

AI Engineer Take-Home Assignment: Azure ML Lead Scoring Weights

Overview

You will design and prototype a machine-learning–driven weighting system for a deterministic lead scoring engine used by a mass SMS/MMS marketing platform. The current system applies manually tuned weights to engineered features (e.g., recency, fatigue, sentiment). Your task is to design an Azure-based ML solution that learns optimal weights from client feedback and can be integrated into the existing deterministic scoring pipeline.

This assignment is scoped to **3–5 hours**. Focus on clarity, sound engineering choices, and reproducibility rather than production-hardening everything.

Data (Provided)

You will receive:

- train.csv — historical features with a label column (positive reaction).
- valid.csv — validation split with the same schema.
- feature_dictionary.md — feature descriptions and intended semantics.

If you do not receive data, simulate a small dataset (document your assumptions).

Objectives

1. **Modelling**
 - Train at least one interpretable model (e.g., Logistic Regression).
 - Optionally train a non-linear model (e.g., XGBoost/LightGBM).
2. **Weight Derivation**
 - Convert model outputs into deterministic feature weights suitable for SQL scoring.
 - Provide a clear mapping or normalization strategy.
3. **Evaluation**
 - Evaluate using appropriate metrics (AUC/PR-AUC, calibration, precision@K).
 - Compare model-driven weights to a baseline of equal weights.
4. **Azure Architecture Design**
 - Propose an Azure ML pipeline for training, evaluation, and weight updates.
 - Include data storage (e.g., ADLS/SQL), orchestration, and monitoring components.

Deliverables

1. **Short Design Doc** (1–3 pages)
 - Architecture diagram or bullet-point flow is fine.
 - How data moves from ingestion → features → training → weights → SQL scoring.
 - How retraining and monitoring are handled in Azure.
2. **Notebook or Script**
 - Loads data, trains model(s), evaluates, and exports weights.
 - Include a small function (or pseudo-SQL) showing how weights map into a deterministic score.
3. **Results Summary**
 - Metrics table and interpretation.
 - Final weight table (feature → weight).
4. **README**
 - How to run your work locally.

Technical Requirements

- Python (3.9+), scikit-learn (at minimum).
- Ensure reproducibility (random seeds, clear data prep).
- Keep PII out of artifacts.

Evaluation Rubric

Category	What We Look For
ML Fundamentals	Reasonable feature handling, model choice, evaluation
Weight Mapping	Clear, justifiable conversion from model to weights
Azure Design	Correct Azure components and pipeline flow
Clarity & Reproducibility	Clean code, concise explanations, easy to run

Bonus (Optional)

- Drift monitoring strategy with concrete metrics.
 - Explainability using SHAP or feature importances.
 - Simple A/B testing plan to validate business impact.
-

Submission

Provide a zip or repo link containing:

- design_doc.md
- notebook.ipynb or train.py
- weights.csv
- README.md