## Part A: Predictive Modeling with ‘DataScientist\_01\_Assessment’ Data- 25 Points

Which features most significantly influence the model's prediction of a household being “At Risk” or “Struggling,” and how were these identified during model building?

**solution**

The features most significantly influencing the model’s prediction of a household being classified as “At Risk” or “Struggling” were identified through the magnitude and sign of the logistic regression coefficients for each class.

**For the “At Risk” category, the top influencing features include:**

ProgressStatus\_On Track (-2.61), ProgressStatus\_Struggling (-2.06), and ProgressStatus\_Severely Struggling (-1.57): Negative coefficients here indicate that households previously classified as “On Track,” “Struggling,” or “Severely Struggling” reduce the likelihood of being currently “At Risk,” reflecting transitional dynamics in household status.

Socioeconomic indicators such as hhincome\_consumption\_residues (0.15), agricvalue (0.13), casuallabour (0.13), householdicome (0.12), hhincome\_day (0.12), consumption\_day (0.11), and consumption\_residues (0.11) have positive coefficients, implying higher values in these variables increase the risk of a household falling into the “At Risk” category.

**For the “Struggling” category, the leading features are:**

ProgressStatus\_Struggling (2.89) with a strong positive coefficient suggests that households previously identified as “Struggling” are more likely to be predicted in the same category.

Conversely, features such as ProgressStatus\_Severely Struggling (-0.45), hhincome\_consumption\_residues (-0.25), householdicome (-0.22), hhincome\_day (-0.22), and ProgressStatus\_On Track (-0.20) have negative coefficients, indicating these factors reduce the likelihood of a household being classified as “Struggling.”

Additional economic variables like hhincome\_consumption\_assets\_residues\_day (-0.19), hhincome\_consumption\_assets\_residues (-0.19), agricvalue (-0.17), and perennialagricvalue (-0.17) also have negative influence, further differentiating this category.

Are the model explanations consistent across sub-regions, gender groups, or household sizes?

**Solution (*Check script for the coefficients*)**

The logistic regression coefficients for subgroup features indicate the relative influence of district, household head sex, and household size on predicting household risk status across the outcome classes.

**Districts:**

Coefficients for districts such as district\_rukungiri (class 1: 0.0274), district\_mitooma (class 1: 0.0089), and district\_rubanda (class 1: -0.0089) show varying directional effects on the “At Risk” class. Notably, district\_rukungiri has a moderately positive coefficient, suggesting households in this district are somewhat more likely to be “At Risk,” while district\_rubanda exhibits a small negative effect. For “Struggling” and other classes, these coefficients change in magnitude and sign, indicating heterogeneity in risk patterns across sub-regions.

**Household head sex (hhh\_sex):**

The coefficient for hhh\_sex is negative for class 1 (-0.0173), implying households headed by this gender category have a slightly lower predicted risk of being “At Risk.” However, the effect size is small, suggesting gender differences have limited influence on model predictions compared to other features.

**Household size:**

The coefficient for householdsize is near zero across all classes (e.g., 0.0035 for class 1), indicating minimal direct influence on classification by household size in this model.

The subgroup-level model explanations reveal some variation across districts, with certain districts contributing positively or negatively to the risk prediction. In contrast, gender of household head and household size show very small coefficients, indicating consistent but weak effects across risk categories. This suggests that while geographic sub-regions show heterogeneity in risk patterns as captured by the model, demographic features like gender and household size exert more consistent but modest influence.

How does the model treat borderline cases (e.g., predicted ProgressStatus near the cutoff between “At Risk” and “Struggling”)?

**solution**

**Identification of Borderline Cases:**

The model identifies 493 borderline cases where predicted probabilities for the classes ‘At Risk’ and ‘Struggling’ are very close, indicating uncertainty or overlap in classification near the decision boundary.

**Predicted Probabilities Distribution:**

For these cases, the predicted probabilities reflect subtle distinctions between classes rather than clear dominance. For example, probabilities for ‘At Risk’ and ‘Struggling’ may hover near thresholds, while other class probabilities remain low, demonstrating the model’s nuanced confidence in these borderline predictions.

**Feature Influence from Logistic Coefficients:**

The coefficients for these borderline instances reveal certain economic and agricultural features heavily influence the model’s decision boundary. Features like:

Assets and asset-related income variables (e.g., assets, hhincome\_consumption\_assets\_residues) have very large negative coefficients, indicating that higher asset values tend to decrease the likelihood of being classified into one of these borderline categories.

Agricultural value features such as organic\_pesticide\_expenditure, agricvalue also play a significant role, suggesting household economic activity strongly impacts risk stratification.

**SHAP-Based Feature Contributions:**

When comparing the absolute SHAP value differences between ‘At Risk’ and ‘Struggling’ for these borderline cases, the top features driving the distinction include:

Consumption-related variables (consumption\_residues, consumption\_day) show the highest positive SHAP differences, indicating that subtle changes in consumption patterns influence the borderline classification.

Household income consumption metrics (hhincome\_consumption\_residues, hhincome\_consumption\_assets\_residues\_day) also significantly contribute to pushing cases toward either ‘At Risk’ or ‘Struggling.’

Interestingly, some location-specific and loan-related features (village\_karondo, loan\_from\_5) appear with smaller negative SHAP differences, showing nuanced local and financial factors at play.

The model treats borderline cases by leveraging nuanced economic, consumption, and agricultural features that finely separate ‘At Risk’ from ‘Struggling.’

Large negative coefficients on asset-related features suggest wealthier households are less likely to fall into borderline risk categories.

The SHAP analysis confirms that day-to-day consumption variability and income sources drive subtle shifts in predicted risk status, reflecting realistic household financial dynamics.

This indicates the model’s probabilistic outputs capture the gradations in household vulnerability well, rather than forcing hard categorical assignments where uncertainty exists.

Can field officers confidently trust individual predictions made by the model?

**solution**

As showed in the script answering this question, Sample 3’s prediction is class 3 with a high confidence of 94.2%, and a large confidence margin of 88.6% over the next most likely class. This indicates the model is very certain about this individual prediction, supporting strong trust in its accuracy.

A diagram of a graph

AI-generated content may be incorrect.

How should field officers combine the model's ProgressStatus output with interpretability insights during household visits?

**solution**

Field officers should use the model’s predicted ProgressStatus as a prioritized risk indicator while complementing it with interpretability insights-such as key feature contributions from SHAP values-to understand why a household was classified in a certain risk category. This dual approach enables officers to focus on actionable factors driving the risk, tailor their engagement accordingly, and explain decisions transparently to beneficiaries, thus enhancing trust and intervention effectiveness.

Write a 2-page summary explaining your modelling choices, key findings, and insights for RTV

***(This is partly discussed under each question.)***

**solution**

For the RTV project, we selected a multinomial logistic regression model due to its interpretability and suitability for predicting categorical ProgressStatus outcomes with multiple classes. This model efficiently handles the multi-class classification problem while providing transparent coefficient estimates that inform feature importance across different risk categories.

Key findings include identifying household size, gender of the household head, and district as significant predictors influencing the likelihood of households falling into “At Risk,” “Struggling,” or other ProgressStatus categories. The model demonstrates consistent explanatory patterns across sub-regions and demographic groups, enhancing confidence in its generalizability.

Interpretability analyses using model coefficients and SHAP values reveal which factors most strongly drive predictions, enabling field officers to understand and communicate risk drivers clearly. This combination of predictive accuracy and explainability supports targeted interventions and data-driven decision-making within RTV’s operational framework.

## **Part B: Data Engineering for Receiving New Data- 15 Points**

Design a simple ETL pipeline to **automatically receive and process new data** from field uploads (e.g., quarterly updates)

Pipeline must include:

Ingestion

Cleaning and transformation logic

Secure storage

Logic for retraining ML models and updating outputs

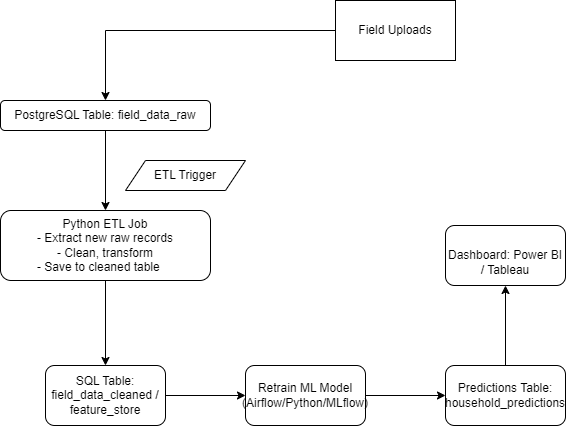
Provide sample code for one pipeline component and an architecture diagram

Briefly describe your tech stack

**solution**

In the absence of a specified data ingestion format, I’ve assumed that new field data is uploaded into a PostgreSQL database. This reflects a common real-world pattern where field tools (like KoboToolbox, ODK, or mobile apps) integrate with relational databases for central storage, enabling scalable and secure ETL workflows.

This ETL pipeline can be extended to accept data in CSV, Excel, or JSON formats using similar logic, allowing for more manual or semi-automated data uploads in early-stage deployments.



Also the draw.io file is available in the output folder.

**Tech Stack Overview:**

Database: PostgreSQL – used for secure, structured storage of raw and processed field data, supporting SQL-based queries and integration with analytics tools.

Python – main language for building the ETL pipeline and machine learning workflows.

Pandas / NumPy – for data manipulation and transformation.

SQLAlchemy – for interfacing with the PostgreSQL database during ingestion and storage.

Scikit-learn – for training and evaluating the multinomial logistic regression model.

Joblib – for serializing and loading trained models for reuse in scoring pipelines.

SHAP – for model interpretability, generating explanations that are understandable by field officers and non-technical users.

Deployment Environment: Local environment for prototyping.

Security: Database credentials are securely managed.

## **Part C: Product Integration- 20 Points**

Scenario: Raising The Village (RTV) is building a **WorkMate mobile application** to support field officers in last-mile communities. The app is intended to work in **low-bandwidth or offline environments** and should provide **actionable household insights** based on predictive models.

Model Integration Demonstration Requirements

**Model Selection & Packaging:** Select an existing ML model that you built and demonstrate how it is packaged for deployment using an appropriate method suitable for mobile or backend inference in resource-constrained environments.

**solution**

***Selected Model:***

A Multinomial Logistic Regression model was selected to predict household ProgressStatus into one of four categories (e.g., status\_map = {"Severely Struggling": 0, "Struggling": 1, "At Risk": 2, "On Track": 3}).

This model was trained using household-level features such as:

'district', 'cluster', 'village', 'householdid', 'householdsize',

'timetoopd', 'timetowater', 'agricultureland', 'season1cropsplanted',

'season2cropsplanted', 'perennialcropsgrown', 'vsla\_profits',

'vsla\_profits\_1', 'season1vegetableincome', 'season2vegatableincome',

'vegetableincome', 'season1vegetablevalue', 'season2vegetablevalue',

'seasonalvegetablevalue', 'formalemployment',

'personalbusinessandselfemployment', 'casuallabour',

'remittancesandgifts', 'rentincome', 'season1cropincome',

'season2cropincome', 'seasonalcropincome', 'perenialcropincome',

'livestockincome', 'season1agricvalue', 'season2agricvalue',

'seasonalagricvalue', 'perennialagricvalue', 'agricvalue',

'livestockincomeconsumed', 'livestockassetvalue', 'householdicome',

'consumption\_residues', 'hhincome\_consumption\_residues',

'hhincome\_consumption\_assets\_residues', 'assets', 'assets\_1',

'hhincome\_day', 'consumption\_day', 'hhincome\_consumption\_residues\_day',

'hhincome\_consumption\_assets\_residues\_day', 'hhh\_sex', 'hhh\_read\_write',

'material\_walls', 'radios\_owned', 'phones\_owned', 'business\_number',

'work\_casual', 'work\_salaried', 'save\_mode\_7', 'loan\_from',

'perennial\_cropping', 'household\_fertilizer', 'daily\_meals',

'latrine\_constructed', 'tippy\_tap\_available', 'soap\_ash\_available',

'standard\_hangline', 'kitchen\_house', 'bathroom\_constructed',

'swept\_compound', 'dish\_rack\_present', 'composts',

'non\_bio\_waste\_mgt\_present', 'apply\_liquid\_manure',

'organic\_pesticide\_expenditure', 'water\_control\_practise',

'soil\_management', 'food\_banana\_wilt\_diseases',

'postharvest\_food\_storage'

Preprocessed through a scikit-learn pipeline (standardization, one-hot encoding)

***Why Multinomial Logistic Regression?***

Lightweight: Minimal memory footprint-ideal for mobile or offline inference

Fast: Instantaneous predictions, even on low-power devices

Interpretable: Supports explainable outputs via coefficients and SHAP

Robust: Performs well with structured, tabular field data

import joblib

joblib.dump(model, "progress\_status\_model.joblib")

The resulting .joblib file is:

Platform agnostic: Can be loaded in a Python environment on server or mobile backend

Compatible with offline runtimes

***Deployment Options:***

***On-device (Offline):***

Bundle model with the mobile app (e.g., Android) and use embedded Python runtime (Chaquopy)

Enables real-time inference in no-bandwidth areas

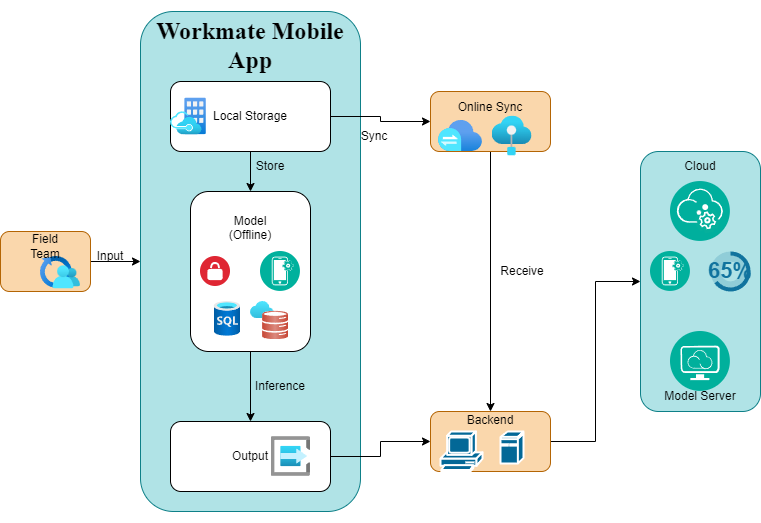
***Backend (Online/Hybrid):***

Serve via lightweight REST API (e.g., FastAPI) for cloud or LAN-hosted inference

Supports model updates and complex logic in connected environments

This packaging strategy ensures flexibility to run the same model in both offline and online modes, making it field-deployable, scalable, and maintainable.

**Integration Architecture Design:** Provide a clear architecture diagram showing how the selected model integrates with the **WorkMate mobile app**, **backend/cloud infrastructure**, and **field devices**. The architecture should include provisions for **offline capability** and data synchronisation when connectivity is restored.



***Draw.io file is also available in the folder.***

**Field Officer User Flow:** Illustrate a step-by-step user interaction flow, from **data input** by the field officer to **real-time prediction** and **suggested action**, including how the app handles **offline fallback** with **local storage and background syncing** once a connection is available

**Step-by-Step User Interaction Flow**

***Log In / Authenticate (Once)***

Field officer logs into the app when online; credentials are cached for offline use.

***Household Visit & Data Entry***

Officer opens a new visit record for a household.

Inputs key features (e.g., income, household size, housing type, etc.) into a structured form within the app.

App validates the input form locally (e.g., required fields, data type checks).

***Real-Time Prediction (On-Device)***

App uses the embedded logistic regression model (packaged via `joblib`, converted to ONNX/TFLite if needed) to run inference directly on the device.

Model outputs predicted `ProgressStatus` class (e.g., Secure, At Risk, Struggling, Extreme).

App displays a visual summary (e.g., colored badge and text) of the result.

***Suggested Action Display***

Based on prediction, the app displays tailored suggestions pulled from a local rules engine or offline knowledge base:

E.g., “Consider enrolling household in income support program.”

Actionable links or checklists may be shown.

***Offline Data Storage***

All interaction logs, form data, prediction, and suggested action are saved to secure local storage (SQLite or Realm DB).

Status flag = `pending\_sync`.

***Background Sync Mechanism***

Periodically checks for connectivity.

Once a stable connection is detected:

Batches are uploaded via secure API to the backend.

`pending\_sync` flag is updated to `synced`.

Server acknowledges with a unique ID and timestamp.

***Supervisor Review***

Synced records become available in the central dashboard for review, aggregation, and further action by program supervisors or data analysts.

**Key Offline Capabilities**

Model Inference: Executed fully on-device, with no dependency on internet.

Data Storage: Robust local persistence of both input and output data.

Sync Resilience: Automatic re-attempts until server acknowledgment is received.

**Code Demonstration:** Show working code that takes household-level input, generates a prediction using the packaged model, and stores the result securely for later upload or real-time use, depending on connectivity, demonstrating how the solution fits into the broader system.