AI DEVELOPMENT WORKFLOW ASSIGNMENT

Part 1: Short Answer Questions (30 points)

1. Problem Definition (6 points)

- Define a hypothetical AI problem (e.g., "Predicting student dropout rates").

  "Predicting Student Dropout Rates in Secondary Schools"

List 3 objectives and 2 stakeholders.

Identify at-risk students early using academic, demographic, and attendance data.

Support schools and educators with timely interventions to reduce dropout rates.

Improve educational outcomes by enabling data-driven policy decisions.

Stakeholders

1. School Administrators and Teachers: Use the AI insights to provide support or counselling.
2. Education Policy Makers: Use the predictions to allocate resources effectively.

Propose 1 Key Performance Indicator (KPI) to measure success.

Recall (Sensitivity): Measures the percentage of actual dropouts correctly identified by the model — important to minimize false negatives and ensure at-risk students are not overlooked.

2. Data Collection & Pre-processing (8 points)

- Identify **2 data sources** for your problem.

  • School Management System (SMS) Records:
  Contains data on student demographics, academic performance (e.g., grades, test scores), attendance, and disciplinary history.
  • Ministry **o**f Education National Database**:**
  Includes aggregated data on school infrastructure, teacher-student ratios, socioeconomic backgrounds, and regional education trends.

Explain **1 potential bias** in the data.

Socioeconomic Bias:
If the dataset contains more students from urban or well-funded schools and fewer from rural or under-resourced areas, the model may learn patterns that favour urban students. As a result, predictions for rural students may be less accurate or unfairly label them as high-risk due to lack of similar training data. This bias can lead to unequal interventions and misinformed policy decisions.

- Outline **3 pre-processing steps** (e.g., handling missing data, normalization).

Pre-processing Steps:

1. **Handling Missing Data:**
    - Fill missing values using appropriate techniques (e.g., mean/median for numerical data, mode for categorical).
    - Alternatively, remove records with excessive missing information if imputation isn't feasible.
2. **Encoding Categorical Variables:**
    - Convert non-numeric features such as gender, region, or school type into numerical form using methods like One-Hot Encoding or Label Encoding.
3. **Feature Scaling (Normalization/Standardization):**
    - Scale numeric features (e.g., grades, attendance rates) to ensure all variables contribute equally to the model, especially for algorithms sensitive to scale like KNN or SVM.

3. Model Development (8 points)

- Choose a model (e.g., Random Forest, Neural Network) and justify your choice.

Random Forest Classifier

JUSTIFICATION

1. **Handles Diverse Data Types:**
   Random Forest works well with both numerical and categorical features, making it suitable for mixed student data (e.g., grades, attendance, demographics).
2. **Robust to Overfitting:**
   By averaging the results of multiple decision trees, Random Forest reduces the risk of overfitting compared to a single tree.
3. **Feature Importance:**
   It provides insights into which features (e.g., absenteeism, exam scores) most influence dropout risk, supporting transparency and decision-making.
4. **Performs Well with Missing Data:**
   Random Forest can still perform reasonably well even if some features have missing values, making it practical in real-world school data scenarios.

Describe how you would split data into training/validation/test sets.

Data Splitting Strategy:

To evaluate and train the model effectively, the dataset should be divided as follows:

1. **Training Set (70%)**
    - Used to train the model and help it learn patterns in the data (e.g., relationships between attendance, grades, and dropout risk).
2. **Validation Set (15%)**
    - Used to fine-tune model parameters (e.g., tree depth in Random Forest) and prevent overfitting by testing model performance during training.
3. **Test Set (15%)**
    - Held back until final evaluation. Used to assess how well the trained model generalizes to unseen data.

- Name **2 hyperparameters** you would tune and why.
- **`n_estimators` (Number of Trees):**
    - **Why:** Determines how many decision trees the forest will use. More trees generally improve performance, but too many can increase training time.
    - **Goal:** Find a balance between accuracy and computational efficiency.
- **`max_depth` (Maximum Depth of Each Tree):**
    - **Why:** Controls how deep each decision tree can grow. Shallow trees might underfit, while very deep trees can overfit the training data.
    - **Goal:** Prevent overfitting while capturing important patterns.

4. Evaluation & Deployment (8 points)

- Select **2 evaluation metrics** and explain their relevance.
- **Recall (Sensitivity):**
    - **Why it's relevant:** Measures the percentage of actual dropouts that the model correctly identifies.
    - **Importance:** In dropout prediction, it's crucial to catch as many at-risk students as possible — missing them could mean no intervention is made.
- **Accuracy:**
    - **Why it's relevant:** Measures the proportion of total correct predictions (both dropouts and non-dropouts).
    - **Importance:** Gives an overall view of the model's performance but must be interpreted carefully in case of class imbalance.

What is **concept drift**? How would you monitor it post-deployment?

Concept drift occurs when the statistical properties of the target variable (e.g., student dropout risk) change over time in unforeseen ways. As a result, a model trained on past data becomes less accurate because the patterns it learned no longer reflect current realities.

Example:
If a new education policy or pandemic changes how students attend school or perform academically, the original model may start making poor predictions.

Monitoring Concept Drift Post-Deployment:

1. **Track Prediction Accuracy Over Time:**
   Continuously evaluate model performance (e.g., accuracy, recall) on recent data. A consistent drop in these metrics may signal drift.
2. **Use Drift Detection Tools:**
   Implement tools like **Population Stability Index (PSI)** or **Kolmogorov-Smirnov (KS) tests** to monitor changes in feature distributions or prediction outputs.
3. **Periodic Model Retraining:**
   Schedule regular retraining using the latest labelled data to keep the model updated with new patterns.

Describe **1 technical challenge** during deployment (e.g., scalability).

Technical Challenge During Deployment:

Scalability

DESCRIPTION

As the number of schools, students, and data records grows, the model must handle increased data volume and prediction requests efficiently. A model that works well during development on small datasets may slow down or fail when deployed across a national education system with thousands of students.

IMPACT

- Delayed predictions could hinder timely interventions.
- Increased server load may cause crashes or slow response times.
- Costs for infrastructure and computing may rise.

POSSIBLE SOLUTIONS

- Use cloud-based infrastructure with auto-scaling (e.g., AWS, Azure).
- Optimize the model with techniques like model compression **or** batch prediction**.**
- Deploy using efficient frameworks **(e.g.,** ONNX**,** TensorFlow Lite**)** for lightweight inference.