Detailed Documentation: End-to-End Data Management Pipeline for Machine Learning

# Explanation of the Pipeline Design Overview

The objective of this pipeline is to design, implement, and orchestrate a complete data management pipeline for customer churn prediction. The pipeline encompasses the full lifecycle of data management, from ingestion to orchestration, ensuring data quality and model reliability.

# Pipeline Architecture

The pipeline follows a modular architecture with the following stages:

## 1. Problem Formulation

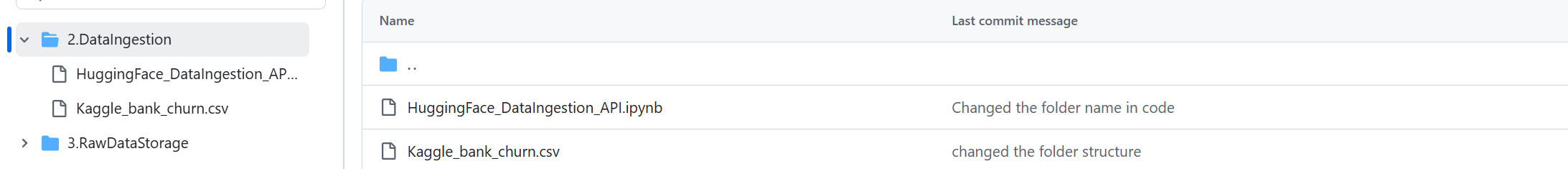
## The goal is to predict customer churn to help businesses take early retention actions. We use two datasets: Kaggle\_bank\_churn.csv (bank customer churn data) and HuggingFace\_DataIngestion\_API.ipynb (churn dataset via API). The expected outputs are clean datasets, engineered features, and a deployable churn prediction model.

## 2. Data Ingestion

## Data ingestion is automated from two sources:

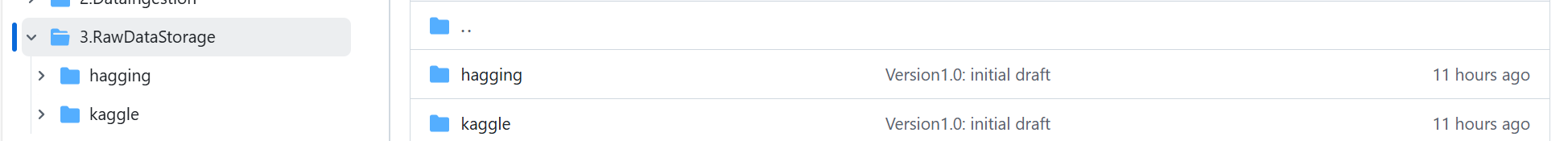
## Kaggle\_bank\_churn.csv – downloaded directly from Kaggle.

## HuggingFace\_DataIngestion\_API.ipynb – uses the Hugging Face API to fetch churn data in CSV format.



## 3. Raw Data Storage

## In this project, the ingested datasets are first stored in the Colab environment for processing. To ensure versioning and accessibility, the processed data is further uploaded and maintained in GitHub. This acts as a simple data lake substitute, where data is organized by source (Kaggle/Hugging Face) and processing stage (raw, validated, prepared)



### 4. Data Validation

### After storing the raw datasets from Kaggle and Hugging Face in Colab and versioning them on GitHub, validation checks are performed to ensure data quality.

A screenshot of a computer

AI-generated content may be incorrect.

* **5. Data Preparation**
* Once validated, both the Kaggle and Hugging Face datasets are cleaned and transformed in Colab using a single unified preprocessing script. This step includes handling missing values, encoding categorical attributes, and standardizing numerical features. By applying a consistent preparation pipeline across both sources, we ensure that the final datasets are uniform, reliable, and ready for feature engineering.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a report

AI-generated content may be incorrect.

## 6. Data Transformation & Storage

## After preparation, both datasets undergo feature engineering and transformation in Colab. New features such as tenure groups, average charges, and contract/payment encodings are created to capture customer behavior more effectively. The transformed data is then stored in a feature store (SQLite/CSV in Colab, with versioning on GitHub). This centralized storage ensures that engineered features are easily accessible for both model training and future reuse in production.

## A screenshot of a computer AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

## 7. Feature Store

## To manage and reuse engineered features, a feature store is implemented using SQLite within Colab. Key customer attributes such as tenure, monthly charges, total charges, contract type, payment method, and churn labels are stored in a structured database table. This setup allows consistent access to curated features during both model training and future iterations, while GitHub serves as the version-controlled backup for the feature store files.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

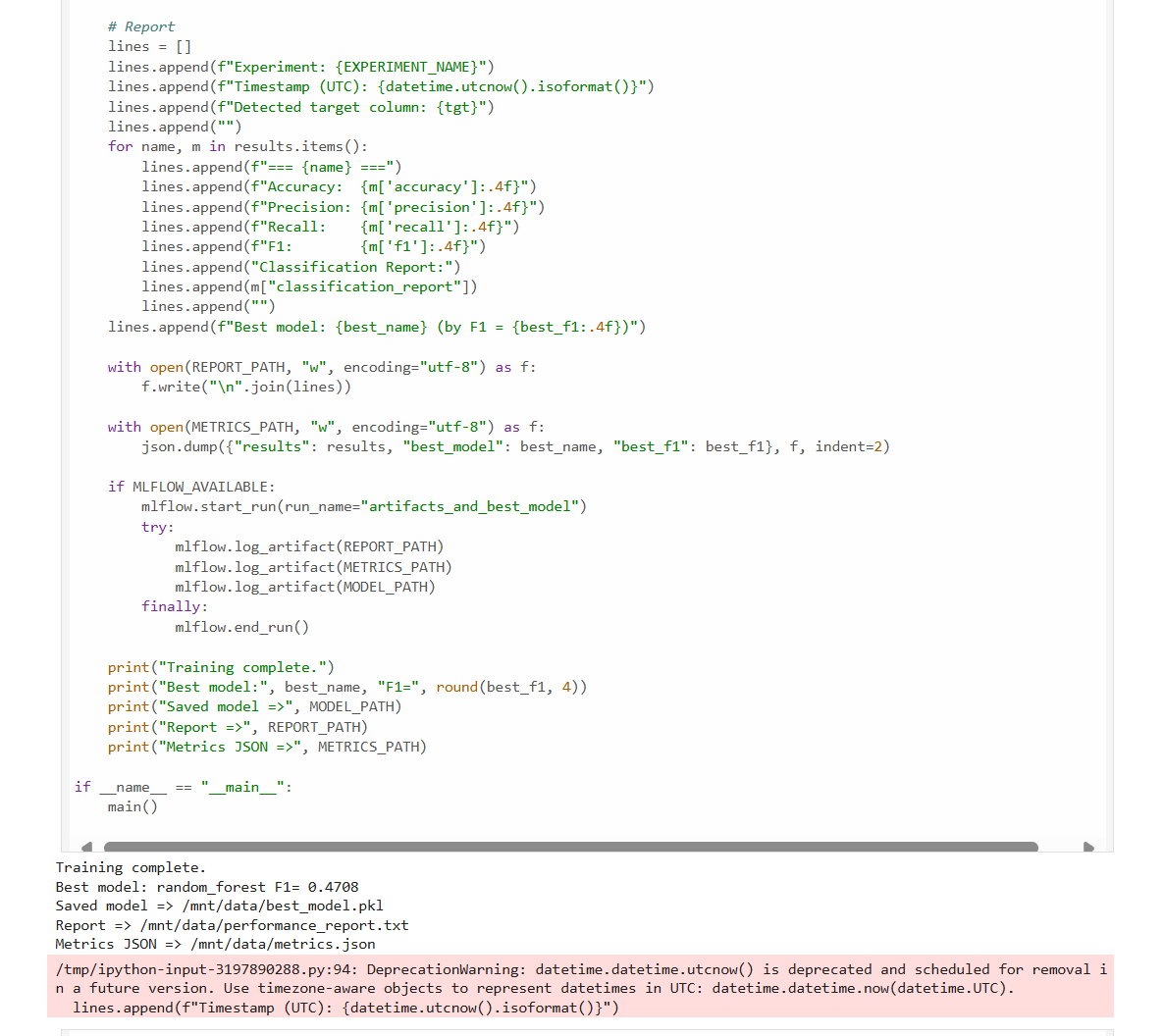
8. Data versioning

We maintained versioning by saving datasets, features, and models with version numbers These files are visible in the Colab directory and tracked in GitHub commit history.

https://github.com/naveenpahal31/DataManagement

9. Model Building

Using the processed and feature-engineered datasets from Kaggle and Hugging Face, machine learning models are developed in Colab. We experiment with algorithms such as Logistic Regression and Random Forest, splitting the data into training and test sets to evaluate performance. Metrics like accuracy, precision, recall, and F1-score are used to measure model effectiveness. The best-performing model is then saved and tracked via GitHub, ensuring it can be reused or further improved in later iterations.



A white background with black text

AI-generated content may be incorrect.

* **10**. Pipeline Orchestration

The entire workflow—from data ingestion (Kaggle + Hugging Face) to validation, preparation, transformation, feature storage, versioning, and model training—is orchestrated to run seamlessly in Colab with tracking on GitHub. Logging statements provide transparency for each task, and error handling ensures failed ingestions or validations can be retried. While Colab scripts manage execution in this project, tools like Apache Airflow or Prefect can be adopted in production for scheduling, monitoring, and visualizing dependencies through DAGs, ensuring a fully automated and reliable pipeline.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer code

AI-generated content may be incorrect.

A graph with blue bars

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.