DATA MANAGEMENT FOR MACHINE LEARNING – ASSIGNMENT I

Submission Date: 24 Aug 2025 11.50 PM

Weightage: 20%

Title: End-to-End Data Management Pipeline for Machine Learning

Objective:

- Design, implement, and orchestrate a complete data management pipeline for a machine learning project, addressing all stages from problem formulation to pipeline orchestration.

Business Context:

Customer churn occurs when an existing customer stops using a company's services or purchasing its products, effectively ending their relationship with the company. While certain types of churn, such as those resulting from unavoidable circumstances like death, are considered non-addressable, this discussion focuses on addressable churn—scenarios where intervention could prevent customer loss.

Churn poses significant challenges for businesses, leading to revenue declines and increased pressure on teams to compensate for the loss. One approach to offset churn is acquiring new customers, but this is both costly and difficult. High customer acquisition costs further strain the company's overall revenue. Additionally, churn has indirect effects: former customers often turn to competitors, potentially influencing other loyal customers to follow suit.

A recent research note from PWC highlights the gravity of this issue:

> **"Financial institutions will lose 24% of revenue in the next 3-5 years, mainly due to customer churn to new fintech companies."**

Given these challenges, reducing customer churn has become a critical business strategy for most organizations. Even when it's not explicitly a strategic objective, retaining existing customers is always in the company's best interest.

You are working as a Data Engineer for a startup specializing in predictive analytics. The company aims to build a robust automated pipeline to process customer data collected from multiple sources (e.g., web logs, transactional systems, and third-party APIs) for a machine learning model that predicts customer churn. Your task is to design and implement this pipeline while adhering to best practices for data management.

Tasks:

1. Problem Formulation

- Clearly define the business problem
- Identify key business objectives
- List the key data sources and their attributes
- Define the expected outputs from the pipeline:
    - Clean datasets for exploratory data analysis (EDA)
    - Transformed features for machine learning
    - A deployable model to predict customer churn
- Set measurable evaluation metrics
- **Deliverables:**
    - A PDF/Markdown document with the business problem, objectives, data sources, pipeline outputs, and evaluation metrics.

2. Data Ingestion

- Identify at least two data sources (e.g., CSV files, REST APIs, database queries)
- Design scripts for data ingestion, ensuring:
    - Automatic fetching of data periodically (e.g., daily or hourly)
    - Error handling for failed ingestion attempts
    - Logging for monitoring ingestion jobs
- **Deliverables:**
    - Python scripts for ingestion (e.g., using pandas, requests etc.)
    - A log file showing successful ingestion runs
    - Screenshots of ingested data stored in raw format

3. Raw Data Storage

- Store ingested data in a data lake or storage system (e.g., AWS S3, Google Cloud Storage, HDFS, or a local filesystem)
- Design an efficient folder/bucket structure:
    - Partition data by source, type, and timestamp
- **Deliverables:**
    - Folder/bucket structure documentation
    - Python code demonstrating the upload of raw data to the storage system

4. Data Validation

- Implement data validation checks to ensure data quality:
    - Check for missing or inconsistent data
    - Validate data types, formats, and ranges
    - Identify duplicates or anomalies
- Generate a comprehensive data quality report
- **Deliverables:**
    - A Python script for automated validation (e.g., using pandas, great_expectations, or pydeequ)
    - Sample data quality report in PDF or CSV format, summarizing issues and resolutions

5. Data Preparation

- Clean and preprocess the raw data:
    - Handle missing values (e.g., imputation or removal)
    - Standardize or normalize numerical attributes
    - Encode categorical variables using one-hot encoding or label encoding

- Perform EDA to identify trends, distributions, and outliers.
- **Deliverables:**
  - Jupyter notebook/Python script showcasing the data preparation process
  - Visualizations and summary statistics (e.g., histograms, box plots)
  - A clean dataset ready for transformations

6. Data Transformation and Storage

- Perform transformations for feature engineering:
  - Create aggregated features (e.g., total spend per customer)
  - Derive new features (e.g., customer tenure, activity frequency)
  - Scale and normalize features where necessary
- Store the transformed data in a relational database or a data warehouse.
- **Deliverables:**
  - SQL schema design or database setup script
  - Sample queries to retrieve transformed data
  - A summary of the transformation logic applied

7. Feature Store

- Implement a feature store to manage engineered features:
  - Define metadata for each feature (e.g., description, source, version)
  - Use a feature store tool (e.g., Feast) or a custom solution
- Automate feature retrieval for training and inference
- **Deliverables:**
  - Feature store configuration/code
  - Sample API or query demonstrating feature retrieval
  - Documentation of feature metadata and versions

8. Data Versioning

- Use version control for raw and transformed datasets to ensure reproducibility:
  - Track changes in data using tools like DVC, Git LFS, or a custom tagging system
  - Store version metadata (e.g., source, timestamp, change log)
- **Deliverables:**
  - DVC/Git repository showing dataset versions
  - Documentation of the versioning strategy and workflow

9. Model Building

- Train a machine learning model to predict customer churn using the prepared features:
  - Use a framework like scikit-learn or TensorFlow
  - Experiment with multiple algorithms (e.g., logistic regression, random forest)
  - Evaluate model performance using metrics such as accuracy, precision, recall, and F1 score
- Save the trained model using a versioning tool (e.g., MLflow)
- Deliverables:
  - Python script for model training and evaluation
  - Model performance report
  - A versioned, saved model file (e.g., .pkl, .h5)

10. Orchestrating the Data Pipeline

- Automate the entire pipeline using an orchestration tool (e.g., Apache Airflow, Prefect, or Kubeflow):
  - Define a Directed Acyclic Graph (DAG) for pipeline tasks.
  - Ensure task dependencies are well-defined (e.g., ingestion → validation → preparation).
  - Monitor pipeline runs and handle failures gracefully.
- **Deliverables:**
  - Pipeline DAG/script showcasing task automation
  - Screenshots of successful pipeline runs in the orchestration tool
  - Logs or monitoring dashboard screenshots

Additional Instructions:

- **Ensure modularity in your codebase, with separate scripts for each stage.**
- **Use proper logging and error handling in all scripts.**
- **Provide detailed documentation, including:**
  - **Explanation of the pipeline design.**

- Challenges faced and solutions implemented.

- Submit a short video (5–10 minutes) demonstrating your pipeline workflow. – This is very important without which evaluation will not happen. So do not miss this.

Submission Requirements:

- Source Code: Organized into folders by stage.
- Documentation: Markdown or PDF format.
- Video Walkthrough: Demonstrating the pipeline.
- Final Deliverables: Compressed .zip file with all code, data, and documentation.

General Notes:

- Although specific tools, products, and platforms are mentioned as examples in the tasks, you are free to choose and justify a toolchain of your preference, provided it aligns with the objectives, expectations, and deliverables of the assignment.
- Refer the document used while registering the groups. In case of discrepancies, write to me separately (copying all your group members) with subject line as "Cluster DM4ML Group <your_group_number>". email – sunilbhutada@wilp.bits-pilani.ac.in
- Using the LMS, only one member of group has to upload the file. No submission over email will be considered.
- Make sure that you upload the file well ahead of deadline. At last moments, we have seen several groups have faced issues while doing the submissions.
- Note - As it's a group assignment, only one submission is expected from each group. Unnecessary don't upload the solution on individual basis. If it's observed, then the penalty (25% reduction) will be applicable on it.