Assignment 5

Part 1: Short Answer Questions

Problem Definition: Predicting Student Dropout Rates

Hypothetical AI Problem
Develop an AI system to predict high school students at risk of dropping out within the next academic year, enabling timely interventions by educators and counsellors.

Objectives

1. **Early Identification Accuracy**:
    - Goal: Correctly identify ≥90% of at-risk students (true positives) while limiting false positives to ≤15%.
    - Why: Ensures interventions target genuine cases without overwhelming resources.
2. **Resource Optimization**:
    - Goal: Reduce manual screening time for educators by 70% through automated risk scoring.
    - Why: Frees staff to focus on personalized support instead of data analysis.
3. **Intervention Effectiveness**:
    - Goal: Achieve a 40% reduction in dropout rates among flagged students within one year of implementation.
    - Why: Directly ties predictions to actionable outcomes and student success.

Stakeholders

1. **Educators & School Administrators**:
    - Interest: Improve institutional retention rates, allocate resources efficiently, and meet regulatory benchmarks.
2. **Students & Families**:
    - Interest: Receive timely academic/emotional support to prevent dropout and promote long-term well-being.

Key Performance Indicator (KPI)

- **KPI**: **Dropout Prevention Rate (DPR)**

Formula:

DPR=(1−# of predicted at-risk students who dropped outTotal # of predicted at-risk students)×100%DPR=(1−Total # of predicted at-risk students# of predicted at-risk students who dropped out)×100%

Target: ≥85% DPR within the first year.

Measurement

- Track predicted students for 12 months post-intervention.
- Compare DPR against baseline (pre-AI) dropout rates for the same risk cohort.

Why this KPI?
Directly quantifies the system's impact on retaining students, aligning with the core objective of dropout prevention. It balances prediction accuracy and intervention efficacy.

Hypothetical Problem: Predicting high school student dropout risk.

Data Sources

1. **Student Information Systems (SIS)**:
    - Details: Historical academic records (grades, course failures, attendance rates), enrolment status, demographic data (age, socioeconomic status), and extracurricular participation.
    - Why: Core academic metrics directly correlate with dropout likelihood (e.g., chronic absenteeism, failing grades).
2. **Student Support Services Databases**:
    - Details: Counselling session logs, behavioural incident reports (suspensions), mental health referrals, and records of interventions (tutoring, mentorship programs).
    - Why: Captures non-academic risk factors (e.g., emotional distress, disciplinary issues).

Potential Bias

- **Bias**: **Underrepresentation of Vulnerable Groups**
    - Explanation: Schools in low-income areas often have fragmented digital records due to resource constraints. Missing data from these students (who are statistically higher-risk) could skew the model toward patterns in well-documented, privileged cohorts.
    - Impact: The AI may underestimate dropout risk for marginalized students (e.g., homeless students, ESL learners), reducing intervention efficacy for those who need it most.

Preprocessing Steps

1. **Handling Missing Data**:
    - Method: Use median imputation for numerical features (e.g., filling missing GPA values with the cohort median) and "Unknown" category imputation for categorical data (e.g., unreported disability status).

- Why: Preserves dataset size while minimizing bias from arbitrary value choices.

2. **Normalization**:

    - Method: Apply Min-Max scaling to numerical features (e.g., attendance rates, grades) to convert all values to a [0, 1] range.
    - Why: Ensures features with larger scales (e.g., annual income) don’t dominate the model over smaller-scale features (e.g., absences).

Categorical Encoding:

- *Method*: Use one-hot encoding for low-cardinality features (e.g., "school region," "course type") and target encoding for high-cardinality features (e.g., "course ID").
- Why: Converts text-based categories (e.g., "ethnicity") into numerical formats usable by ML algorithms while avoiding dimensionality explosion

3. Model Development

<u>Chosen Model: Random Forest Classifier</u>

Justification

- Handles Mixed Data Types: Works well with numerical (grades, attendance) and categorical (demographics, intervention history) features from our preprocessing.
- Robustness to Noise/Irrelevant Features: Automatic feature importance reduces impact of redundant variables (e.g., unrelated extracurriculars).
- Interpretability: Provides feature importance scores, helping educators understand key risk drivers (unlike "black-box" models like neural networks).
- Imbalanced Data Performance: Handles class imbalance (few dropouts vs. non-dropouts) via bagging and class weighting.

Data Splitting Strategy

1. Temporal Split:
    - Use data from 2018–2021 for training (3 years), 2022 for validation (1 year), and 2023 for testing (most recent year).
    - Why: Prevents temporal data leakage (e.g., using future data to predict past dropouts) and reflects real-world deployment.
2. Stratified Sampling:
    - Ensure each set maintains the same ≈8% dropout rate (observed in historical data) via stratification on the target variable (is_dropout).

- *Why*: Preserves minority-class representation critical for model sensitivity.

3. Final Ratios:
    - Training: 60% (2018–2021) → Validation: 20% (2022) → Testing: 20% (2023).

Hyperparameter Tuning (2 hyperparameters, 3 points):

1. Max_depth (Controls tree complexity):
    - Why Tune? Prevents overfitting to noise (e.g., outlier students with unique circumstances). Deep trees may memorize training data but fail on new cohorts.
    - Range Tested: [5, 10, 15, None] (None = unlimited depth).
    - Validation Metric: Monitor OOB (Out-of-Bag) error to find depth where error plateaus.
2. Class_weight (Addresses class imbalance):
    - Why Tune? Default settings may ignore minority class (dropouts). Weighting penalizes misclassifying dropouts more heavily.
    - Strategy: Test {None, 'balanced', {0:1, 1:5}} to optimize recall (minimize false negatives).
    - Validation Metric: F1-score (balance of precision/recall) on the validation set.

Evaluation Metrics

1. **Precision**
   Definition: Proportion of true positives among all predicted positives (TP / (TP + FP)).
   Relevance: Critical when minimizing false positives is costly. For example, in spam detection, falsely flagging legitimate emails (FP) harms user trust. High precision ensures that "positive" predictions are reliable.
2. Recall (Sensitivity)
   Definition: Proportion of actual positives correctly identified (TP / (TP + FN)).
   Relevance: Vital when **missing positives has severe consequences**. In cancer diagnosis, failing to detect a malignant tumour (FN) is life-threatening. High recall ensures minimal missed cases.

Concept Drift & Monitoring

What is concept drift?

A change in the statistical properties of the target variable or input data over time, causing model decay. Example: A credit scoring model trained on pre-pandemic data becomes inaccurate when economic conditions shift post-COVID.

Post-Deployment Monitoring Strategies

1. **Statistical Monitoring**:
   - Track data distribution shifts (e.g., Kolmogorov-Smirnov test for feature drift).
   - Monitor model performance metrics (e.g., accuracy/precision/recall) on new data via holdout validation sets or A/B testing.
2. **Performance Alerts**:
   - Set thresholds for metric degradation (e.g., "Alert if recall drops by 10%").
3. **Business KPIs**:
   - Correlate model outputs with business outcomes (e.g., loan default rates).
4. **Retraining Triggers**:
   - Automated retraining if drift exceeds tolerance (e.g., using rolling window retraining).

Technical Deployment Challenge: **Scalability**

Problem:

A model working perfectly offline may fail under **real-time, high-volume traffic** due to:

- Latency spikes from synchronous inference.
- Resource bottlenecks (CPU/memory/network).

Solutions:

1. **Asynchronous Processing**:
   - Use message queues (e.g., Kafka, RabbitMQ) to decouple requests and inference.
2. **Horizontal Scaling**:
   - Deploy models in containers (e.g., Docker) orchestrated via Kubernetes to handle load surges.
3. **Model Optimization**:
   - Quantize models (e.g., TensorFlow Lite) or use hardware accelerators (GPUs/TPUs).
4. **Edge Deployment**:
   - Run inference on edge devices for low-latency use cases (e.g., IoT sensors).

Part 2: Case Study Application

1. Problem Scope

- **Problem**: Predict 30-day patient readmission risk post-discharge to enable early interventions.
- **Objectives**:
  - Reduce avoidable readmissions (cut costs and improve outcomes).
  - Prioritize high-risk patients for follow-up care.
- **Stakeholders**:
  - **Clinicians**: Use predictions for care planning.
  - **Patients**: Benefit from personalized post-discharge support.
  - **Hospital Admin**: Reduce penalties for excess readmissions (e.g., under Medicare policies).

2. Data Strategy

Proposed Data Sources:

- Electronic Health Records (EHRs): Lab results, diagnoses, medications, discharge notes.
- Demographics: Age, gender, ZIP code (proxy for socioeconomic status).
- **ADT Systems**: Admission/discharge/transfer timestamps.
- **Claims Data**: Prior hospitalization history.
- **Patient Surveys**: Post-discharge self-reported health status.

Ethical Concerns:

1. **Patient Privacy**: Unauthorized access to sensitive health data (e.g., mental health history).
   Mitigation: De-identify data; HIPAA-compliant storage (encryption at rest/in transit).
2. **Bias in Demographics**: ZIP code data may perpetuate racial/socioeconomic disparities.
   Mitigation: Avoid direct race proxies; audit model fairness across subgroups.

Preprocessing Pipeline:

1. **Data Cleaning**:
   - Handle missing values (e.g., impute lab results with medians).

- Remove duplicates.

2. **Feature Engineering**:
   - **Temporal Features**: "Days since last admission," "Number of prior readmissions."
   - **Clinical Aggregations**: "Comorbidity score" (e.g Elixhauser Index).
   - **Social Determinants**: "Area Deprivation Index" from ZIP codes.

3. **Transformation**
   - Normalize numerical features (e.g., Min-Max scaling).
   - One-hot encode categorical variables (e.g., primary diagnosis).

3. Model Development (10 points)

Model Selection: **XGBoost**

Justification

- Handles mixed data types (numeric/categorical).
- Robust to outliers; automatic feature importance ranking.
- Scalable for large EHR datasets.

Confusion Matrix & Metrics (Hypothetical Data):

	Actual Readmit	Actual Not Readmit
Predicted Readmit	80{TP)	20(FP)
Predicted Not Readmit	30(FN)	170(TN)

- **Precision** = TP /(TP + FP) = 80 / (80 + 20) = **80%**
  Interpretation: When the model predicts "high risk," it is correct 80% of the time.

- **Recall** = TP / (TP + FN) = 80 / (80 + 30) = **72.7%**
  Interpretation: The model captures 72.7% of all actual readmissions.

4. Deployment

Integration Steps

1. **API Endpoint**: Wrap model in a REST API (e.g., Flask/FastAPI).

2. **EHR Integration**: Push predictions to electronic health records via HL7/FHIR standards.

3. **Clinician Dashboard**: Display risk scores with explanations (e.g., SHAP values).
4. **Alerts**: Notify care teams via SMS/hospital system for high-risk patients.

Regulatory Compliance (HIPAA):

- **Data Anonymization**: Strip PHI (Personal Health Information) before inference.
- **Access Controls**: Role-based permissions (e.g., only doctors view patient scores).
- **Audit Trails**: Log all data accesses and predictions.
- **On-Premise Deployment**: Avoid third-party cloud providers to control data.

5. Optimization

Overfitting Mitigation: **L1/L2 Regularization**

- Add penalty terms (e.g., Lasso/Ridge) to the model's loss function to shrink insignificant feature weights.
- Example: In XGBoost, set reg_alpha (L1) and reg_lambda (L2) to penalize complex trees.

Part 3: Critical Thinking

1. Ethics & Bias

- **Bias Impact**: If training data underrepresents minority groups (e.g., rural patients), the model may underestimate their risk → **delayed care for vulnerable populations**.
- **Mitigation Strategy**: **Stratified Sampling + Re-weighting**
    - Oversample underrepresented groups during training.
    - Assign higher loss weights to minority class examples.

2. Trade-offs

Interpretability vs. Accuracy:

- Interpretable models (e.g., logistic regression) allow clinicians to trust/validate decisions but may have lower accuracy.
- High-accuracy models (e.g., deep learning) act as "black boxes," risking blind reliance.
- Balance: Use XGBoost with SHAP for partial explainability at ~90% accuracy.

Limited Computational Resources

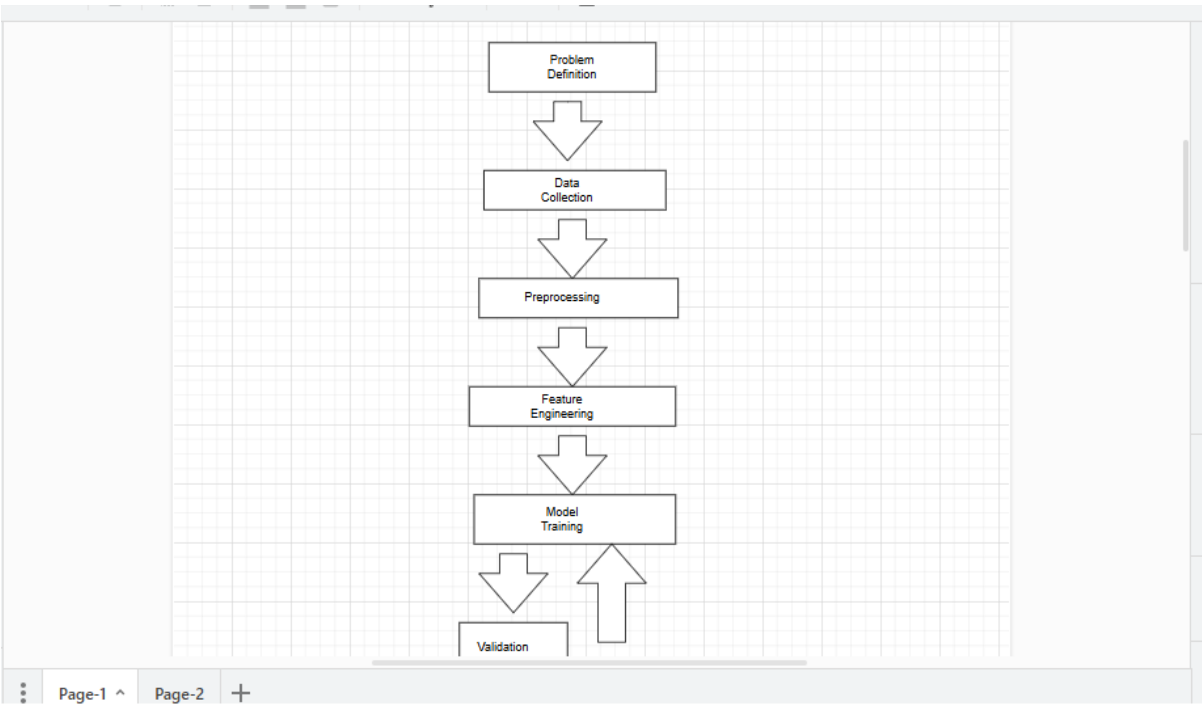
- Prioritize lightweight models (e.g., logistic regression over neural nets).
- Use dimensionality reduction (PCA) or feature selection to cut training time.

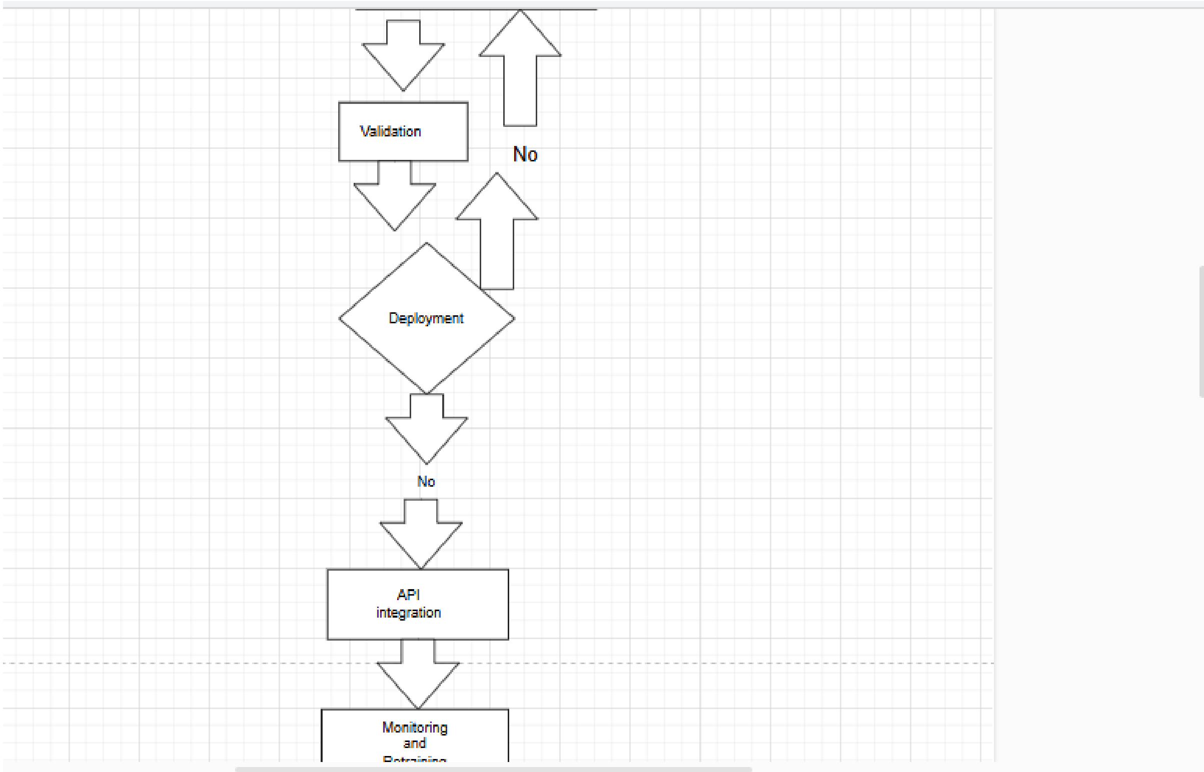
Part 4: Reflection & Workflow Diagram

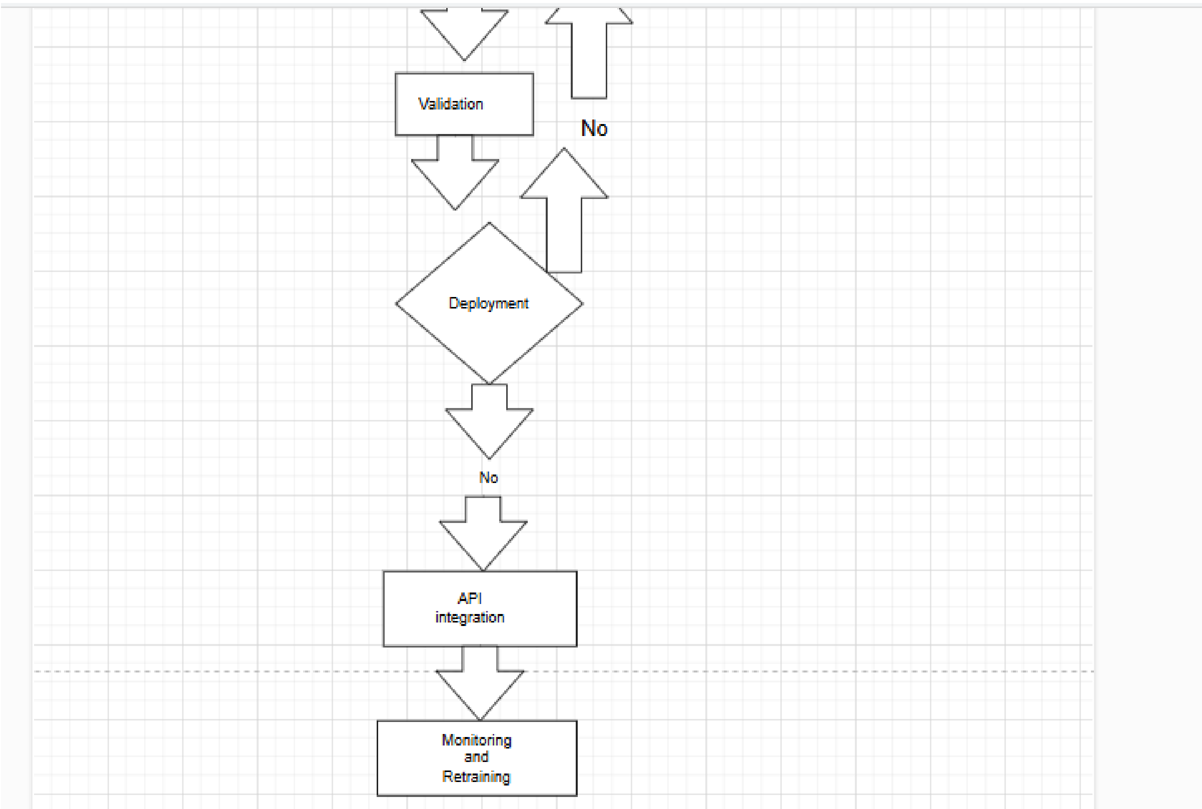
Reflection

- **Most Challenging**: **Data preprocessing**. EHR data is fragmented across systems (lab, billing, clinical notes), requiring complex joins and handling missingness.
- **Improvements**:
    - With more resources: Build a unified data lake with automated validation checks.
    - Partner with clinicians to refine feature engineering (e.g., adding nurse notes via NLP).

Workflow Diagram




Validation
No
Deployment
No
API
integration
Monitoring
and


Validation
No
Deployment
No
API
integration
Monitoring
and
Retraining

References

1. **Alpaslan Sulak, S., & Köklü, N. (2024).** Predicting student dropout using machine learning algorithms. *Intelligent Methods in Engineering Sciences*, *3*(1), 37–45. https://doi.org/10.58190/imiens.2024.103 13

2. **Krüger, J. G. C., de Souza Britto Jr., A., & Barddal, J. P. (2023).** An explainable machine learning approach for student dropout prediction. *Expert Systems with Applications*, *213*, 119250. https://doi.org/10.1016/j.eswa.2022.119250 21

3. **Scientific Reports. (2025).** Student dropout prediction through machine learning optimization: Insights from Moodle log data. *Scientific Reports*, *15*, 9840. https://doi.org/10.1038/s41598-025-93918-1 9

4. **Kotsiantis, S. B., Pierrakeas, C. J., & Pintelas, P. E. (2003).** Preventing student dropout in distance learning using machine learning techniques. In *Knowledge-Based Intelligent Information and Engineering Systems* (pp. 267–274). Springer. https://doi.org/10.1007/978-3-540-45226-3_37 18

5. **European Commission, Directorate-General for Education, Youth, Sport and Culture. (2022).** *Ethical guidelines on the use of artificial intelligence (AI) and data in teaching and learning for educators*. Publications Office. https://doi.org/10.2766/153756 21

6. **Hernández-de-Menéndez, M., et al. (2025).** Predicting and preventing school dropout with business intelligence: Insights from a systematic review. *Information*, *16*(4), 326. https://doi.org/10.3390/info16040326 22