Week 5 Assignment

Part 1: Short Answer Questions

1. Problem Definition

The core AI problem is Optimising Urban Traffic Flow in a Megacity. This requires developing an AI system to dynamically adjust city-wide traffic signal timings in real-time to minimize overall travel time and reduce congestion, particularly during peak hours and unpredictable events.

The system focuses on three main objectives: minimizing average **Travel Time** (a measurable reduction over static systems), **Reducing Idle Emissions** (by decreasing standstill time to improve air quality), and prioritizing **Emergency Response** by quickly clearing paths for critical vehicles.

Two primary stakeholders are crucial: The **City Transportation Department** needs a reliable, economically sound, and compliant system. **Commuting Citizens and Commercial Drivers** benefit from reduced stress, predictable travel times, and improved logistical efficiency.

Success is measured by the core Key Performance Indicator (KPI): **Reduction in Average Vehicle Travel Time (RAVTT)**. This metric tracks the percentage decrease in mean peak-hour travel time versus the established baseline, often targeting a specific goal like a 15% RAVTT across the ten most congested routes within six months.

2. Data Collection & Preprocessing

- **Data Sources**

Two critical data sources are required for training and operating the Reinforcement Learning model. **Sensor Data** (from loops/cameras) provides real-time inputs (volume, speed, queue length) to define the AI's State. **Mobile and GPS Data** (from navigation apps/cellular providers) offers a macro view of the network by tracking actual end-to-end travel times, serving as the essential ground truth for the KPI (Reduction in Average Travel Time).

- **Potential Data Bias**

A significant risk is **Geographical/Socioeconomic Bias** within the Mobile and GPS Data. Reliance on specific apps, often used by wealthier or commercial users, can skew the data. This bias may cause

the AI to disproportionately optimise sampled routes (e.g., central business districts), potentially worsening congestion in underrepresented, peripheral neighborhoods.

- **Preprocessing Steps**

Three key steps prepare the sensor and GPS data for the Reinforcement Learning model. First, **Handling Missing and Erroneous Data** addresses frequent sensor failures; missing values require imputation via time-series or spatial interpolation. Second, **Temporal Normalisation and Aggregation** structures high-frequency data into meaningful 30-second State intervals, while temporal features (hour, day) are cyclically normalised to capture periodicity. Third, **Feature Scaling** standardises numerical features (e.g., queue lengths, speeds) using methods like Min-Max or Z-score, ensuring data falls within a consistent range ($$[0, 1]$) to prevent disproportionate influence on the learning algorithm.

3. Model Development

- **Data Splitting Strategy**

The RL problem requires **Time-Series Splitting** of historical traffic data (sensor/GPS) in chronological order to preserve sequential dependencies. The dataset is divided into three blocks: **Training (70%)** for initial DQN agent training; **Validation (15%)** for hyperparameter tuning and preventing overfitting; and **Test (15%)** for final, unbiased evaluation against the KPI (Reduction in Average Vehicle Travel Time).

- **Key Hyperparameters for Tuning**

Two crucial DQN hyperparameters are the **Learning Rate** (α) and the **Discount Factor** (γ). α (tuned between 10^{-3} and 10^{-5}) sets the step size for weight updates, critical for stable and efficient convergence. The ‘Discount Factor’ \(γ) determines the balance between future and immediate rewards. Since traffic requires balancing local relief (immediate) with global congestion avoidance (long-term), γ is critical. A high value (e.g., 0.99) prioritizes long-term outcomes, while a low value (e.g., 0.6) favors short-term relief.

4. Evaluation & Deployment

- **Evaluation Metrics and Relevance**

The first metric is the **Average Vehicle Delay (AVD)**, measured in seconds per vehicle. AVD quantifies the total time vehicles spend stopped or moving slowly due to traffic control, making it a granular measure of congestion. A percentage reduction in AVD directly validates the objective of

Reducing Idle Emissions, leading to better air quality. The second metric is the **Emergency Vehicle Response Time (EVRT)**. This measures the time an emergency vehicle takes to travel from origin to destination under AI signal control. A reduced EVRT validates the objective of Prioritising Emergency Response, demonstrating improved public safety efficiency compared to the baseline.

- **Concept Drift and Monitoring**

Concept Drift occurs when the statistical properties of the traffic problem (the relationship between traffic conditions and optimal signal timing) permanently change post-deployment. This can be caused by fundamental city dynamics shifting, such as major construction or a permanent change in commuter behavior. Such events make the original DQN policy outdated and suboptimal. Monitoring for drift involves continuously tracking the model's prediction error and watching for a persistent, statistically significant increase in average queue length or a deterioration of the RAVTT KPI on specific routes over several weeks, which signals the need for model retraining.

- **Technical Deployment Challenge**

The most significant technical challenge is **Low-Latency Scalability and Distributed Control**. The AI cannot be housed on a single server, as it must manage thousands of intersections (Agents) across a megacity. Each Agent must process real-time data, compute the optimal action, and execute the signal change within a strict, low-latency window (milliseconds) to ensure effective reaction. This necessitates deploying a Distributed Control System, often involving Edge Computing (computation at the intersection controller) or regional servers, backed by a robust, high-speed, and fault-tolerant communication network for agent coordination.

Part 2: Case Study Application

1. Problem Scope

Component	**Description**
Problem Definition	To accurately predict which discharged patients are at high risk of readmission to the hospital within 30 days of their discharge date.
Objective	Reduce preventable 30-day readmissions by

	identifying high-risk patients, allowing for targeted post-discharge interventions and resource allocation. The ultimate goal is to improve patient care quality and reduce healthcare costs/penalties.
Stakeholders	Key stakeholders are: Patients (benefit from improved care and reduced readmissions); Hospital Administration (focus on quality metrics and cost reduction); Clinical Staff (use predictions to guide care interventions); and the Data Science Team (responsible for building and maintaining the model).

2. Data Strategy

Data Sources:

- Electronic Health Records (EHRs): Primary source. Includes diagnosis codes (ICD-10), procedure codes, admission and discharge summaries, lab results (e.g., haemoglobin A1c, creatinine), and vital signs.
- Patient Demographics: Age, gender, zip code (proxy for socioeconomic status/access to care), insurance type, and primary language.
- Medication Records (e.g., e-Prescribing): Discharge medications, medication adherence history, and known drug allergies.
- Past Utilisation Data: Number of prior hospitalisations, Emergency Department (ED) visits, and chronic disease management history (e.g., previous cardiology appointments).

Ethical Concerns:

- Patient Privacy and HIPAA Compliance: The use of detailed patient health information (PHI) for model training and prediction carries a high risk of breach. The ethical concern is the potential for unauthorised access or re-identification of sensitive health data, violating the fundamental right to privacy.
- Algorithmic Bias and Health Equity: If the training data disproportionately represent certain demographic groups or are based on historically biased medical practices, the model may systematically underestimate the risk for underserved populations or those of a specific

race/socioeconomic status. This could lead to a lack of necessary post-discharge resources for those patients, worsening health disparities.

Preprocessing Pipeline:

Step	Description	Feature Engineering (FE) Example
1. Data Cleaning	Handle missing values. Correct data entry errors and standardise units.	N/A
2. Data Transformation	Convert categorical variables into a numerical format. One-hot encoding for nominal categories. Label Encoding for ordinal categories (e.g., severity scale).	N/A
3. Feature Engineering	Create new, predictive features from existing raw data.	CCI Score: Summarizes illness burden based on chronic conditions (ICD-10 codes). LOS: The number of days between patient admission and discharge.
4. Feature Scaling	Standardise or normalise continuous features to ensure all features contribute equally to the model training process.	Standardisation ($\frac{x - \mu}{\sigma}$): Applied to features like age and lab values.
5. Data Splitting	Divide the processed dataset into Training, Validation, and Test sets (e.g., 70/15/15 split).	N/A

3. Model Development

Selected Model: XGBoost (Extreme Gradient Boosting) or LightGBM.

Justification:

- High Predictive Power: Tree-based ensemble methods, especially boosting algorithms, typically achieve state-of-the-art performance on tabular healthcare data, often outperforming traditional methods like Logistic Regression.
- Handles Non-Linearity: Readmission risk is complex, and the relationship between features (e.g., age, LOS, lab values) and the target variable is highly non-linear, which XGBoost handles well.
- Feature Importance: XGBoost provides a clear measure of feature importance, which is crucial for clinical interpretability and validating that the model is making sense based on medical knowledge.

Scalability: Efficiently handles large datasets common in EHR systems.

Confusion Matrix and Metrics (Hypothetical Data)

1. **Precision (Positive Predictive Value):** The proportion of patients predicted as high-risk who actually readmit. High precision reduces unnecessary intervention costs (False Positives).

2. **Recall (Sensitivity)**: The proportion of patients who were actually readmitted that the model correctly identified as high-risk. High recall ensures that very few genuinely high-risk patients are missed (False Negatives). In healthcare, recall is often prioritised to ensure the most vulnerable patients receive the necessary care.

4. Deployment

Integration Steps

1. **API Development:** Wrap the trained model (e.g., a pickled XGBoost object) in a **RESTful API** (using frameworks like Flask or FastAPI) to allow for simple, standardised requests and responses. The API should be hosted on a secure, internal hospital server.
2. **EHR System Hook:** Create a trigger within the EHR system (e.g., at the time a physician begins the discharge order). This trigger automatically sends the patient's necessary data features to the **Prediction API**.
3. **Real-Time Scoring:** The API instantly processes the data, runs the prediction, and returns the **readmission probability score** (e.g., 0.85) to the EHR system.

4. **Clinical Interface Display:** The score is displayed prominently in the EHR's discharge summary/dashboard. If the score exceeds a predefined threshold (e.g., >0.70), an **alert** is triggered to notify the care coordinator to initiate a high-intensity intervention plan.

Regulatory Compliance (HIPAA)

The primary method for ensuring compliance with HIPAA's **Privacy Rule** and **Security Rule** is:

5. **De-identification/Minimum Necessary Principle:** In the development and testing phases, PHI must be **de-identified** (e.g., removal or hashing of direct identifiers like names, SSN, etc.). For the live model, only the **minimum necessary** data fields required to generate the prediction should be transmitted from the EHR to the prediction service.
6. **Encryption and Access Controls (Security Rule):**
    a. All data transfer between the EHR and the AI service must be secured using **end-to-end encryption** (e.g., TLS/SSL).
    b. The model's host server must be protected with strict **role-based access controls** (RBAC) and monitored with detailed audit logs to track who accesses the system and when.

7. Optimization

Proposed Method: Regularisation (for XGBoost)

Description: Regularisation adds a penalty term to the model's loss function based on the complexity of the model (e.g., the number of leaves or the magnitude of weights).

- **L1 (Lasso) and L2 (Ridge) Regularisation:** XGBoost utilises L1 (λ) and L2 (α) regularisation terms in its objective function to **penalise large weights** (coefficients for the leaf nodes in the trees).
- **Impact:** By penalising complexity, the model is discouraged from learning noise and idiosyncrasies in the training data, resulting in smoother decision boundaries and better generalisation performance on unseen data.
- **Implementation:** Tune the λ and α **hyperparameters** during model training (e.g., using cross-validation) until the model's performance on the validation set stabilises or improves, even if the training set performance slightly drops.

Part 3: Critical Thinking

1. Ethics & Bias

Biased data (e.g., incomplete socio-economic information or under-reported student populations) severely compromises student outcomes by causing flawed systemic decision-making. When student needs are misrepresented (e.g., poverty levels are not fully captured), resources, funding, and support services are allocated unfairly, inadequately serving those who need them most. This deficiency widens the achievement gap. Furthermore, biased data used in predictive algorithms can amplify existing inequities, potentially leading to unfair academic tracking, disproportionate disciplinary actions, or lower expectations for underrepresented groups.

A key method to reduce this bias is **Data Diversification and Granularity**. Instead of relying solely on narrow metrics like test scores, systems must actively seek a wider variety of information to form holistic student profiles. This includes collecting complete contextual data (access to housing, health, technology) alongside traditional academic records. Crucially, data must be analyzed with high granularity—broken down into specific subgroups (e.g., specific race/ethnicity, language learner status, precise socio-economic tiers)—to pinpoint disparities and enable precise, equitable, and targeted interventions.

2. Trade-offs

Interpretability is prioritized over marginal accuracy gains in educational modeling because decisions must be justifiable, transparent, and actionable due to ethical implications. Educators require clear articulation of driving factors (e.g., low attendance) to design effective support programs; without this, even an accurate "black box" model lacks necessary insights. Therefore, simpler techniques like Logistic Regression are preferred as their structure directly maps features to outcomes, simplifying model auditing and intervention design.

Limited computational resources common in many school districts severely constrain viable machine learning approaches by restricting processing power and training time. Training complex models like large deep learning architectures becomes computationally prohibitive, resulting in slow, unresponsive real-time predictions. The recommended strategy is twofold: first, prioritize simpler ML models (e.g., Linear Models, Decision Trees) for their lower overhead and sufficient accuracy on existing hardware. Second, utilize Batch Prediction for non-critical tasks by scheduling large datasets to run overnight, maximizing idle compute power and reducing continuous operational costs.

Part 4: Reflection & Workflow Diagram

1. Reflection