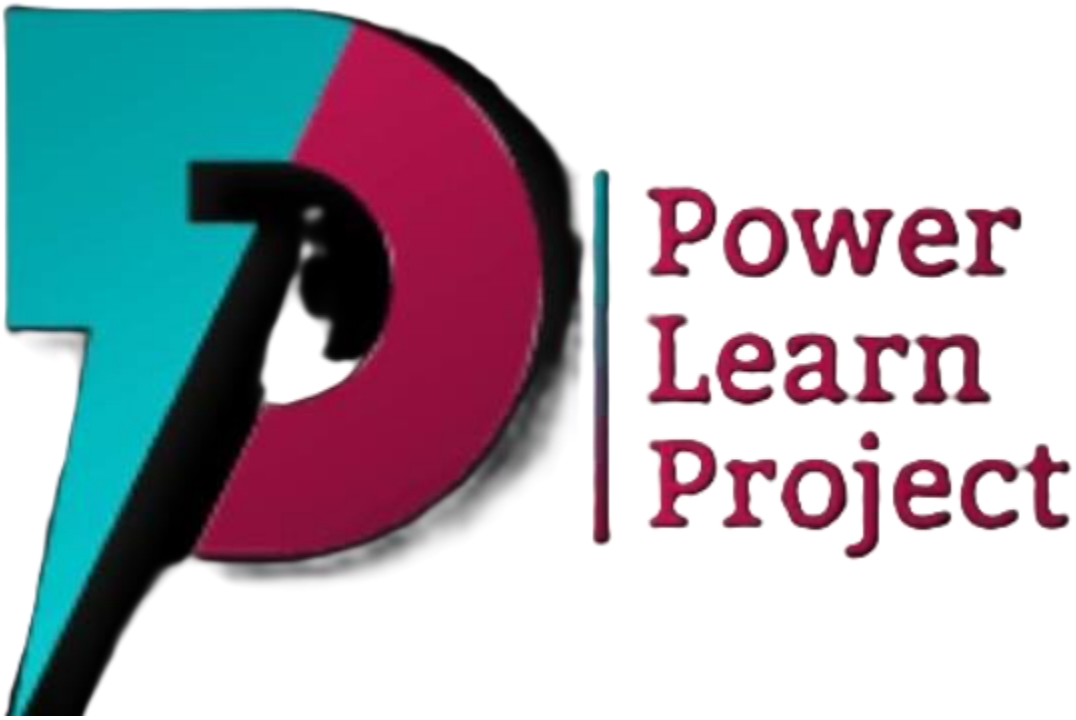


POWER LEARN PROJECT

SCHOOL OF SOFTWARE ENGINEERING SHOLARSHIP

CSC - AI FOR SOFTWARE ENGINEERING

Prepared

BY

BENEDICT OOKO

BSC

SUPERVISOR:

AI Engineer. Evans Mutuku

NOVEMBER 2025

PART ONE

1.(a). Problem Definition (6 points)

Hiring Trend Analysis for Predicting Future Skill Demand

1.(b). List 3 objectives and 2 stakeholders.

i. List 3 objectives

a. To understanding what skills, roles, and qualifications employers are actively seeking in the present job market using job postings and workforce data.

b. To extract and classify skill requirements from job descriptions using NLP techniques.

c. To develop an AI-powered predictive model that forecasts future skill demand based on historical and real-time hiring trends.

ii. List 2 stakeholders

a. Employers

b. Job Seekers

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

a. Prediction Accuracy of Future Skill Demand (%).

2. Data Collection & Preprocessing (8 points)

2. (a). Identify 2 data sources for your problem.

a. Online Job Portals – LinkedIn API

b. Open datasets (Kaggle job market datasets)

2. (b). Explain 1 potential bias in the data.

Job Posting Bias (Industry or Role Overrepresentation)

If the dataset has **more job postings from certain industries or popular job portals**, the AI model may learn trends that favor those sectors while ignoring others.

Example:
If most job postings come from the **tech industry**, the model may overestimate demand for tech-related skills (e.g., Python, cloud computing) and underestimate skills needed in underrepresented sectors like healthcare, education, or manufacturing.

2. (c) Outline 3 preprocessing steps (e.g., handling missing data, normalization).

Handling missing or In-complete data - Some job postings may have missing fields such as required skills, salary, or job location. Removing entries with too many missing values or Add missing values (e.g., fill location using company info).

Cleaning and Normalization for Job Descriptions - apply NLP preprocessing such as: Lowercasing

3. Model Development (8 points)

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

Gradient Boosted Trees (LightGBM / XGBoost)

Why: excellent performance on mixed tabular and engineered text features, fast to train, robust to irrelevant features, handles missing values well, and offers feature-importance explanations (useful for stakeholders).

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

Because this problem depends on time, random split is avoided to prevent temporal leakage.

Use a time-aware split:

Training: earliest 70% of the time range.

Validation: next 15% (used for hyper-parameter tuning and early stopping).

Test: final 15% (held out for the unbiased assessment of forecasting performance).

3. c) Name 2 hyper-parameters you would tune and why.

learning_rate (aka eta) it controls the step size at each boosting iteration.

Reason: lower values often improve generalization but require more trees; tuning balances bias vs variance and affects convergence speed and final accuracy. Use with early stopping.

num_leaves (LightGBM) / max_depth (XGBoost) it controls model complexity (size of individual trees)

Reason: affects model capacity (too large) overfitting to past trends (dangerous for forecasting); under-fitting and missed non-linear relationships between features and skill demand.

4. Evaluation & Deployment (8 points)

4. (a) Select 2 evaluation metrics and explain their relevance.

Mean Absolute Error (MAE): measures the average difference between the predicted skill demand and the actual observed demand.

Relevance:

- It is easy to interpret (in real units of demand).

- Less sensitive to large outliers than RMSE.

- Helps determine how accurately the model forecasts future hiring trends.

F1-Score: F1-score is the harmonic mean of precision and recall.

Relevance:

- Useful when classes are imbalanced (e.g., few skills become "high-demand").

- Ensures the model not only identifies high-demand skills, but does so accurately without too many false positives or false negatives.

- Balances both precision and recall for more reliable decision-making.

4. (b) What is concept drift? How would you monitor it post-deployment?

Concept drift refers to the phenomenon where the statistical properties of the target variable in a predictive model change over time, leading to decreased model accuracy

Track model performance over time using metrics like MAE or F1-score. A consistent decline indicates drift.

Monitor data distributions (skill frequencies, job-posting trends) and compare them with training data.

Use drift detection algorithms such as DDM (Drift Detection Method) or ADWIN.

Set alerts when key features (e.g., emerging skills) show unusual changes

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

Scalability of Real-time Predictions

When deployed, the system may need to process large volumes of incoming job postings or trend data in real time.
Why this is challenging:

- NLP feature extraction (e.g., TF-IDF or embeddings) can be computationally heavy.
- Spikes in traffic from API calls or data ingestion may overwhelm the system.

Result:
Slow response times, failed requests, or delayed trend insights.

Possible solution:
Use distributed systems like Docker and Kubernetes, implement **batch processing**, or use **message queues** (Kafka) to scale efficiently.

PART TWO: Case Study Application (40 points)

Scenario: A hospital wants an AI system to predict patient readmission risk within 30 days of discharge.

1. Problem Scope (5 points): Define the problem, objectives, and stakeholders.

   Build an AI system that, at or before hospital discharge, predicts whether a patient will be readmitted within 30 days.

   Objectives (3–4 measurable objectives):

   a. Accurately predict 30-day readmission risk with a target
   b. Identify top contributing risk factors per patient (explainability) so clinicians can act on modifiable risks
   c. Integrate predictions into discharge workflow and measure downstream impact

   Primary stakeholders:

   a. Clinical team
   b. Hospital administration

2. Data Strategy (10 points):
   a. Data source

   Publicly available datasets from Kaggle or MIMIC-IV Medical Information Mart for Intensive Care).
   Restricted Healthcare Databases like Nationwide Readmissions Database (NRD) or Electronic Health Records (EHRs).

3. Two Ethical concerns to consider:

   Patient privacy and confidentiality.

   Patient making informed consent before his or her data is used.

   Algorithmic bias and fairness

4. Design a preprocessing pipeline (include feature engineering steps).
5. Model Development (10 points):
   a. Select a model and justify it.

   Gradient boosted trees (e.g., XGBoost / LightGBM) or an ensemble that combines GBTs with simple logistic regression baseline.

   Reasons:
   Excellent performance on tabular clinical data and heterogeneous features (structured and engineered NLP features).

Robust to missing values, fast training, interpretable via feature importance and SHAP explanations; useful for clinician acceptance.
Many readmission studies report competitive or superior performance with boosted trees relative to deep networks on EHR tabular data (Davis, 2022).

b. Create a confusion matrix and calculate precision/recall (hypothetical data).

6. Deployment (10 points):
   a. Outline steps to integrate the model into the hospital's system.
      i. **Proof of concept (sandbox):** deploy model in a controlled environment using de-identified data; validate metrics and clinician feedback.
      ii. **Integration architecture:** expose model via a secure REST API or FHIR Clinical Decision Support (CDS) Hooks service that the EHR can call at discharge event. Use containerization (Docker) and orchestrate with Kubernetes for scalability.
      iii. **Workflow embedding:** add model output (risk score and top contributors) inside the discharge summary or in the EHR task queue for discharge planners; include actionable recommendations (e.g., schedule follow-up, home health referral).
      iv. **Monitoring and logging**: implement real-time logging of predictions, input distributions, and outcomes (for monitoring accuracy and drift). Ensure logs are stored securely with limited access.
      v. **Feedback loop and retraining:** capture actual readmission outcomes and periodically retrain models (or trigger retraining when performance degrades).
      vi. **User training and governance:** train clinicians on model use and limitations; set governance for change control and clinical sign-off.
7. How would you ensure compliance with healthcare regulations (e.g., HIPAA)?
   a. Data protection
   b. Conducting Audit and monitoring logging
   c. Continuous compliance: perform periodic security and privacy audits, penetration testing, and compliance reviews to ensure ongoing HIPAA conformance
   d. Business Associate Agreements (BAAs): sign BAAs with vendors that process PHI.
8. Optimization (5 points):
   a. Propose 1 method to address overfitting.
      Dealing with overfitting:
         Simplify the model
            Reduce Complexity: Decrease the number of layers or neurons in your model. A simpler architecture prevents the model from memorizing noise in the training data.
            Feature Selection: Use only the most relevant features to reduce unnecessary complexity.

Part 3: Critical Thinking (20 points)

1. Ethics & Bias (10 points):

   a. How might biased training data affect patient outcomes in the case study?

      Biased data may lead the model to overestimate or underestimate readmission risk for certain groups (e.g., based on age, gender, or ethnicity). This can result in unfair treatment decisions, such as unnecessary interventions for some patients or insufficient care for others.

   b. Suggest one strategy to mitigate this bias.

      Use bias-aware sampling: rebalance the dataset so all demographic groups are fairly represented, and evaluate model performance across subgroups.

2. Trade-offs (10 points):

   a. Discuss the trade-off between model interpretability and accuracy in healthcare.

      More complex models, for instance neural networks - may achieve higher accuracy but are harder to interpret, which is problematic in healthcare where clinicians must understand and trust predictions. Simpler models (e.g., logistic regression) are easier to explain but may offer lower accuracy.

   b. If the hospital has limited computational resources, how might this impact model choice?

      The hospital may need to use lightweight, low-complexity models (e.g., logistic regression or decision trees) that require less memory, compute power, and time, instead of resource-intensive deep learning models.

Part 4: Reflection & Workflow Diagram (10 points)

Reflection (5 points):

1. What was the most challenging part of the workflow? Why?

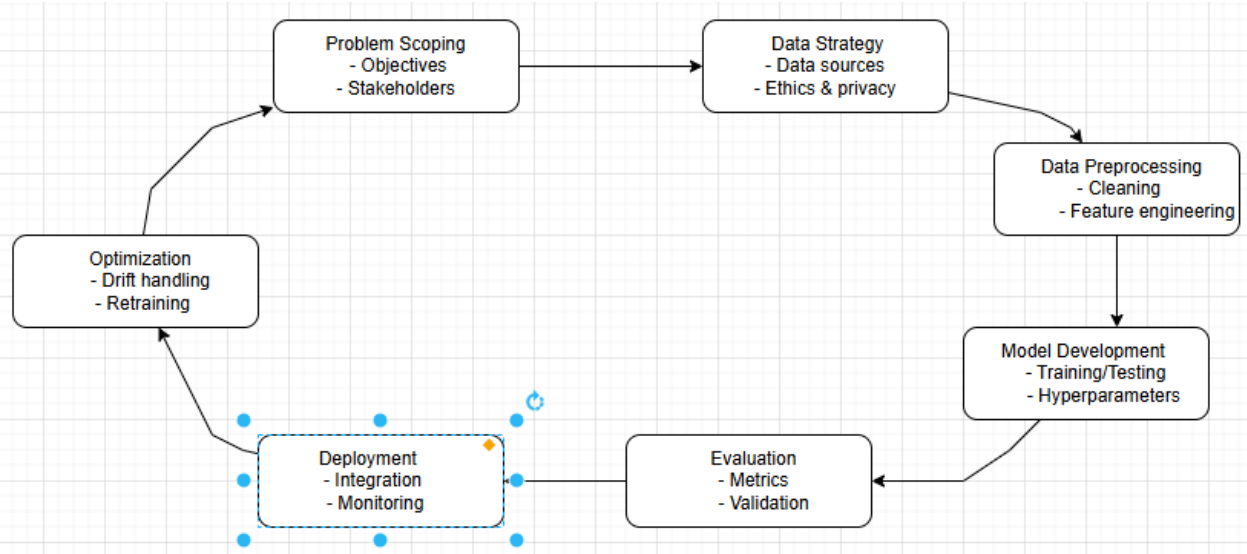
The most challenging part was feature engineering, because healthcare data is complex, inconsistent, and often contains missing or sensitive information. Transforming raw clinical data into meaningful features required careful preprocessing and domain understanding.

2. How would you improve your approach with more time/resources?

With more time and resources, I would incorporate larger, more diverse datasets, collaborate with medical experts for deeper clinical insights, and perform more extensive hyper-parameter tuning to improve model accuracy and fairness.

Diagram (5 points):

Sketch a flowchart of the AI Development Workflow, labeling all stages.



REFERENCE

Davis, S., Zhang, J., Lee, I. *et al.* Effective hospital readmission prediction models using machine-learned features. *BMC Health Serv Res* **22**, 1415 (2022). https://doi.org/10.1186/s12913-022-08748-y