Part 1: Short Answer Questions

1. Problem Definition

- **Hypothetical AI Problem**:
  Predicting student dropout rates in online higher education programs.
- **Objectives**:
  1. **Early identification of at-risk students**: Enable proactive interventions (e.g., tutoring, counseling) before dropout occurs.
  2. **Resource optimization**: Allocate academic support resources (e.g., advisors, financial aid) efficiently.
  3. **Curriculum enhancement**: Identify courses/modules with high attrition to redesign content or delivery methods.
- **Stakeholders**:
  1. **Students**: Directly impacted by interventions; seek educational success.
  2. **Academic Administrators**: Responsible for retention rates, funding, and institutional reputation.
- **KPI**:
  Recall (Sensitivity) = *True Positives / (True Positives + False Negatives)*.
  Why: Minimizing false negatives (missed at-risk students) is critical to ensure no student slips through intervention nets.

---

2. Data Collection & Preprocessing (8 points)

- **Data Sources**:
  1. **Learning Management System (LMS) Logs**: Clickstream data (logins, video views, assignment submissions), forum interactions.
  2. **Student Information Systems (SIS)**: Demographics (age, nationality), academic history (prior GPA, course enrollments), financial aid status.
- **Potential Bias**:
  Socioeconomic Bias:
  - Low-income students may:
    - Lack reliable internet, leading to sparse LMS engagement (misclassified as "disengaged").

        - Work part-time, causing irregular activity patterns.
    - **Consequence**: Model may underestimate dropout risk for financially strained students.
- **Preprocessing Steps**:
    1. **Missing Data Imputation**:
        - *Numerical* (e.g., assignment scores): **Median imputation** (robust to outliers).
        - *Categorical* (e.g., major): **"Unknown" category** to preserve data.
    2. **Normalization**:
        - **Min-Max scaling** for engagement metrics (e.g., weekly login count) to [0, 1].
    3. **Temporal Feature Engineering**:
        - Create **rolling-window features** (e.g., "average assignment delay in past 4 weeks").

---

3. Model Development

- **Model Choice**: **XGBoost (Extreme Gradient Boosting)**.
  Justification:
    - Handles **imbalanced data** (dropouts are minority class) via scale_pos_weight.
    - Captures **nonlinear interactions** (e.g., declining forum activity + low quiz scores).
    - Provides **feature importance** for interpretability (e.g., identifies "assignment delay" as top predictor).
- **Data Splitting**:
    - **Stratified temporal split**:
        - Training (2018-2020): **60%** (preserves seasonality).
        - Validation (2021): **20%** (hyperparameter tuning).
        - Test (2022): **20%** (unseen data for final evaluation).
    - *Why temporal?* Prevents data leakage (future data influences past training).

- **Hyperparameters to Tune**:
    1. max_depth: Controls tree complexity. *Why*: Prevents overfitting noisy patterns (e.g., COVID-era anomalies).
    2. subsample: Fraction of data sampled per tree. *Why*: Improves generalization via bagging.

---

4. Evaluation & Deployment

- **Evaluation Metrics**:
    1. **F1-Score**:
        - *Relevance*: Balances **precision** (avoid overwhelming advisors with false alarms) and **recall** (capture true at-risk students).
    2. **AUC-ROC**:
        - *Relevance*: Evaluates ranking capability across thresholds (e.g., prioritizes top 10% highest-risk students).
- **Concept Drift**:
    - **Definition**: Shift in data distribution post-deployment (e.g., new online learning tools alter engagement patterns).
    - **Monitoring**:
        - Track **PSI (Population Stability Index)** on feature distributions monthly.
        - Alert threshold: **PSI > 0.1** indicates significant drift.
        - Retrain model if **F1-score drops by >5%** on new data.
- **Technical Challenge**:
  Scalability during enrollment peaks:
    - *Challenge*: 10x prediction requests during semester start.
    - *Solution*: **Containerized microservices** (Docker + Kubernetes) with auto-scaling.

Part 2: Case Study Application

Problem Scope

- **Problem**: Predict 30-day hospital readmission risk post-discharge.

- **Objectives**:
    1. Reduce avoidable readmissions by **20%** in 1 year.
    2. Prioritize high-risk patients for **transitional care programs** (e.g., home visits).
    3. Lower healthcare costs (average readmission costs $15,000).
- **Stakeholders**: Patients, clinicians, hospital administrators, insurers.

Data Strategy

- **Data Sources**:
    1. **Electronic Health Records (EHRs)**: Lab results, medications, diagnoses (ICD codes), discharge notes.
    2. **Socioeconomic Data**: ZIP code-linked indices (e.g., Area Deprivation Index).
- **Ethical Concerns**:
    1. **Patient Privacy**:
        - *Risk*: Unauthorized access to PHI (Protected Health Information).
        - *Mitigation*: **De-identification** (remove names, MRNs) pre-processing.
    2. **Algorithmic Bias**:
        - *Risk*: Underestimating risk for non-English speakers (less documented social needs).
        - *Mitigation*: **Disparity testing** across language groups.
- **Preprocessing Pipeline**:
    1. **Cleaning**:
        - Remove duplicate admissions (same patient within 30 days).
    2. **Imputation**:
        - **Multivariate imputation** (MICE) for missing vitals (e.g., blood pressure).
    3. **Feature Engineering**:
        - **Clinical**:
            - *Comorbidity score* (e.g., Elixhauser Index) from ICD codes.
            - *Polypharmacy flag* (>10 medications at discharge).

        - **Temporal**:
            - *Days since last admission*.
        - **Social Determinants**:
            - *Transportation access* (via ZIP code public transit scores).

Model Development

- **Model Choice**: **Logistic Regression**.
  Justification:
    - **Interpretability**: Coefficients show clinical drivers (e.g., +0.5 weight for heart failure).
    - **Regulatory compliance**: Meets "right to explanation" under HIPAA.
    - **Efficiency**: Fast inference (<100ms/patient) on EHR systems.
- **Confusion Matrix & Metrics**:
    - **Hypothetical Data**:
        - TP = 120, FP = 30, FN = 50, TN = 800
    - **Precision** = TP / (TP + FP) = 120 / 150 = **0.80**
    - **Recall** = TP / (TP + FN) = 120 / 170 = **0.71**

Deployment

- **Integration Steps**:
    1. **API Endpoint**: Deploy model as **REST API** (FastAPI) in hospital's Azure cloud.
    2. **EHR Trigger**: Automatically run predictions at discharge using **FHIR standards**.
    3. **Clinical Alerts**: Flag high-risk patients in clinician dashboards (Epic/Cerner).
- **Regulatory Compliance (HIPAA)**:
    1. **Data Governance**:
        - Store data in **encrypted** databases (AES-256).
        - **Access logs** for audit trails.

2. **Model Certification**:
    - **Validation** by hospital's ethics board.
    - **Documentation** of training data provenance.

Optimization

- **Overfitting Mitigation**: **L1 Regularization (LASSO)**.
    - *Why*: Penalizes irrelevant features (e.g., rare diagnostic codes), shrinking coefficients to zero.

Part 3: Critical Thinking

Ethics & Bias

- **Bias Impact**:
    - If training data underrepresents **homeless patients** (less documentation, frequent ER visits):
        - Model may **underpredict** their readmission risk → denied post-discharge shelter support.
        - **Consequence**: Higher readmissions, perpetuating health inequities.
- **Mitigation Strategy**: **Reweighting**.
    - Assign **higher sample weights** to underrepresented groups during training.
    - *Example*: Triple weight for patients with "homeless" in discharge notes.

Trade-offs

- **Interpretability vs. Accuracy**:

Model Type	**Accuracy**	**Interpretability**	**Healthcare Impact**
Logistic Regression	Moderate	High (coefficients)	Trust from clinicians; auditable.
Deep Learning	High	Low ("black box")	Mistrust; rejected in practice.

- **Resolution**: Use **SHAP values** with XGBoost for accuracy + interpretability.

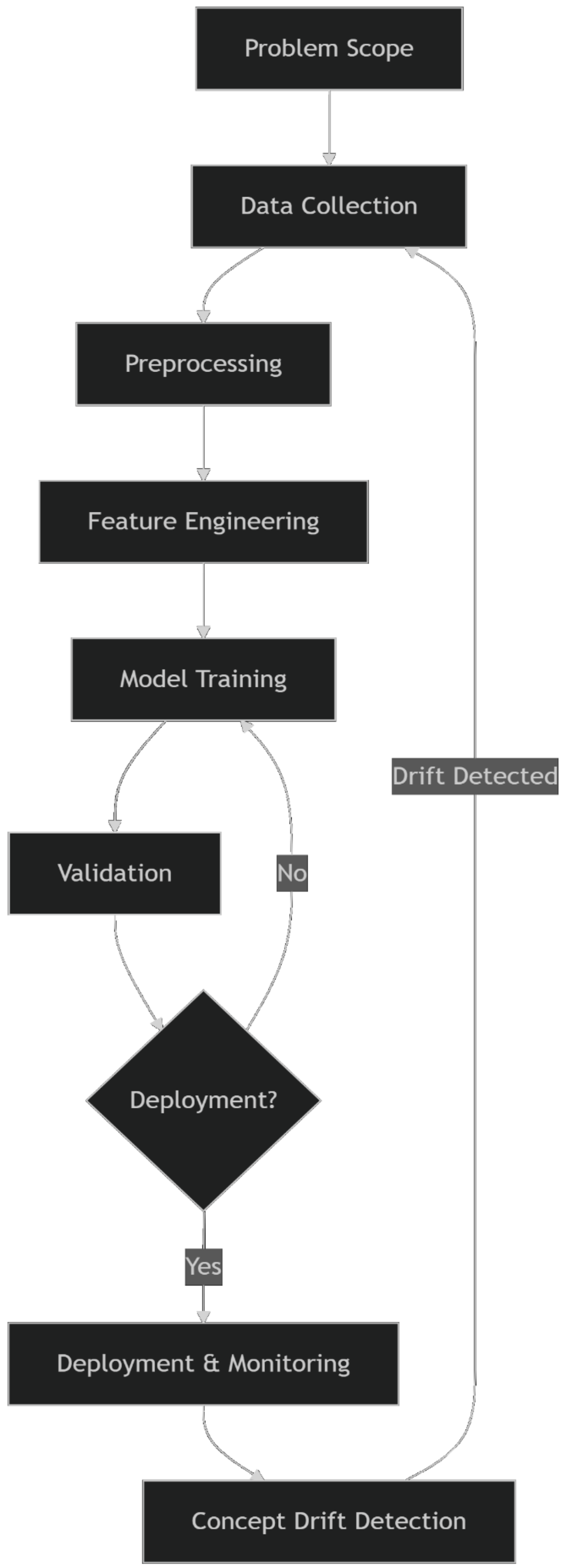
- **Limited Computational Resources**:
  - **Impact**: Forces simpler models (logistic regression > neural networks).
  - **Solution**:
    - **Feature selection** (e.g., Recursive Feature Elimination).
    - **Edge deployment**: Run inference on local servers (not cloud).

Part 4: Reflection & Workflow Diagram

Reflection

- **Most Challenging**: **Bias mitigation**.
  - *Why*: Requires domain expertise to identify subtle biases (e.g., coding disparities for chronic pain in Black patients).
- **Improvement with More Resources**:
  1. **Longitudinal validation**: Track model impact on readmission disparities over 2 years.
  2. **Causal analysis**: Use tools like **DoWhy** to distinguish correlation from causation (e.g., homelessness → readmission).

Workflow Diagram


Problem Scope
Data Collection
Preprocessing
Feature Engineering
Model Training
Validation
No
Drift Detected
Deployment?
Yes
Deployment & Monitoring
Concept Drift Detection

Stages:

1. **Problem Scope**: Define objectives, constraints, stakeholders.
2. **Data Collection**: EHRs, socioeconomic indices, claims data.
3. **Preprocessing**: Cleaning, imputation, normalization.
4. **Feature Engineering**: Comorbidity scores, temporal features.
5. **Model Training**: Algorithm selection, hyperparameter tuning.
6. **Validation**: Metrics (AUC, F1), fairness audits.
7. **Deployment & Monitoring**: API integration, real-time alerts.
8. **Concept Drift Detection**: PSI, performance tracking.