Predicting Meeting Room Occupancy

# Introduction

This document outlines the steps for developing a predictive model to estimate the occupancy percentage of meeting rooms based on the day of the week and the hour. The process involves data preprocessing, training a neural network model, and deploying the model for predictions.

# 1. Data Preparation

## 1.1 Loading Data

Start by loading the meeting room data from CSV files using pandas. Two datasets are used: hackathon-schema.csv for the schema and meeting-rooms.csv for the meeting room bookings.

## 1.2 Parsing Dates

Convert the date columns in both datasets to datetime objects to enable easy extraction of the day of the week.

## 1.3 Extracting Day of the Week

For each entry in the datasets, calculate the day of the week from the date column and create a new column to store this information.

# 2. Feature Selection and Preprocessing

## 2.1 Selecting Features

From the meeting-rooms.csv dataset, select relevant features for the model while dropping unnecessary ones such as 'row' and 'date' columns.

## 2.2 Target Variable Creation

Combine the occupancy columns (nineToEleven, elevenToOne, oneToThree, threeToFive) into a single binary target variable that indicates if the room is occupied at any time within the specified ranges.

## 2.3 Splitting Data

Split the data into training and test sets to evaluate the model's performance on unseen data.

## 2.4 One-Hot Encoding

Use a ColumnTransformer with OneHotEncoder to convert categorical data into a format that can be provided to the model, focusing on the 'room' feature.

# 3. Model Building and Training

## 3.1 Defining the Model

Construct a Sequential neural network model with Dense layers and Dropout for regularization.

## 3.2 Compiling the Model

Compile the model with the Adam optimizer and binary cross-entropy loss function, tracking accuracy as the performance metric.

## 3.3 Training the Model

Fit the model on the transformed training data over a specified number of epochs, adjusting parameters like batch size as needed.

# 4. Model Evaluation

## 4.1 Testing the Model

Evaluate the model on the test set to obtain the accuracy and loss, providing insights into its performance on unseen data.

## 4.2 Predictions and Accuracy

Use the model to predict occupancy on the test set, comparing these predictions against the actual values to calculate the model's accuracy.

# 5. Model Deployment

## 5.1 Saving the Model and Preprocessor

Save the trained model and preprocessor to disk for later use in predictions.

## 5.2 Creating a Prediction API

Implement a Flask application that loads the saved model and preprocessor, offering an endpoint for occupancy predictions based on incoming JSON data.

## 5.3 Making Predictions

Detail how to make POST requests to the Flask application with the necessary data to receive occupancy predictions.

# Conclusion

Summarize the process, highlighting the model's ability to predict meeting room occupancy based on the day and hour with a given accuracy. Emphasize the potential applications of this model in optimizing room bookings and managing resources efficiently.

# Appendices

Include any additional information, such as code snippets, library documentation, or data schema details, to provide further context or support for the processes described.