Introduction

Team Members

Lee Napthine
Computer Science

Parker DeBruyne
Computer Science & Health Information Science

Wesley Ducharme
Computer Science & Statistics

Shaban Shala
Software Engineering

Predicting Patient Visit Duration

via Synthetic Electronic Health Records

Hospitals Struggle To Manage Staffing

Bed Hours Occupied (BHO)

- **BHO = Discharge Time - Arrival Time**
- The total time a patient occupies a bed
- Summed over a period for all patients

BHO Usage

- Analyze Capacity Management
- Operational Efficiency (bottlenecks)
- Resource Allocation
- Staffing & Finances
- Quality of Care
- Policy & Planning

Without BHO Forecasts

- Inaccurate staffing
- Nurse overwork
- Resource strain
- Longer patient wait times
- Decreased Quality of Care

Data Source: Synthea

Synthetic Health Data

- Open-source software tool that generates synthetic, realistic electronic health records (EHRs)
- Simulates patient lifecycles and healthcare interactions

Research & Development

- Allows researchers and developers to test and refine health IT applications
- Explore data science concepts
- Conduct simulations

Privacy

- Entirely synthetic
- Anonymous
- i.e, No concerns with Data Privacy regulations

https://synthetichealth.github.io/synthea/

Pivot: Patient Visit Duration

m = 53,347

Now:

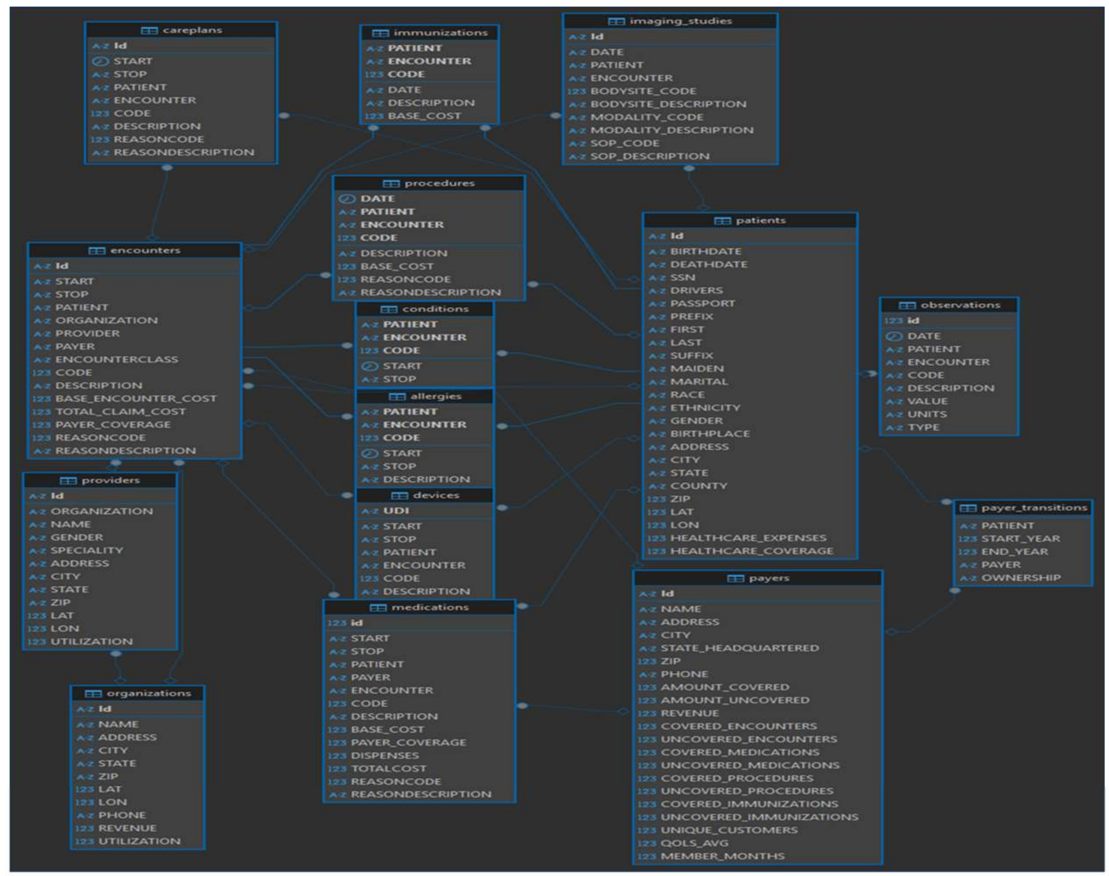
- 53,347 Patient Encounters
- Enriched with individual Patient health data:
  - Age, Medications, Immunizations
- Enriched with Calendar data:
  - Day of the Week, Month, Year, Holidays
- Focused more on method and implementation

Methods

Setting up the Data

First steps:

- Made a PostgreSQL database to store the datasets for future querying.
- Hosted it on Railway for shared access.
- Queried the database to engineer our feature set and target variable. (Also engineered some features in our python scripts)
- Limited our data from 2010-2020.



Features and Target

Initial feature set included:

- Description: Reason for the medical encounter
- Reason Description: Extra details about the encounter (if any)
- Gender
- Patient Age
- Organization Name
- And more . . .

- Initial Target: Duration of Encounter (hours)

Initial Models

Linear regression and SGD models with degree 2 polynomial features.
Neural network with 1 hidden layer and 64 neurons. (RELU activation).

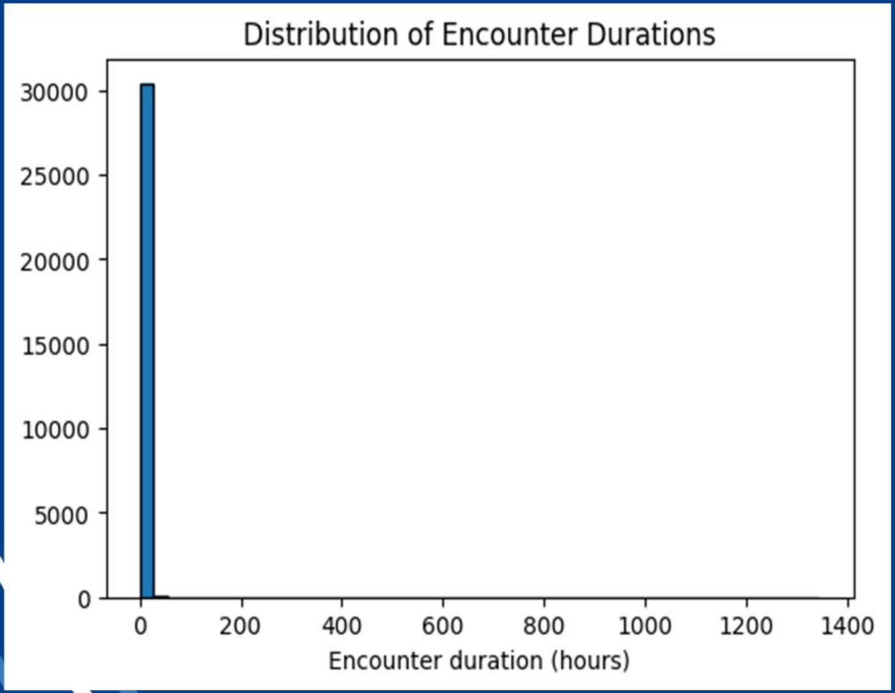
Both resulted in:

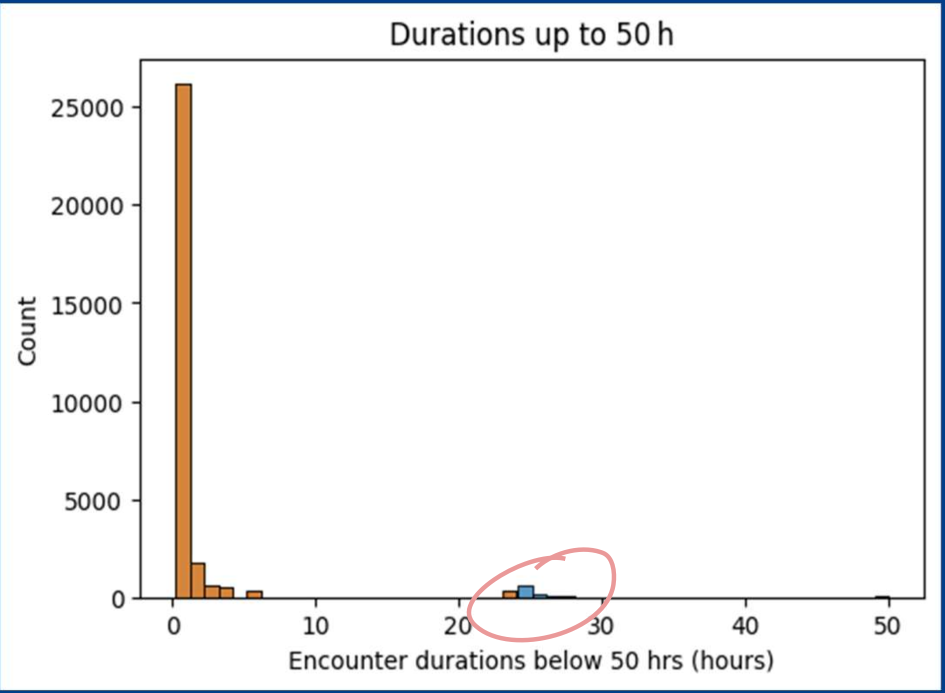
- very low R^2(approx 0.00001) metric
- very high RMSE(approx 10000+)
- log / sqrt transformations lead to no improvement

What could be causing problems?

Extreme outliers!

Target Distribution





New Approach

Swapped to a classification approach.

New target variable: encounter_length_class

Split it into 6 categories of duration length in hours:

- Quick (duration <= 0.25)
- Short (0.25 < duration <= 3)
- Moderate (3 < duration <= 8)
- Long (8 < duration <= 24)
- Very Long (24 < duration <= 50)
- Extreme (duration > 50)

Allowed us to both deal with the outlier problem while still giving accurate representation of the data.

New Models

Now a classification problem!

New models that we will be comparing:

- Multinomial Logistic Regression model
- 3 Layer Neural Network with the new target

Workflow and Debugging

Common workflow in both models:

- 80% training, 20% testing
- 3-Fold cross validation on the training set to fine tune hyper parameters.
- One last round of training and validation testing to view learning curves for possible overfitting and underfitting.
- After tuning do one final training and testing run on all training data and the unseen test data.

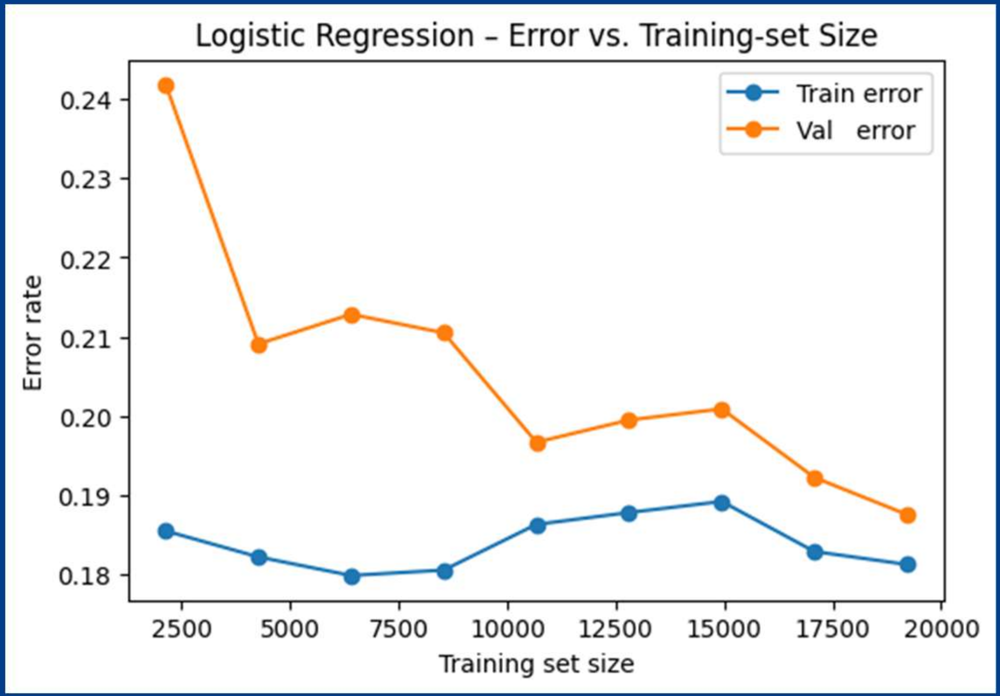
Debugging approach:

- Baseline was 70% accuracy and minimal overfitting / underfitting
- Check for Bias and Variance with learning curves
- Make changes appropriately to reduce overfitting / underfitting

The Logistic Model

- Used Scikit-learn's built in Logistic Regression
- Set the class weight parameter to balance the weights
- Noticed highest accuracy with degree 2 polynomial features
- Initial learning curves showed high variance with no convergence
    - Introduced feature reduction to reduce this variance

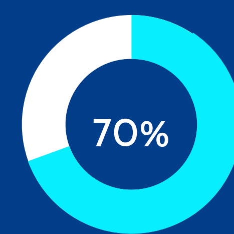
Early Implementation Learning Curve

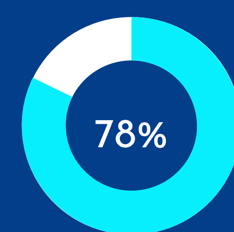


Logistic Regression Results

Test set results:

	precision	recall	f1-score
Quick (≤0.25h)	0.79	0.83	0.81
Short (0.25-3h]	0.82	0.70	0.75
Moderate (3-8h]	0.47	0.99	0.63
Long (8-24h]	0.85	0.89	0.87
Very Long (24-50h]	0.71	0.98	0.82
Extreme (>50h)	0.65	0.96	0.78
accuracy			0.78
macro avg	0.71	0.89	0.78
weighted avg	0.79	0.78	0.78





LR Learning Curve

The Neural Network

- Used PyTorch to build the NN
- Balanced the class weights manually
- Early learning curves showed extreme variance as before
- Tried multiple layers and neurons assignments

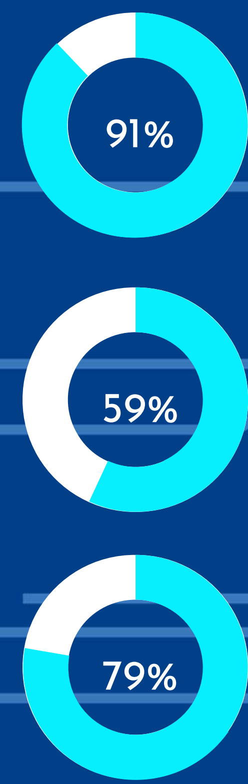
Final setup: 3 hidden layers

- First layer 128 Neurons
- Second layer 64 Neurons
- Third layer 32 Neurons
- RELU activation function
- Used dropout regularization (p_drop=0.35)

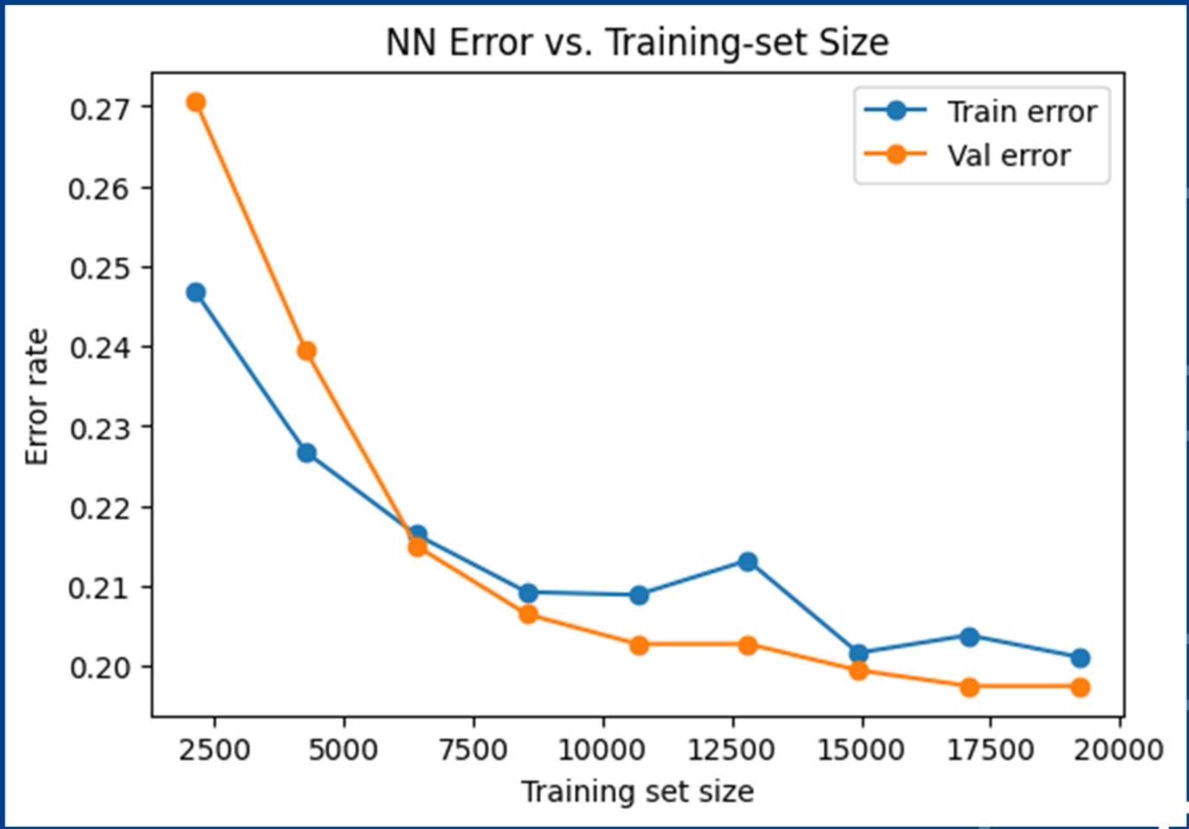
(This being the final setup after having tested and checked multiple other layer and Neurons distributions gave the best results)

Neural Network Results

	precision	recall	f1-score
Quick (≤0.25h)	0.73	0.96	0.83
Short (0.25–3h]	0.96	0.59	0.73
Moderate (3–8h]	0.50	0.98	0.66
Long (8–24h]	0.81	0.96	0.88
Very Long (24–50h]	0.88	0.98	0.93
Extreme (>50h)	0.48	1.00	0.65
accuracy			0.79
macro avg	0.73	0.91	0.78
weighted avg	0.83	0.79	0.78



NN Learning Curve



Comparisons

Results Comparisons

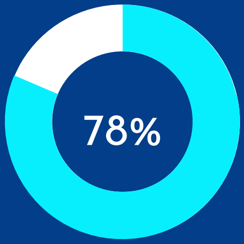
Logistic Regression

Test set results:

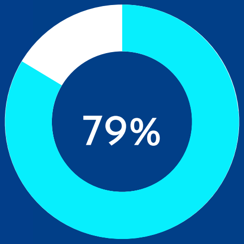
	precision	recall	f1-score
Quick (≤0.25h)	0.79	0.83	0.81
Short (0.25-3h]	0.82	0.70	0.75
Moderate (3-8h]	0.47	0.99	0.63
Long (8-24h]	0.85	0.89	0.87
Very Long (24-50h]	0.71	0.98	0.82
Extreme (>50h)	0.65	0.96	0.78
accuracy			0.78
macro avg	0.71	0.89	0.78
weighted avg	0.79	0.78	0.78

Neural Network

	precision	recall	f1-score
Quick (≤0.25h)	0.73	0.96	0.83
Short (0.25-3h]	0.96	0.59	0.73
Moderate (3-8h]	0.50	0.98	0.66
Long (8-24h]	0.81	0.96	0.88
Very Long (24-50h]	0.88	0.98	0.93
Extreme (>50h)	0.48	1.00	0.65
accuracy			0.79
macro avg	0.73	0.91	0.78
weighted avg	0.83	0.79	0.78



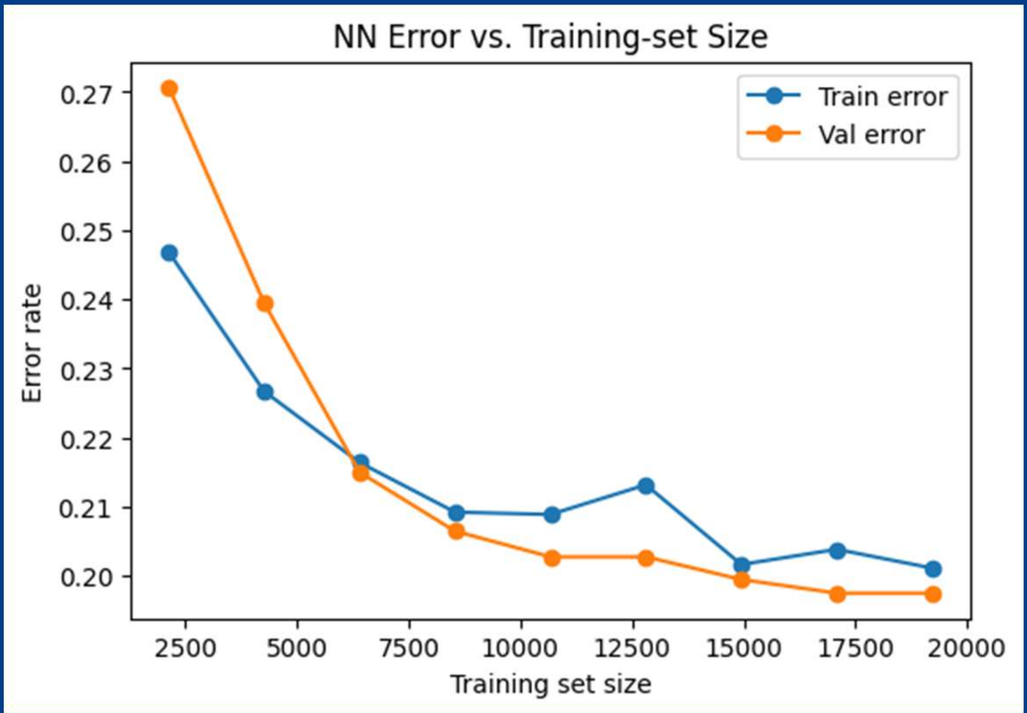
Logistic Regression Accuracy



Neural Network Accuracy

Visual Comparisons

Neural Network



Logistic Regression

Conclusions

Neural Networks captured patterns in our healthcare data with an impressive level of recall

Recall was our prime metric, as missing on an outlier prediction could be costly.

Best performance came from cost bucket classification

Reached 79% accuracy, and our recall metrics were strong at predicting lengthy visits.

Model accuracy suffered when predicting short to moderate length visits with an average 54% accuracy.

Data preprocessing and feature encoding are critical

Highest weighted features: visit type, visit description, medication costs, and age.

These were predictably influential and reinforced our trust that the model was working.

Neural nets are powerful even with limited tuning

With finer tuning our model could be capable of predicting encounter duration with enough accuracy for hospital planning.

Limitations of Our Approach

Limited time frame

The dataset was very large and we could have spent more time engineering, testing, and iterating our features and model.

Synthetic data

Very large synthetically generated dataset likely overstates the accuracy.

Features grew too-large too-fast

Many of the features needed to be categorically encoded losing their true semantic meaning.

Future Improvements

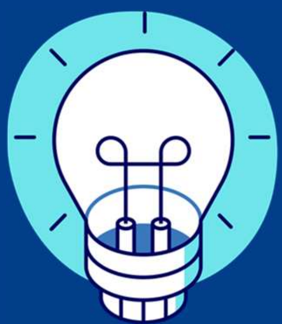
Improve mid-range accuracy

The model struggled with mid-range visit durations. We plan to engineer additional features to address this gap and understand the causes

Explore other types of NN's for time series cost forecasting

Certain studies had success using Recurrent Neural Networks for analysis on large medical datasets.

Future Applications

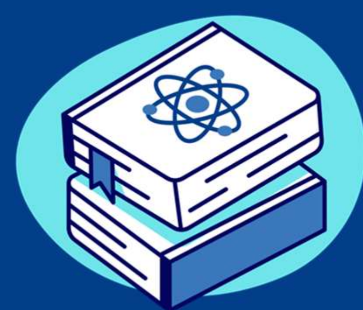


Insurance analytics

Evaluating risk groups and costs

Hospital resource planning

Staff and bed scheduling



Public health

Cost projections by region and Statistics Canada application

Fairness and Bias

Feature engineering and model training may enforce biases that affect certain population over others

Ongoing evaluation is needed to prevent unfair outcomes

Thank you!