Joey Everette, Colton Hammond, Ian Young

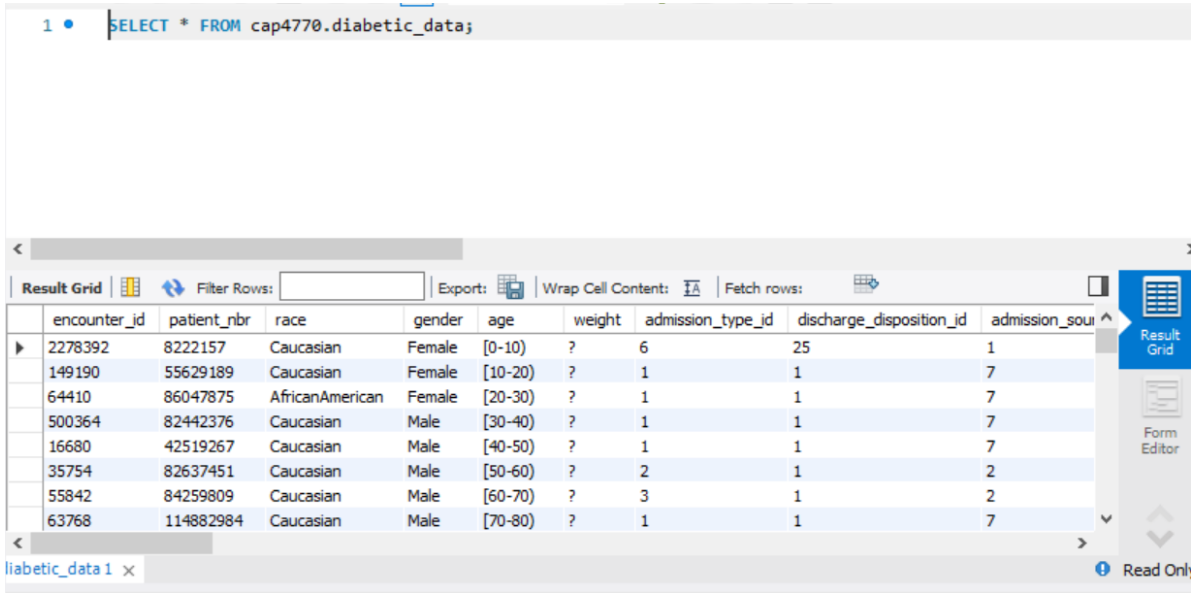
Final Project Report

## Problem Statement and Background Information

In this project, we are aiming to predict the likelihood of hospital readmission within 30 days for diabetic patients using the "Diabetes 130-US hospitals" dataset. By identifying key features such as age, gender, and race that influence early readmissions.

## Data must be loaded into MySQL (or other) database

The dataset we used for this project, *Diabetes 130-US hospitals for years 1999–2008*, was loaded into a MySQL database for data handling and easy integration into the Jupyter notebook environment. We used MySQL Workbench to create a database named cap4770 and imported the diabetic_data.csv file using the built-in Table Data Import Wizard. Once the data was loaded, we connected MySQL to  Jupyter Notebook using the SQLAlchemy and Mysql-connector-python libraries. This allowed us to query and load the dataset into a pandas DataFrame for preprocessing, exploration, and modeling.




*Screen snip of dataset loaded into MySQL Workbench*

## Data Collection and Preprocessing

**Data Collection:** The dataset used in this project is the "Diabetes 130-US hospitals for years 1999–2008" dataset, originally obtained from the UCI Machine Learning Repository. It contains over 100,000 records of inpatient encounters for diabetic patients and includes 47 features such as demographics, diagnoses, lab results, and medication usage. We downloaded the dataset as a CSV file (diabetic_data_csv) and then imported the dataset into MYSQL Workbench before connecting it to Jupyter Notebook.

**Data Preprocessing:** We implemented different cleaning methods to make the dataset more streamlined and accurate. Some steps taken were to clean up missing values, remove rows that were missing important values, and the use of a target variable.

```
import pandas as pd
import numpy as np
from sqlalchemy import create_engine
from sklearn.preprocessing import LabelEncoder

# MySQL connection
engine = create_engine("mysql+mysqlconnector://root:Joseph1985!!@localhost/cap4770")

# Load data
query = "SELECT * FROM diabetic_data"
df = pd.read_sql(query, con=engine)

# Replace '?' with NaN
df.replace('?', np.nan, inplace=True)

# Drop unneeded columns
df.drop(columns=['weight', 'payer_code', 'medical_specialty', 'encounter_id', 'patient_nbr'], inplace=True)

# Drop rows with missing race, gender, or readmitted
df.dropna(subset=['race', 'gender', 'readmitted'], inplace=True)

# Create binary target
df['readmit_30'] = df['readmitted'].apply(lambda x: 1 if x == '<30' else 0)
df.drop(columns=['readmitted'], inplace=True)

# Label encode categoricals
cat_cols = df.select_dtypes(include='object').columns
encoders = {}

for col in cat_cols:
    le = LabelEncoder()
    df[col] = le.fit_transform(df[col])
    encoders[col] = le
```

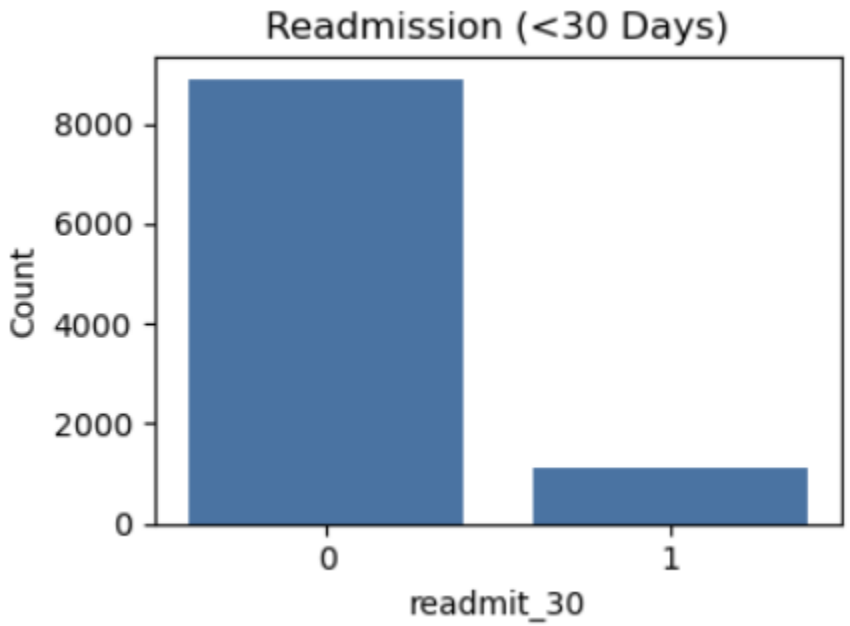
*Screen snip of code for data preprocessing*

## Exploring Data

To better understand the structure and distribution of the data, several exploratory data analysis (EDA) techniques were applied. This helped us identify feature distributions, target class imbalance, and early relationships between variables and readmission outcomes.

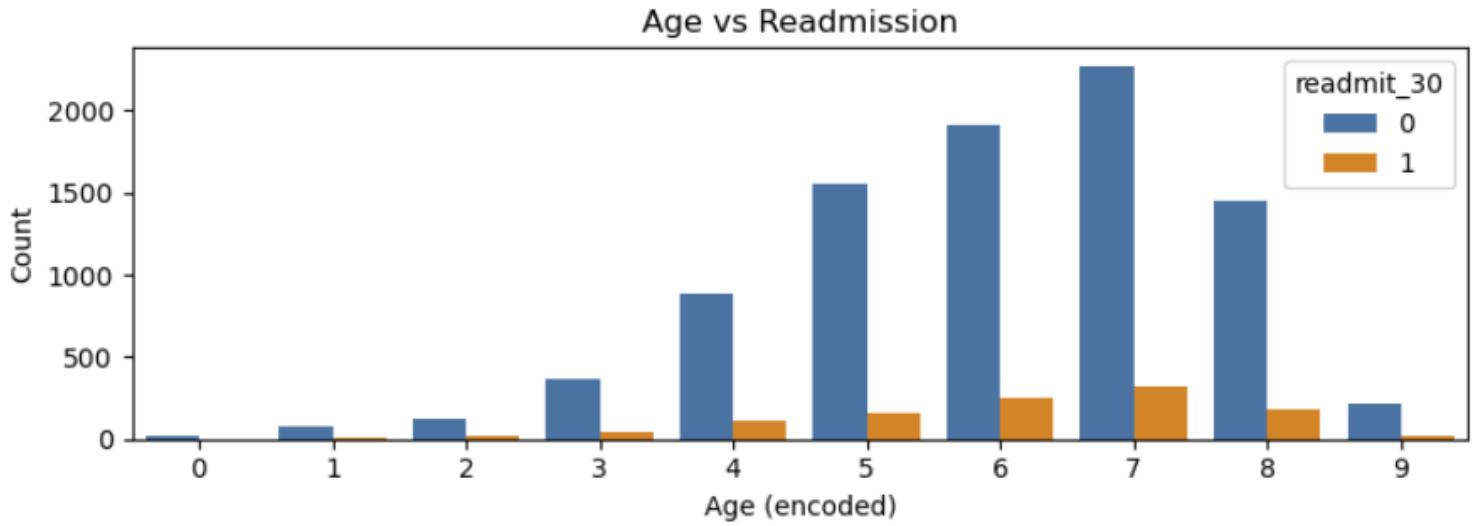
A random sample of 10,000 records was used to make visualizations more efficient while preserving overall patterns.

A count plot of the readmit_30 variable revealed a class imbalance, with the majority of patients not readmitted within 30 days.

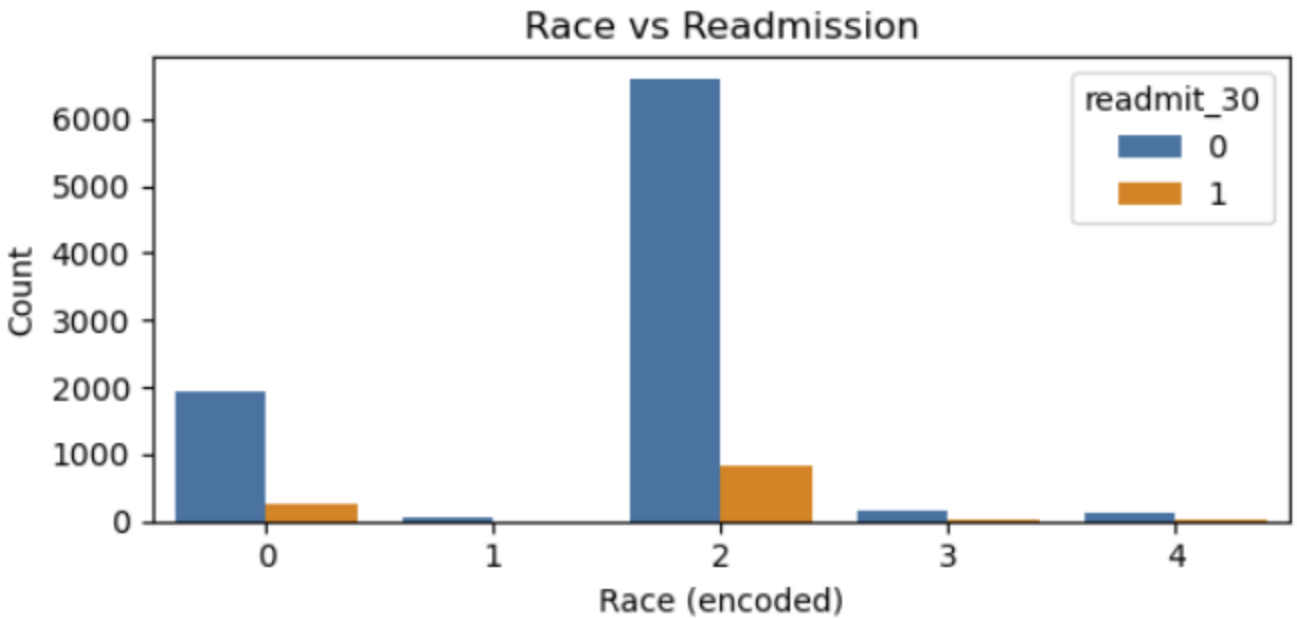



*The count plot shows that the majority of patients were not readmitted within 30 days.*

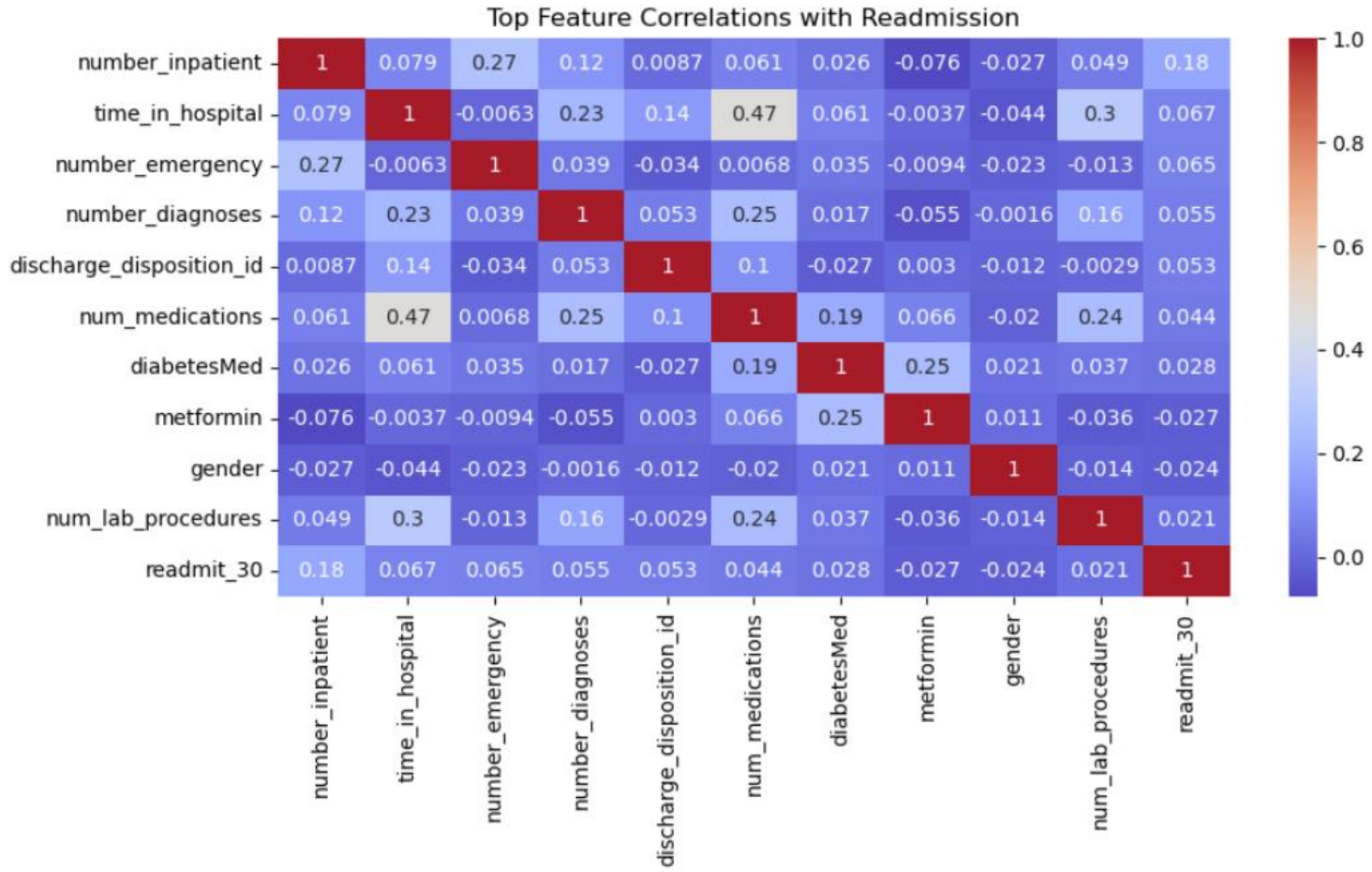
The age feature was compared against readmission outcomes. Patients in older age brackets (e.g., [70-80), [60-70)) had slightly higher rates of readmission, though all age groups leaned heavily toward non-readmission.




While the majority of the dataset consisted of Caucasian and African American patients, no dramatic differences in readmission rates were observed across races. However, this feature was retained for model training to account for possible indirect relationships.




To identify which features had the strongest relationships with readmit_30, a correlation heatmap was generated. Features like number_inpatient, number_emergency, and number_diagnoses showed moderate positive correlation with early readmission.




After exploring the data, we concluded that the dataset is “imbalanced” favoring non-readmitted cases. Features related to previous visits (inpatient, emergency) are most correlated with readmission and categorical features like race and gender only showcased minor discrepancies, but we still kept them in the model for research purposes.

## Data Modeling

We used three different modeling techniques to predict whether diabetic patients would be readmitted within 30 days. The classification models used were Logistic Regression, Decision Tree Classifier, and Random Forest Classifier. To improve performance and reduce computation time, a random sample of 10,000 records was used for training and evaluation. The data was split into 80% training and 20% testing.

The dataset was split using scikit-learn's train_test_split function.

```
# Sample 10,000 rows for faster modeling
df_model = df.sample(n=10000, random_state=42)

# Split features and target
X = df_model.drop(columns=['readmit_30'])
y = df_model['readmit_30']

# Train/test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

*Screen snip of test_split function*

Each model was fit on the training data and evaluated on the test data using precision, recall, F1-score, and confusion matrix.

```python
# Define models
models = {
    "Logistic Regression": LogisticRegression(max_iter=1000, solver='liblinear'),
    "Decision Tree": DecisionTreeClassifier(),
    "Random Forest": RandomForestClassifier()
}
```

*Screen snip of modeling code.*

The metrics provided insight into the overall accuracy and how well each model predicted the minority class (readmit_30 = 1).

```python
# Train + evaluate
results = {}

for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    report = classification_report(y_test, y_pred, output_dict=True)
    matrix = confusion_matrix(y_test, y_pred)
    results[name] = {
        "Classification Report": report,
        "Confusion Matrix": matrix
    }
```

*Screen snip of evaluation metrics code*

## Results & Interpretation

After training and evaluating three machine learning models, Logistic Regression, Decision Tree, and Random Forest, the results showed that while all models achieved high overall accuracy, they struggled to correctly identify patients who were readmitted within 30 days. This was largely due to class imbalance, where the majority of patients in the dataset were not readmitted.

**Logistic Regression**

- Accuracy: ~90%
- Precision (readmitted): 50%
- Recall (readmitted): 2.9%
- F1-Score (readmitted): 5.6%
- Confusion Matrix: Predicted only 6 out of 203 actual readmissions

**Decision Tree**

- Accuracy: ~80%
- Precision (readmitted): 14%
- Recall (readmitted): 19.2%
- F1-Score (readmitted): 16.2%
- Confusion Matrix: Predicted 39 out of 203 actual readmissions

**Random Forest**

- Accuracy: ~90%
- Precision (readmitted): 50%
- Recall (readmitted): 0.5%
- F1-Score (readmitted): 0.97%
- Confusion Matrix: Predicted only 1 out of 203 actual readmissions

All three models were highly effective at predicting non-readmission cases (the majority class), but performed poorly on detecting early readmissions, the class we were most interested in. Among the models, the Decision Tree performed slightly better in identifying true positives (patients readmitted within 30 days), but at the cost of lower overall accuracy.
This imbalance highlights a common issue in healthcare datasets. Important/strong outcomes may be underrepresented, making them harder for models to learn.

## Conclusion

This project explored the use of machine learning to predict 30-day hospital readmissions for diabetic patients using a real-world healthcare dataset. After cleaning and preprocessing the data, we trained three models: Logistic Regression, Decision Tree, and Random Forest.

While all models achieved high accuracy overall, they struggled to correctly identify early readmissions due to class imbalance. The Decision Tree performed slightly better in identifying these cases, but with reduced accuracy.

## Works Cited


Anastasopoulou, L., Anastasopoulos, E., Theofanopoulou, M., Siafakas, C., & Fotiadis, D. I. (2023). *Predicting 30-day hospital readmission in patients with diabetes using machine learning on electronic health record data*. Cureus, 15(1), e34567. https://doi.org/10.7759/cureus.34567

**MySQL**

Oracle. (2023). *MySQL 8.0 Reference Manual*. Oracle Corporation. https://dev.mysql.com/doc/

**Pandas (Data Handling)**

McKinney, W. (2022). *pandas: Powerful Python data analysis toolkit*. https://pandas.pydata.org/

**scikit-learn (Machine Learning Models)**

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). *Scikit-learn: Machine learning in Python*. Journal of Machine Learning Research, 12, 2825-2830.

https://jmlr.org/papers/v12/pedregosa11a.html

**Matplotlib & Seaborn (Visualizations)**

Hunter, J. D. (2007). *Matplotlib: A 2D graphics environment*. Computing in Science & Engineering, 9(3), 90-95.

https://doi.org/10.1109/MCSE.2007.55

Waskom, M. L. (2021). *Seaborn: Statistical data visualization*. Journal of Open Source Software, 6(60), 3021.

https://doi.org/10.21105/joss.03021

- **Logistic Regression:**

  Hosmer, D. W., Lemeshow, S., & Sturdivant, R. X. (2013). *Applied Logistic Regression* (3rd ed.). Wiley.

- **Decision Trees & Random Forests:**

  Breiman, L. (2001). *Random forests*. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324

**Project Breakdown**

Part 1: Assigned to Joey Everette

- Loaded the dataset into **MySQL** using MySQL Workbench.
- Connected the database to **Jupyter Notebook** using SQLAlchemy.
- Handled data cleaning: replaced missing values, dropped irrelevant columns
- Created the target variable `readmit_30`
- Encoded categorical variables for modeling

Part 2: Assigned to Colton Hammond

- Conducted EDA using **pandas**, **matplotlib**, and **seaborn.**
- Created and interpreted visualizations (e.g., readmission distribution, age vs. readmission, race vs. readmission, and correlation heatmap.)
- Summarized insights from the visualizations and added them to the final report.
- Helped determine which features would be most useful for modeling.

Part 3: Assigned to Ian Young

- Built and trained **Logistic Regression**, **Decision Tree**, and **Random Forest** models using **scikit-learn**
- Evaluated each model using accuracy, precision, recall, F1-score, and confusion matrix

- Interpreted the results, discussed class imbalance issues, and documented recommendations for future work