ML Engineer / Data Scientist Task: Call Data Scoring Model

Task Overview

You are provided with a dataset containing **call records** from a call center, enriched with geo-location and census/IRS (US public database of tax information) information. Some calls in the dataset are labeled as confirmed sales. Your goal is to build a predictive model that scores the likelihood of each call resulting in a sale. This exercise is intended to evaluate your approach to data understanding, feature engineering, model building, and overall solution design.

Data Description

1. **Call Data**
   - Call specifics
   - Labels indicating whether a sale was confirmed or not
   - Note: The distribution of sales vs. non-sales may not be balanced
2. **Geo Data**
   - Location-based features
3. **Census Data**
   - Demographic or socioeconomic attributes (e.g., average income in the region, population density)

A **Data Dictionary** will be provided, detailing each column’s meaning and data type.

Task Requirements

1. **Data Exploration & Analysis**
   - Perform an initial exploration of the dataset.
     - Look for missing values, outliers, and data distributions.
     - Investigate any imbalance in the target variable.
     - Summarize your findings.
2. **Feature Engineering**
   - Propose ways to transform or engineer new features from the existing data.
   - Handle missing or inconsistent data systematically.
   - Discuss any domain-specific insights you might leverage.
3. **Model Development**
   - **Model Selection**
     - Pick at least one algorithm to predict whether a call will result in a sale.
     - Explain why you chose this model and how you might compare or consider alternatives.
   - **Training & Validation**
     - Describe your overall training approach, including data splitting and validation methodology.
     - Explain any relevant considerations for ensuring robust and unbiased performance estimates.
   - **Handling Imbalanced Data**
     - If the target variable is imbalanced, discuss how you address it.
     - Justify your chosen technique and its impact on your model’s performance.
   - **Hyperparameter Tuning**
     - Document your strategy for optimizing model parameters.
     - Summarize key hyperparameters and how you selected the final values.

4. **Performance Evaluation**
    - Define relevant metrics (e.g., precision, recall, F1, ROC-AUC) and justify your choice(s).
    - Evaluate the model's performance and discuss strengths and weaknesses, especially with respect to handling the imbalance.
5. **Solution Integration**
    - Outline a plan for integrating the model into a production environment (e.g., scoring API endpoint).
    - Discuss how you would handle model updates, real-time or batch predictions, and potential data shifts over time.
6. **Documentation & Deliverables**
    - Summarize your approach, assumptions, and conclusions.
    - Include a brief explanation of each step (data cleaning, feature engineering, model selection, etc.).
    - Include code snippets or pseudo-code illustrating key steps (training, scoring, etc.).
7. **Bonus (Optional)**
    - Propose methods for monitoring the model post-deployment (e.g., drift detection, performance monitoring).
    - Suggest how you would optimize or scale the solution with larger datasets.

Outcome

This exercise aims to assess your ability to:

- **Understand** new data and problem statements quickly.
- **Apply** data science best practices in feature engineering and modeling.
- **Handle** real-world challenges like class imbalance and missing values.
- **Communicate** your process and findings effectively.
- **Propose** a solution that can be integrated and scaled.

Data to use

ml_task_data.csv

Data Dictionary

Column Name	Data Type	Example Value	Description
phone	*String*	`8438643371`	Phone number used for the call.
supplier	*String*	`10234`	Supplier of the call.
call_timestamp	*Datetime*	`2024-10-31 17:08`	Timestamp indicating when the call occurred.
call_day_of_week	*Integer*	`5`	Numeric representation of the call day.
call_time_morning_or_afternoon	*String*	`Afternoon`	Indicator whether the call was made in the morning or afternoon.
call_week_of_month	*Integer*	`43`	Numeric representation of the week of the year

target	*Integer*	`0`	Binary target variable indicating if the call resulted in a sale (`1`) or not (`0`).
zipcode	*Integer*	`65305`	ZIP code associated with the call or lead.
Estimate_Households_Total	*Integer*	`1008`	Estimated total number of households in this ZIP code or region (from census data).
Estimate_Households_Median_income_usd	*Integer*	`58895`	Estimated median household income in USD (from census data).
Estimate_Households_Mean_income_usd	*Integer*	`...`	Estimated mean (average) household income in USD.
Estimate_Families_Total	*Integer*	`...`	Estimated total number of families in the region.
Estimate_Families_Median_income_usd	*Integer*	`...`	Estimated median family income in USD.
Estimate_Families_Mean_income_usd	*Integer*	`...`	Estimated mean family income in USD.
Estimate_Married-couple_families_Total	*Integer*	`...`	Estimated total number of married-couple families.
Estimate_Nonfamily_households_Total	*Integer*	`...`	Estimated total number of non-family households (e.g., single-person households or unrelated individuals living together).
Estimate_Nonfamily_households_Median_income_usd	*Integer*	`...`	Median income for non-family households.
Estimate_Nonfamily_households_Mean_income_usd	*Integer*	`...`	Mean income for non-family households.
h_zipcode	*Integer/String*	`...`	ZIP code.
Estimate_Total_*	*Integer*	`...`	Columns beginning with `Estimate_Total_` reflect demographic counts (e.g., total population within certain income brackets, health insurance categories, etc.).

Estimate_Total__With_health_insurance_coverage	*Integer*	`...`	Estimated count of individuals or households in category `<x>` that have health insurance coverage.
Percent_<...>	*Float*/*String*	`100.0`, `2.5`, `(X)`	Columns beginning with `Percent_` represent the percentage (or ratio) of a specific demographic or socioeconomic characteristic. A value of `(X)` indicates missing data or an estimate that could not be calculated.
state	*String*	`MO`	State abbreviation (e.g., Missouri).
countyname	*String*	`Johnson County`	County name corresponding to the ZIP code.