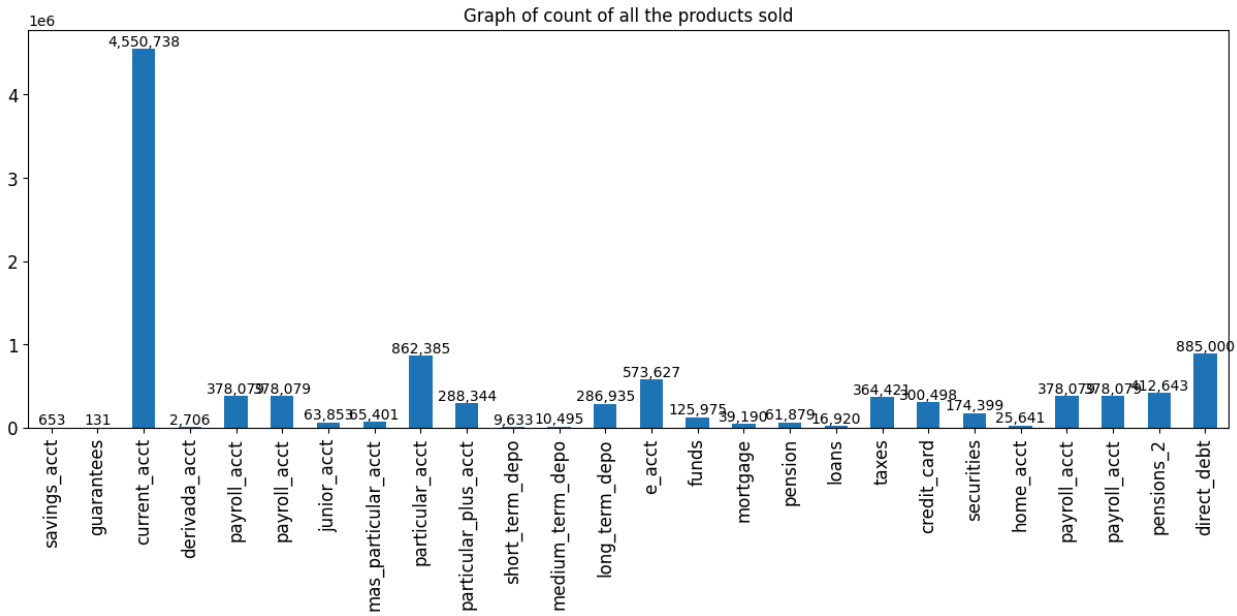


5. Geographic Distribution





6. Count of all the products sold:



---

3. Data Preparation

Data preprocessing is a critical phase in any machine learning project, as it ensures that the raw data is ready to be used for model training. This project followed a comprehensive preprocessing pipeline, as described below.

3.1. Handling Missing Values

- **Missing Data Identification**: The dataset was first examined for missing values. The code used `isnull().sum()` to identify any missing values across the dataset.
- **Imputation Strategy**:
    - Numerical features with missing values (e.g., `income`) were imputed with the **mean** or **median** values based on the feature's distribution.
    - Categorical variables with missing data were filled using the **mode** (most frequent category). This approach helped to retain the data's integrity without losing any valuable information due to missing entries.

3.2. Feature Engineering

- Several new features were created to provide additional insights for the modeling process:
    - **`income_to_product_ratio`**: This was calculated as `income / total_products`, representing the customer's income relative to the number of products they hold.
    - **`income_to_age`**: This was derived by dividing the `income` by the `age`, which could give insights into how income relates to different age groups.

These engineered features were included to capture important relationships between customer characteristics and their product portfolio.

3.3. Categorical Data Transformation

- **One-Hot Encoding** was applied to categorical variables like `join_channel` and `province_name`, transforming them into binary columns for each category. This step ensures that categorical data can be fed into machine learning models without introducing bias from the inherent ordering.
- **Label Encoding** was used on variables like `sex_H` and `cust_type`, as these variables have a relatively small number of categories and don't require one-hot encoding. This encoding technique converts the categories into numeric labels.

3.4. Scaling Numerical Features

- **Standardization** was applied to continuous numerical features, including `income` and `age`, using `StandardScaler` from `sklearn`. This step ensures that all features are on a similar scale, which is essential for models like K-Nearest Neighbors (KNN) and other distance-based algorithms.
- The transformation ensures that the data follows a normal distribution with a mean of 0 and a standard deviation of 1.

3.5. Outlier Detection

- During the exploratory data analysis (EDA), outliers were identified in certain variables (e.g., `income`), but no actions were taken as the outliers were considered realistic and part of the natural data distribution. The decision to leave these values unmodified was based on domain knowledge.

3.6. Data Splitting

- The dataset was split into **80% training** and **20% testing** using `train_test_split` from `sklearn.model_selection`. This division ensures that the model is trained on a majority of the data while still having an independent set of data to evaluate its performance.
- The splitting process was randomized, but stratification was used to ensure that the distribution of categorical variables, such as `cust_type`, was preserved across the training and testing datasets.

---

4. Model Development

4.1. Model Selection

In this project, after experimenting with several models, we ultimately chose **Collaborative Filtering (Logistic Regression-based approach)** as the best model for predicting product adoption. While we initially experimented with models like **XGBoost** and **Random Forest (RF)**, it was Collaborative Filtering that consistently performed the best in terms of **ROC AUC** and **F1 Score** during validation, which made it the ideal choice.

4.2. Model Selection and Rationale

The decision to use **Collaborative Filtering** was based on the specific nature of the problem: predicting the likelihood of product adoption by customers. Collaborative filtering is a widely used technique in recommendation systems that identifies patterns in customer behavior, such as which products are likely to be adopted based on the behaviors of similar customers. Unlike models that rely on detailed product features, collaborative filtering leverages the **interaction data** between customers and products to generate predictions.

In our case, the model performed well because it was able to capture the **relationships between customer behavior** and make accurate predictions on unseen data. This approach does not require detailed product attributes, which aligns perfectly with our dataset, as it contains predominantly **behavioral features** (e.g., product usage, customer demographics) rather than detailed product information.

4.3. Hyperparameter Optimization

Collaborative filtering in this project was implemented using **Logistic Regression** and didn't require extensive hyperparameter optimization. However, we focused on selecting the **best-performing variation** of the collaborative filtering model based on its performance metrics. The final model (Variation 3) showed the highest **ROC AUC score** and was able to handle the **class imbalance** effectively, making it an optimal choice for our prediction task.

```
# Define hyperparameter variations
hyperparameter_variations = [
    {'C': 0.01, 'solver': 'liblinear', 'max_iter': 100},
    {'C': 1, 'solver': 'lbfgs', 'max_iter': 200},
    {'C': 10, 'solver': 'liblinear', 'max_iter': 300},
]
```

```
Summary Table:
  Variation Dataset  Avg ROC AUC  Avg F1 Score
Variation 1   train     0.827547      0.075518
Variation 1     val     0.833304      0.085716
Variation 2   train     0.887300      0.110169
Variation 2     val     0.907706      0.119564
Variation 3   train     0.888038      0.110746
Variation 3     val     0.909773      0.118087

Best Model For This Week: Variation 3
```

4.4. Performance Metrics for Collaborative Filtering

We evaluated the model on the **test dataset**, and the following performance metrics were observed:

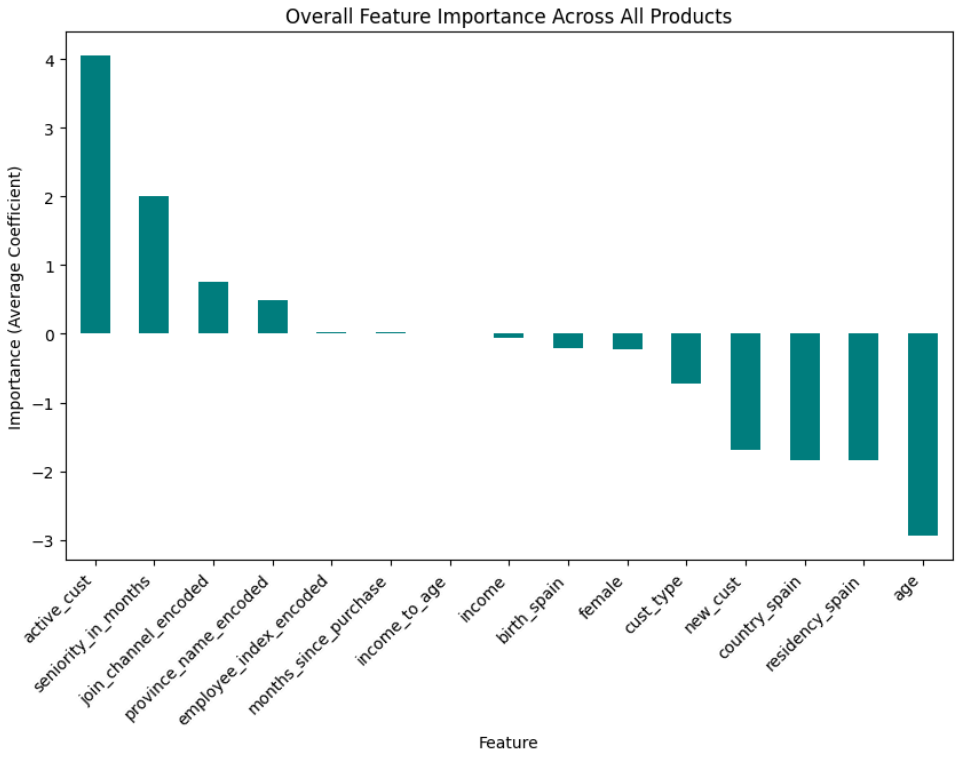
- **Average ROC AUC Score**: 0.8831
  The ROC AUC score indicates how well the model can distinguish between customers who will adopt a product and those who will not. An ROC AUC score of 0.8831 suggests that the model performs well across various thresholds.
- **Average F1 Score**: 0.2014
  The **F1 score** was significantly higher on the **test set** compared to the training and validation sets. Although the model showed some difficulty with class imbalance during training and validation, it performed better in predicting unseen data, especially for precision and recall trade-offs. This improvement in F1 score on the test set indicates the model's robustness and ability to generalize well to new data.

Product	ROC AUC	F1 Score
savings_acct	0.8784	0.0000
guarantees	0.9692	0.0000
current_acct	0.7454	0.7896
derivada_acct	0.8503	0.0039
payroll_acct	0.8640	0.2868
junior_acct	0.9996	0.8862
mas_particular_acct	0.8396	0.0000
particular_acct	0.8831	0.1925
particular_plus_acct	0.8101	0.0006
short_term_depo	0.9399	0.1394
medium_term_depo	0.8942	0.0000
long_term_depo	0.9239	0.2238
e_acct	0.8570	0.3687
funds	0.9190	0.2889
mortgage	0.9238	0.0419
pension	0.9201	0.1304
loans	0.8322	0.0000
taxes	0.8532	0.0681
credit_card	0.8862	0.2446
securities	0.9110	0.2633
home_acct	0.8856	0.0000
pensions_2	0.8600	0.2513
direct_debt	0.8656	0.4523

4.5. Insights from Collaborative Filtering

Key features driving the predictions in the Collaborative Filtering model were the **customer-product interactions**, including customer behavior patterns and the purchase history of similar customers. This highlights that **customer preferences** and **interaction history** were strong indicators of future product adoption.

Customers with higher **income** and **seniority** were more likely to own multiple products, while younger customers tended to prefer **starter accounts**, such as credit cards or savings. This behavior is consistent with what we observe in typical recommendation systems, where products with a wider appeal are often recommended to customers who share similar behaviors to others who have already adopted those products.



---

5. Model Evaluation and Validation

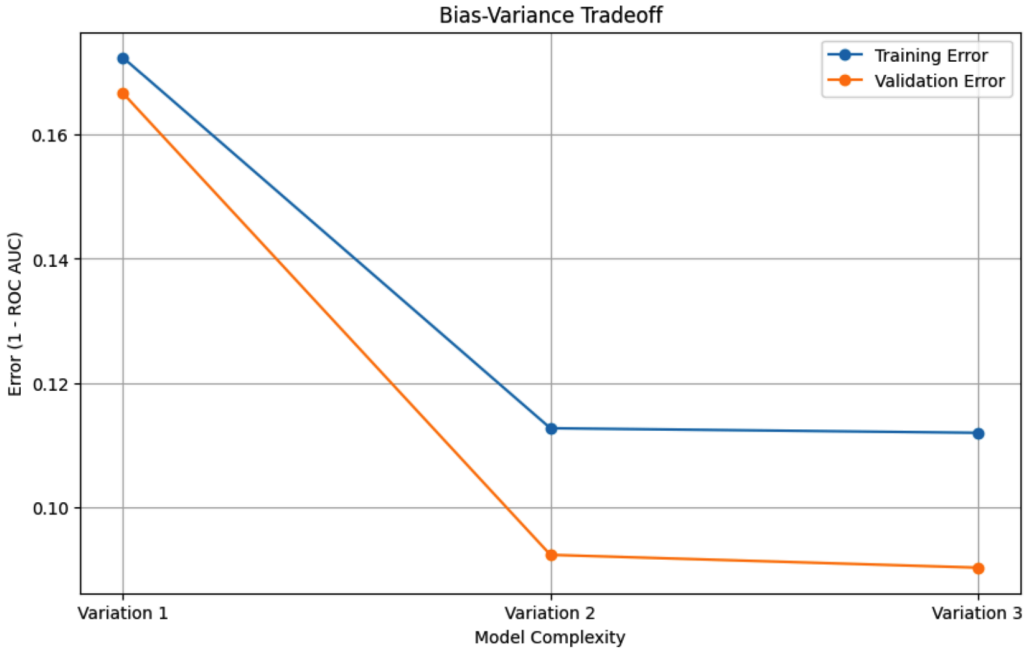
The model was evaluated on unseen validation data to ensure generalizability.

Based on the validation error on the validation dataset, the winning model was the variation 3 of the Collaborative filtering model. The variation 3 of the collaborative filtering model had the best average ROC AUC score. The model that presented the best F1 scores was the XGBoost model, however, the ROC AUC score was lower than the other models.

We believe it was the best model because it was able to capture the relationships between clients behavior and predict well on unseen data. This is an approach frequently used on product recommendation systems as it compares the clients behaviors with previous clients. It takes advantage of customer interactions with products and similar customer's behavior, to help predict the possibility that a customer will buy a product. This approach is effective because it doesn't require detailed product features, and instead relies on the collective preferences of users, which in this case, was enough to identify the patterns in the dataset.

The F1 Score is quite low on both the Train and Validation sets, but it improves substantially on the Test set. This suggests that while the model may have struggled with class imbalance or precision/recall trade-offs during training and validation, it performed better on unseen test data.

We improved the model by adding additional preprocessing steps.



Change #1: Instead of dropping the duplicates on the customer code column and using only the last instance we will keep those duplicates since it could capture some patterns such as if a client buys product x first, it will likely buy y product next.
Change #2: Add the date column as one of the features. For that, we will calculate the time since purchase using the month we are trying to predict: June 2016. For this transformation to make sense, we will also keep the first transformation, since the timeline of purchase matters now, we will keep the duplicate clients' purchases instead of only keeping the last one.
Change #3: We will scale our features using MinMaxScaler from sklearn.preprocessing before training. This could improve convergence and the stability of the model. We were between using MinMaxScaler or StandardScaler, however, since our data is not normal distributed, we will use MinMaxScaler

Dataset	Type	Avg ROC AUC	Avg F1 Score
train	train	0.888038	0.110746
train	test	0.883099	0.201405
train_1	train	0.885656	0.111385
train_1	test	0.883373	0.207875
train_2	train	0.885926	0.111536
train_2	test	0.883623	0.212467
train_3	train	0.885885	0.111541
train_3	test	0.883519	0.205022

- **Model 2** has shown a slight but consistent improvement in **F1 Scores** across test sets when compared to the previous week's model.
- The ROC AUC scores remain stable, and the model appears to generalize well across all datasets.

The inclusion of additional features (such as tracking months since last purchase) seems to improve performance, especially in handling **F1 Scores** during evaluation on the test set.

Feature Importance

- **Top Positive Contributors**:
  - `active_cust`: The highest positive influence (+4.0).
  - `seniority_in_months`: Significant positive influence (+2.0).
  - `join_channel_encoded` and `province_name_encoded`: Moderate positive contributions (between 0.5-1.0).
- **Neutral Features**: Minimal impact near zero.
  - `employee_index_encoded, months_since_purchase, income_to_age, income`.
- **Top Negative Contributors**:
  - `age`: Strongest negative correlation (-3.0).
  - `residency_spain` and `country_spain`: Significant negative impact (-2.0).
  - `new_cust` and `cust_type`: Moderate negative influence.

---

6. Deployment and Maintenance

6.1. Deployment Plan

The model is deployed in **batch mode** due to its monthly recommendation nature. For **existing clients**, recommendations are updated biweekly or monthly, ensuring a balance between computational efficiency and proactive engagement. To address **new clients**, batch predictions are run daily to provide immediate recommendations. This deployment strategy minimizes real-time overhead while maintaining timely and relevant recommendations.

6.2. Environment and Dependencies

The deployment environment includes:

- **Operating System**: Windows 10 (Version 10.0.19045)
- **Python Version**: 3.11.5
- Libraries: Pandas (2.2.2), NumPy (1.24.3), Seaborn (0.13.1), Matplotlib (3.8.0), Scikit-learn (1.3.0), Joblib (1.3.2).

6.3. Performance Metrics and Monitoring Plan

Key metrics are tracked to ensure the model's effectiveness:

1. **Model Metrics**:
    - ROC AUC: Assesses classification capability (Current Avg: 0.884).
    - F1 Score: Evaluates precision-recall balance (Current Avg: 0.212).
2. **Business Metrics**:
    - Conversion Rate: Measures successful recommendations.
    - Error Rate: Highlights missed opportunities.
    - Negative Hits: Tracks irrelevant or frustrating recommendations.
3. **Thresholds**:
    - **Green Flags** (Optimal Performance):
        - ROC AUC ≥ 0.85, F1 Score ≥ 0.20, Conversion Rate ≥ 15%, Error Rate ≤ 5%, Negative Hits ≤ 2%.
    - **Yellow Flags** (Moderate Performance):
        - ROC AUC 0.75–0.85, F1 Score 0.15–0.20, Conversion Rate 10–15%, Error Rate 5–10%, Negative Hits 2–5%.
    - **Red Flags** (Critical Underperformance):
        - ROC AUC < 0.75, F1 Score < 0.15, Conversion Rate < 10%, Error Rate > 10%, Negative Hits > 5%.

6.4. Maintenance Plan

- **Retraining Frequency:** The model is retrained monthly to incorporate new data, address data drift, and reflect customer behavior changes. This ensures adaptability to evolving trends.
- **Risk Mitigation Strategies**:
    - **Yellow Flags**: Investigate potential data or feature drift; review input data and feature distributions.
    - **Red Flags**: Temporarily remove the model, retrain with updated data, and evaluate misclassifications.
    - **Monitoring Data Drift**: Track statistical properties of features and client demographics using exploratory data analysis (EDA).

This structured deployment and maintenance plan ensures the model remains efficient, relevant, and aligned with business goals while minimizing risks associated with drift or underperformance.

Conclusion

This project successfully developed a personalized product recommendation system tailored for financial services. Using logistic regression, the model integrates behavioral, demographic, and account-related features to predict client preferences with high interpretability. Key predictors such as customer activity level, tenure, and income-to-product ratios were instrumental in aligning recommendations with customer needs and business goals. Ethical considerations were incorporated to ensure fairness and prevent biases in predictions, fostering equitable outcomes across diverse customer segments.

The deployment strategy, designed for batch processing, balances computational efficiency with responsiveness by providing regular updates to recommendations. A structured monitoring framework with clearly defined thresholds ensures ongoing evaluation of model performance, while monthly retraining schedules address potential data and concept drift, maintaining the system's adaptability to evolving trends and client behaviors.

Overall, the project successfully aligns technical rigor with business objectives, delivering a reliable, interpretable, and actionable solution for improving client engagement and driving product adoption. Future iterations may explore incorporating advanced modeling techniques or real-time inference capabilities to further enhance system performance and responsiveness.
