Data Science & LLM Technical Assessment

Joseph Farrington, June 2025

1. Predictive Modelling

Approach: I trained and evaluated a scikit-learn RandomForestClassifier to predict readmission within 30 days. I first split the 200 patient stays into a training set (80%) and held-out test set (20%), stratified by the label because 62.5% of the patients were not readmitted.

I chose RandomForestClassifier because it is a standard baseline for binary classification and can handle the non-monotonic relationship I observed between age and likelihood of readmission without manual feature engineering. EDA showed no missing data and that numeric features were within reasonable ranges. Numeric features were not preprocessed, categorical features were low cardinality and were therefore one-hot encoded, and TF-IDF features were created for the discharge notes after removing English stop words.

I tuned the hyperparameters of the RandomForestClassifier, including class_weight to address the imbalance, with stratified 5-fold cross-validation using Bayesian optimization with Optuna. I selected F1 score as the metric to optimize because it equally weights precision and recall, and we have no details about the potential use of the model.

I trained the final classifier using the best identified hyperparameters on the full training set and evaluated its predictions on the test set. I estimated the contribution of each feature using permutation feature importance, calculated on the test set as the reduction in F1 score when the feature column is shuffled.

Results: The model gave an F1 score of 0.50 and a ROC AUC of 0.60 on the test set. The confusion matrix on the test set is presented in Figure 1(a). The permutation feature importances on the test set are presented in Figure 1(b): the free-text discharge notes are the most important feature.

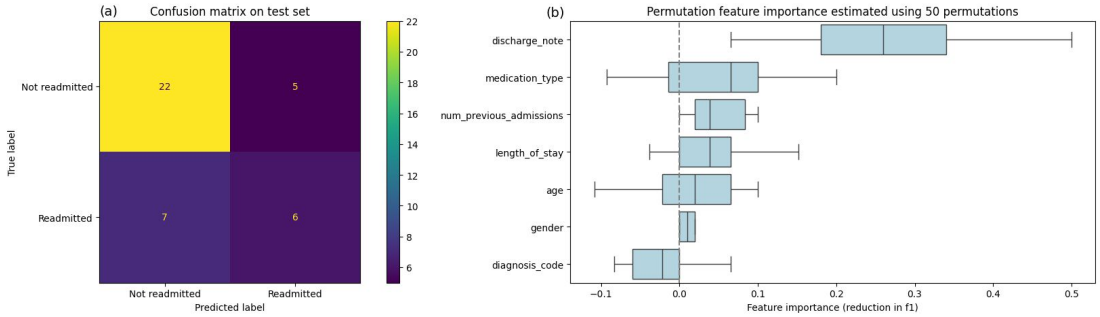

Figure 1: Confusion matrix for the test set (a) and permutation feature importances on the test set (b).

Practical implications: The current model does not identify half of the patients who will be readmitted in 30 days and only half of the patients it predicts will be readmitted within 30 days ultimately are. While the intended use of the model is not known, the current performance is likely insufficient for practical use.

Future work: The appropriate balance between precision and recall will depend on the intended use and the relative cost of false negatives and false positives. With this information, the model could be tuned to maximise ROC AUC or average precision and a threshold selected to best match the use case. Confidence intervals could be estimated on metrics using bootstrapping. Features for the free-text could be created using embedding models to better represent the context including negation.

2. Named Entity Recognition from Discharge Notes

Approach: I adopted a few-shot approach using an open-source instruction fine-tuned LLM because no labelled examples were provided. To reduce hallucinations, I defined a structured output for the model using Pydantic. I used Ollama to serve the model and Langchain as the Python interface because they support structured outputs. I used Meta AI's Llama3.2:3b as the LLM because it is the most recent Llama model supported by Ollama I was able to run locally. I wrote a prompt template, including a description of each entity and example extractions, into which the discharge note is inserted during inference. To detect hallucinations, I wrote a validation function to check if each extracted entity is a verbatim extract from the provided note.

Results: The LLM pipeline generates output in the structured format, and "entities" which are verbatim extracts from the discharge notes. The provided free text does not provide good coverage of the entities, and no entity labels were provided, so no metrics have been computed on whether the extracted entities are complete and correct.

Practical implications: A few-shot approach to extracting developer-specified medical entities using an instruction fine-tuned LLM appears feasible but a proper set of evaluation cases is needed to quantify and optimize performance.

Future work: Develop a suitable set of labelled evaluation cases so that precision and recall can be computed to quantify the performance and enable comparisons between different models and prompts. Incorporate the validation check for hallucinations into the pipeline, with errors fed back into the LLM until a hallucination-free response is produced. Compare the performance to a smaller, fine-tuned, task-specific model which may be cheaper and faster.