Configuration Manual

MSc Data Analytics
Research Project

Thushar Thekkekaripurath Krishnankutty
23181648

School of Computing
National College of Ireland

Supervisor: Shubham Subhnil

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Thushar Thekkekaripurath Krishnankutty		
Student ID:	23181648		
Programme:	MSc Data Analytics	**Year:**	2024/2025
Module:	Research Project		
Supervisor:	Mr. Shubham Subhnil		
Submission Due Date:	12/12/2024		
Project Title:	Predicting Hospital Readmissions with a Hybrid LSTM-CNN Model: An Evaluation of Deep Learning Techniques in Healthcare Analytics		
Word Count:	739	**Page Count:**	10

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Thushar Thekkekaripurath Krishnankutty

Date: 11/12/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Thushar Thekkekaripurath Krishnankutty
23181648

Introduction:

This document explains how to set up and deploy (deploy) the hospital readmission prediction system created during this project. A very sophisticated hybrid model of Long Short Term Memory (LSTM) and Convolutional Neural Network (CNN) technologies is used by the system. The integration of these deep learning techniques is designed to improve the accuracy of predicting hospital readmissions in support of better patient management and healthcare planning.System Requirements:

To guarantee efficient model processing and to minimize the duration required, it's crucial to be equipped with the necessary hardware and software resources.

1.1. Hardware Requirements:

The implementation is performed on an HP Pavilion; the configuration of the device is as follows.

1. Processor:	Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
2. RAM:	8.00 GB (7.85 GB usable)
3. Hard Disk:	512 GB SSD
4. OS	Windows 10 Pro 64 – bit

1.2 Software Requirements:

Before beginning the model construction phase, the below mentioned software, libraries, and tools were set up and installed on the system.

Software/Tools	Version	Information
Python		To develop the model python is used in this project.
Anaconda		A highly favoured platform within the data science community, Anaconda provides its users with the ability to work computationally, manage libraries and deploy models within a friendly environment for Windows

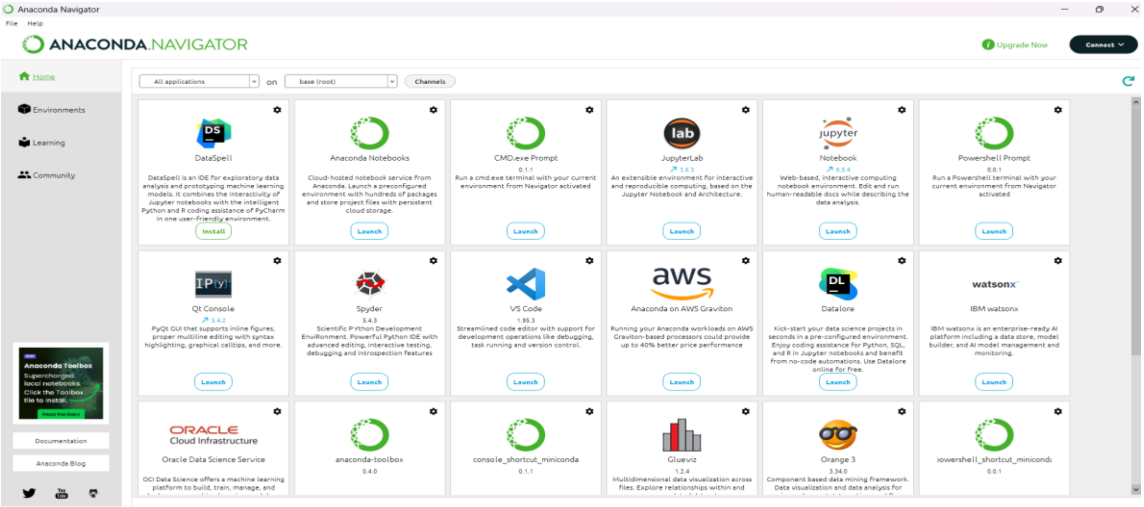
Pandas		It is especially well suited to the task of dealing with tabular data, that is data in spreadsheets or databases.
NumPy		While NumPy is an open source tool from 2023, we can use it for handling complex mathematical problems for data.
Tensorflow		TensorFlow is an open source library developed by Google majorly for deep learning applications.
sklearn		It offers a full set of supervised and unsupervised learning algorithms.
matplotlib		Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.
imblearn		The methods we use to generate a data set with an equal ratio of classes are called Imblearn techniques.

2. Implementation:

In this section there is a complete guide to run the project in any windows system.

1. Download and Install Anaconda Software in the windows system. (https://www.anaconda.com/products/individual)

2. Open the Jupyter Notebook from Anaconda.



DataSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jupyter notebooks with the intelligent Python and R coding assistance of PyCharm in one user-friendly environment.

Cloud-hosted notebook service from Anaconda. Launch a preconfigured environment with hundreds of packages and store project files with persistent cloud storage.

Run a cmd.exe terminal with your current environment from Navigator activated

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Run a Powershell terminal with your current environment from Navigator activated

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Streamlined code editor with support for development operations like debugging, task running and version control.

Running your Anaconda workloads on AWS Graviton-based processors could provide up to 40% better price performance

Kick-start your data science projects in seconds in a pre-configured environment. Enjoy coding assistance for Python, SQL, and R in Jupyter notebooks and benefit from no-code automations. Use Datalore online for free.

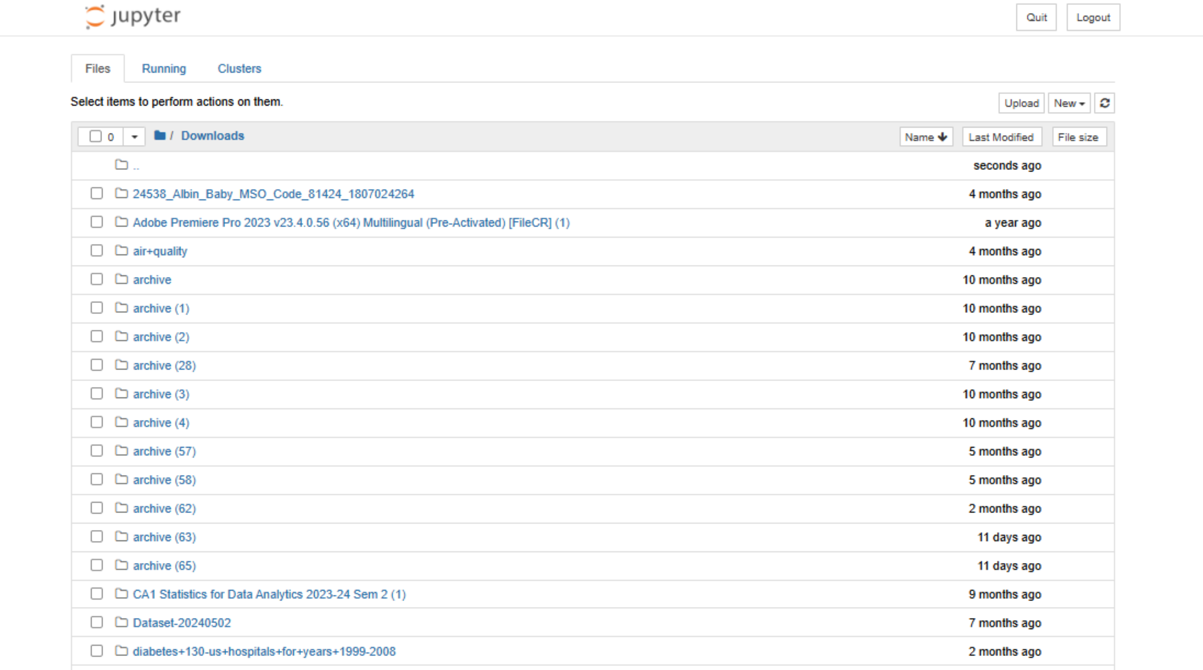
IBM watsonx is an enterprise-ready AI platform including a data store, model builder, and AI model management and monitoring.

OCI Data Science offers a machine learning platform to build, train, manage, and

Multidimensional data visualization across files. Explore relationships within and

Component based data mining Framework. Data visualization and data analysis for

3. After opening jupyter notebook click on the new notebook (python 3).

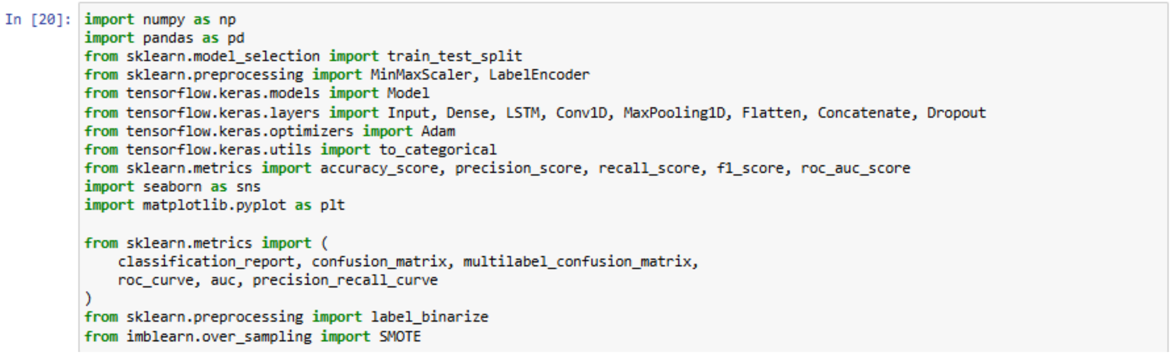


Select items to perform actions on them.

/ Downloads

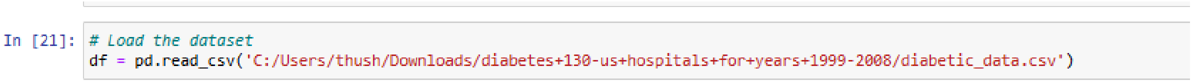
Name	Last Modified
□ ..	seconds ago
□ 24538_Albin_Baby_MSO_Code_81424_1807024264	4 months ago
□ Adobe Premiere Pro 2023 v23.4.0.56 (x64) Multilingual (Pre-Activated) [FileCR] (1)	a year ago
□ air+quality	4 months ago
□ archive	10 months ago
□ archive (1)	10 months ago
□ archive (2)	10 months ago
□ archive (28)	7 months ago
□ archive (3)	10 months ago
□ archive (4)	10 months ago
□ archive (57)	5 months ago
□ archive (58)	5 months ago
□ archive (62)	2 months ago
□ archive (63)	11 days ago
□ archive (65)	11 days ago
□ CA1 Statistics for Data Analytics 2023-24 Sem 2 (1)	9 months ago
□ Dataset-20240502	7 months ago
□ diabetes+130-us+hospitals+for+years+1999-2008	2 months ago

4. In notebook, Import all the required libraries.



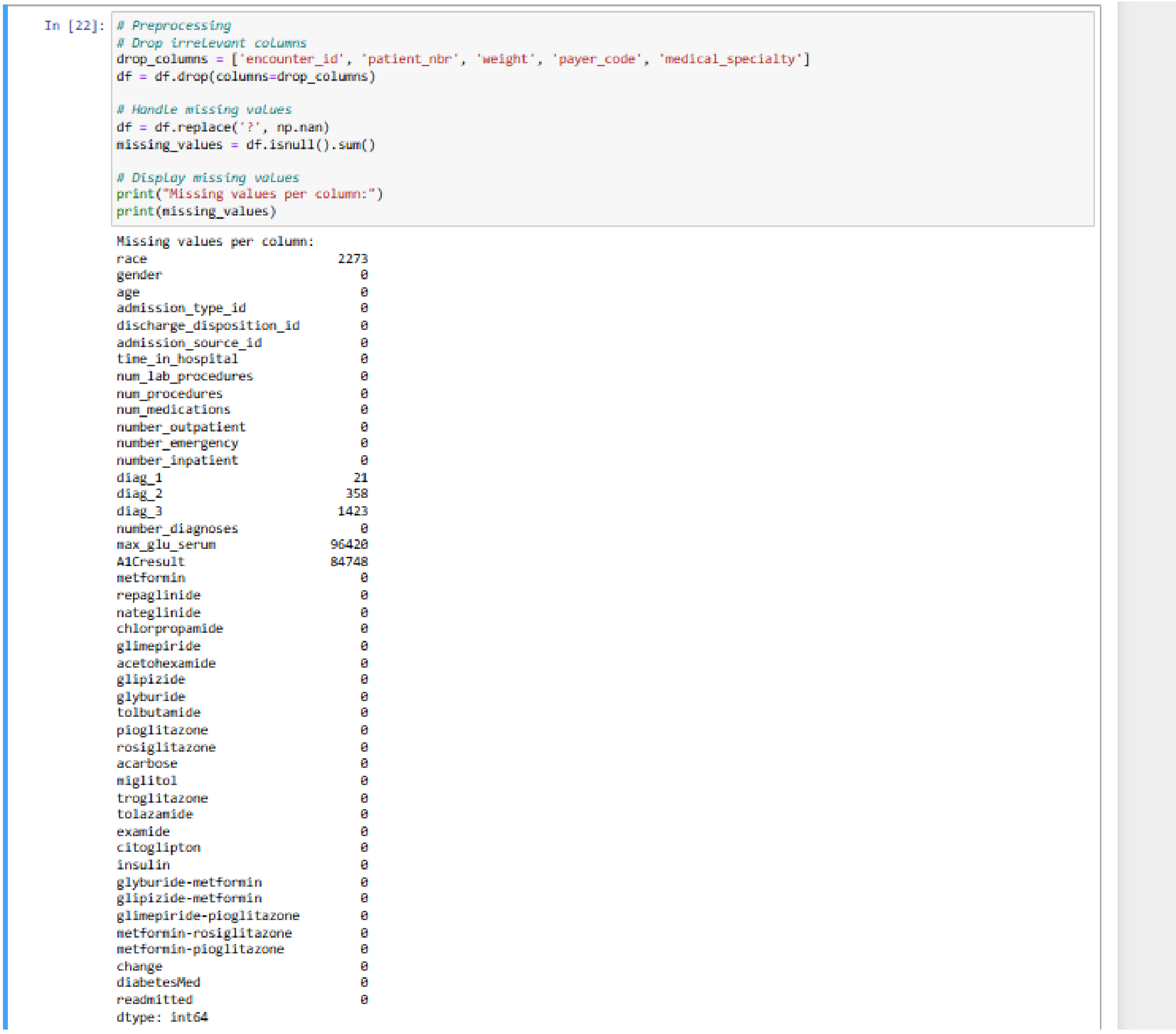
```
In [20]: import numpy as np
         import pandas as pd
         from sklearn.model_selection import train_test_split
         from sklearn.preprocessing import MinMaxScaler, LabelEncoder
         from tensorflow.keras.models import Model
         from tensorflow.keras.layers import Input, Dense, LSTM, Conv1D, MaxPooling1D, Flatten, Concatenate, Dropout
         from tensorflow.keras.optimizers import Adam
         from tensorflow.keras.utils import to_categorical
         from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score
         import seaborn as sns
         import matplotlib.pyplot as plt

         from sklearn.metrics import (
             classification_report, confusion_matrix, multilabel_confusion_matrix,
             roc_curve, auc, precision_recall_curve
         )
         from sklearn.preprocessing import label_binarize
         from imblearn.over_sampling import SMOTE
```

5. Import the Provided Dataset.



```
In [21]: # Load the dataset
         df = pd.read_csv('C:/Users/thush/Downloads/diabetes+130-us+hospitals+for+years+1999-2008/diabetic_data.csv')
```

6.Data Pre Processing Step will be performed using following Code.

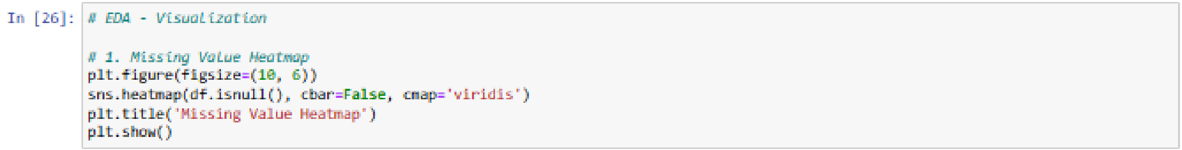


In [22]:

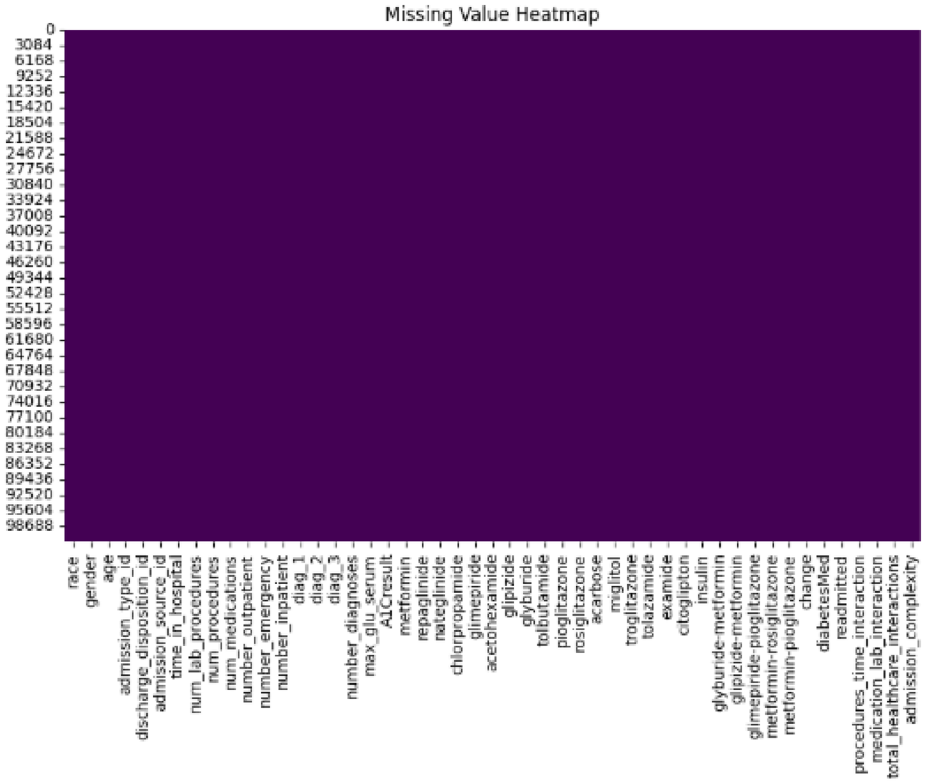
```
# Preprocessing
# Drop irrelevant columns
drop_columns = ['encounter_id', 'patient_nbr', 'weight', 'payer_code', 'medical_specialty']
df = df.drop(columns=drop_columns)

# Handle missing values
df = df.replace('?', np.nan)
missing_values = df.isnull().sum()

# Display missing values
print("Missing values per column:")
print(missing_values)
```

```
Missing values per column:
race                         2273
gender                          0
age                             0
admission_type_id               0
discharge_disposition_id        0
admission_source_id             0
time_in_hospital                0
num_lab_procedures              0
num_procedures                  0
num_medications                 0
number_outpatient               0
number_emergency                0
number_inpatient                0
diag_1                         21
diag_2                        358
diag_3                       1423
number_diagnoses                0
max_glu_serum               96420
A1Cresult                   84748
metformin                       0
repaglinide                     0
nateglinide                     0
chlorpropamide                  0
glimepiride                     0
acetohexamide                   0
glipizide                       0
glyburide                       0
tolbutamide                     0
pioglitazone                    0
rosiglitazone                   0
acarbose                        0
miglitol                        0
troglitazone                    0
tolazamide                      0
examide                         0
citoglipton                     0
insulin                         0
glyburide-metformin             0
glipizide-metformin             0
glimepiride-pioglitazone        0
metformin-rosiglitazone         0
metformin-pioglitazone          0
change                          0
diabetesMed                     0
readmitted                      0
dtype: int64
```

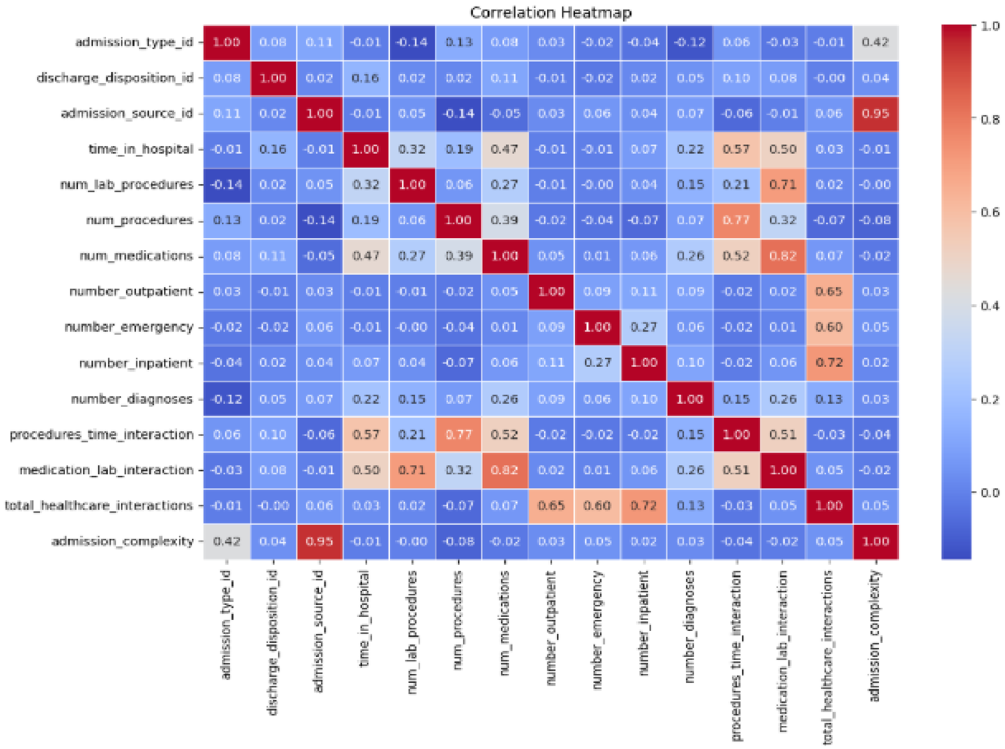
7.Exploratory Data Analysis has been Performed and Visualisation has been done using following Code



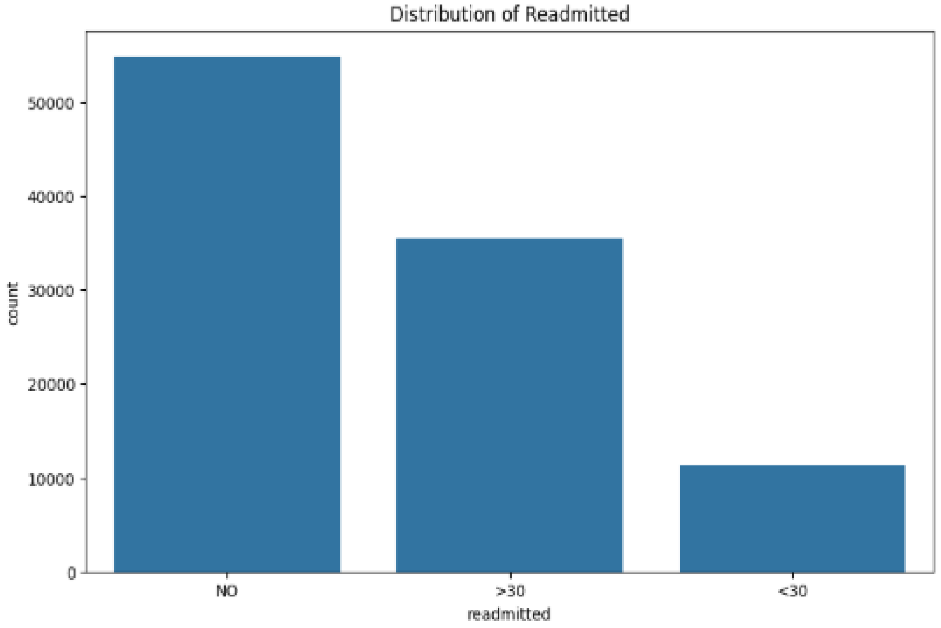
```
In [26]: # EDA - Visualization

         # 1. Missing Value Heatmap
         plt.figure(figsize=(10, 6))
         sns.heatmap(df.isnull(), cbar=False, cmap='viridis')
         plt.title('Missing Value Heatmap')
         plt.show()
```



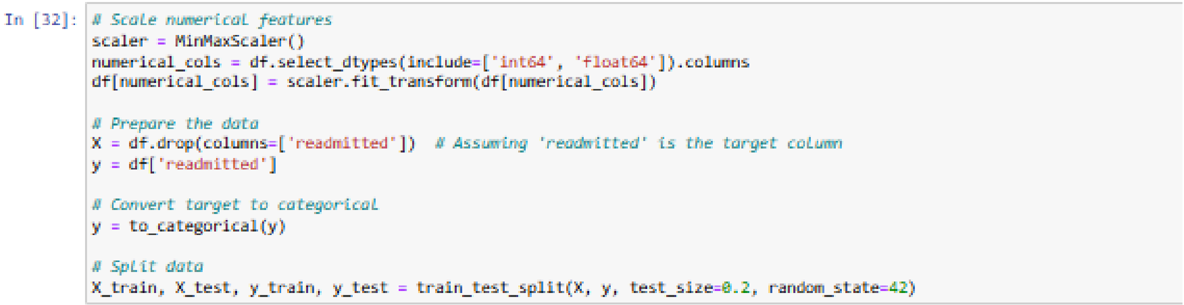
```
In [27]: # 2. Correlation Heatmap (for numerical features)
         plt.figure(figsize=(12, 8))
         correlation_matrix = df[numerical_cols].corr()
         sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
         plt.title('Correlation Heatmap')
         plt.show()
```



```
In [30]: # 5. Categorical Feature Distribution - Visualize 'readmitted' column
         plt.figure(figsize=(10, 6))
         sns.countplot(x='readmitted', data=df)
         plt.title('Distribution of Readmitted')
         plt.show()
```

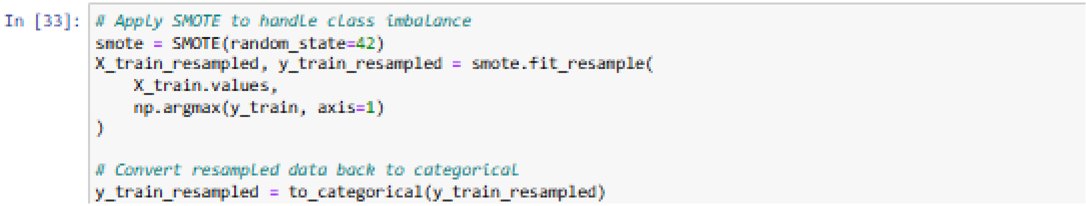


8. After Data Pre Processing the Data Splitting is Performed before Building a Model



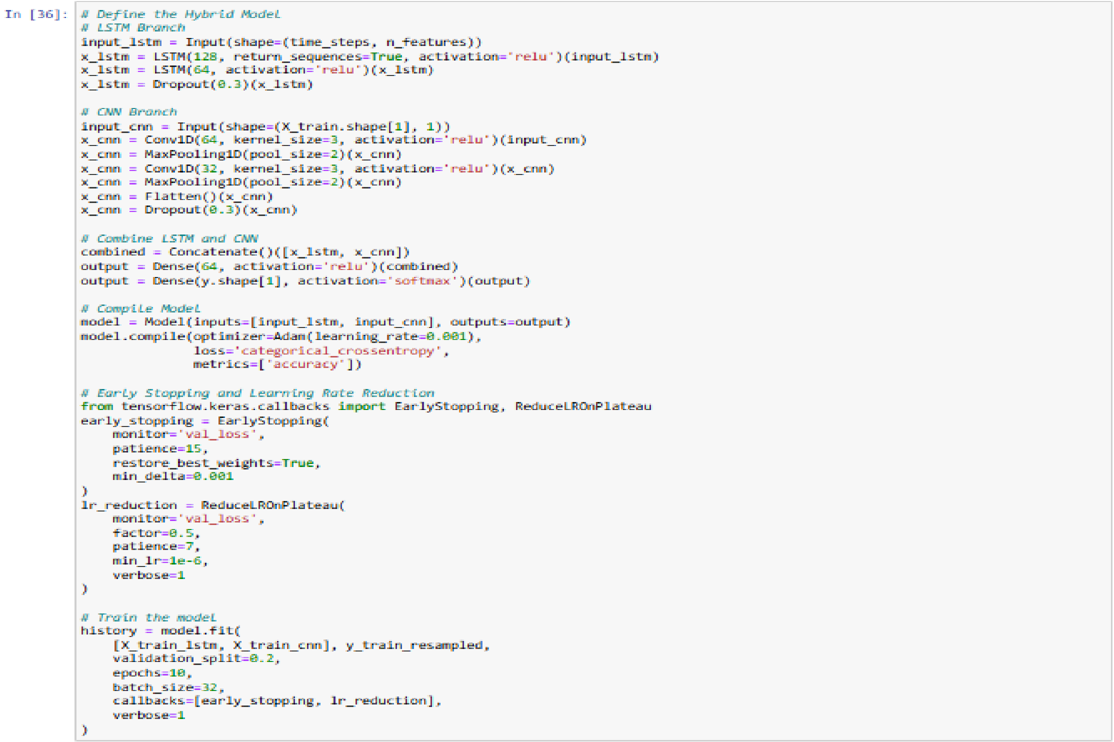
```
In [32]: # Scale numerical features
         scaler = MinMaxScaler()
         numerical_cols = df.select_dtypes(include=['int64', 'float64']).columns
         df[numerical_cols] = scaler.fit_transform(df[numerical_cols])

         # Prepare the data
         X = df.drop(columns=['readmitted'])  # Assuming 'readmitted' is the target column
         y = df['readmitted']

         # Convert target to categorical
         y = to_categorical(y)

         # Split data
         X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

9. Applying SMOTE To handle imbalance

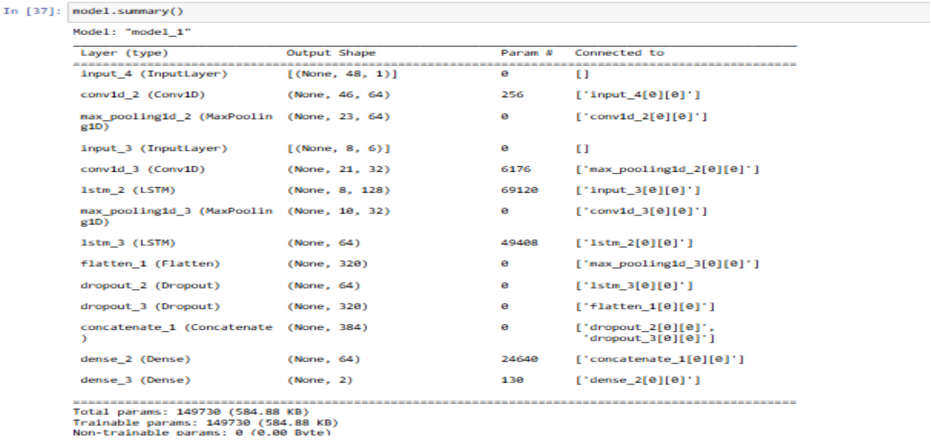


```
In [33]: # Apply SMOTE to handle class imbalance
         smote = SMOTE(random_state=42)
         X_train_resampled, y_train_resampled = smote.fit_resample(
             X_train.values,
             np.argmax(y_train, axis=1)
         )

         # Convert resampled data back to categorical
         y_train_resampled = to_categorical(y_train_resampled)
```

10. Building a Hybrid LSTM and CNN model.



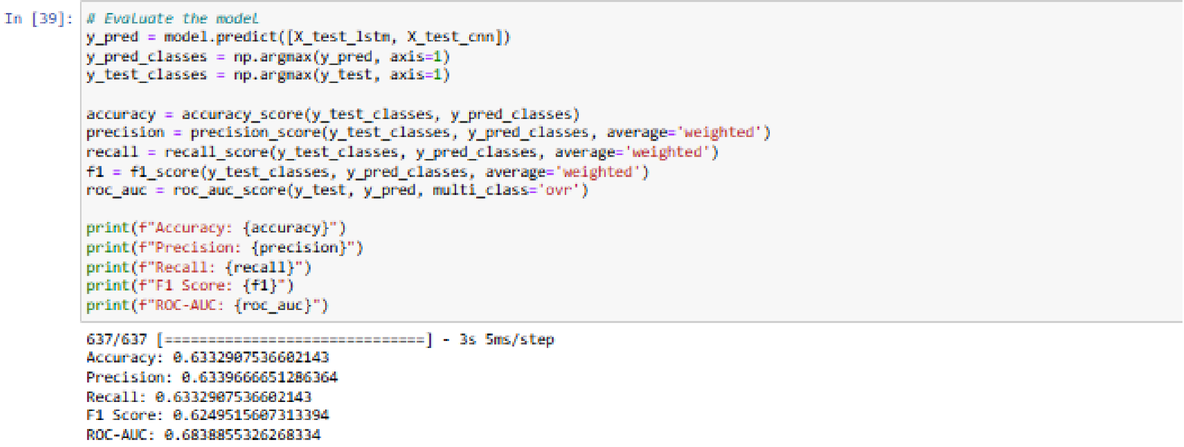
```
In [36]: # Define the Hybrid Model
         # LSTM Branch
         input_lstm = Input(shape=(time_steps, n_features))
         x_lstm = LSTM(128, return_sequences=True, activation='relu')(input_lstm)
         x_lstm = LSTM(64, activation='relu')(x_lstm)
         x_lstm = Dropout(0.3)(x_lstm)

         # CNN Branch
         input_cnn = Input(shape=(X_train.shape[1], 1))
         x_cnn = Conv1D(64, kernel_size=3, activation='relu')(input_cnn)
         x_cnn = MaxPooling1D(pool_size=2)(x_cnn)
         x_cnn = Conv1D(32, kernel_size=3, activation='relu')(x_cnn)
         x_cnn = MaxPooling1D(pool_size=2)(x_cnn)
         x_cnn = Flatten()(x_cnn)
         x_cnn = Dropout(0.3)(x_cnn)

         # Combine LSTM and CNN
         combined = Concatenate()([x_lstm, x_cnn])
         output = Dense(64, activation='relu')(combined)
         output = Dense(y.shape[1], activation='softmax')(output)

         # Compile Model
         model = Model(inputs=[input_lstm, input_cnn], outputs=output)
         model.compile(optimizer=Adam(learning_rate=0.001),
                       loss='categorical_crossentropy',
                       metrics=['accuracy'])

         # Early Stopping and Learning Rate Reduction
         from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau
         early_stopping = EarlyStopping(
             monitor='val_loss',
             patience=15,
             restore_best_weights=True,
             min_delta=0.001
         )
         lr_reduction = ReduceLROnPlateau(
             monitor='val_loss',
             factor=0.5,
             patience=7,
             min_lr=1e-6,
             verbose=1
         )

         # Train the model
         history = model.fit(
             [X_train_lstm, X_train_cnn], y_train_resampled,
             validation_split=0.2,
             epochs=10,
             batch_size=32,
             callbacks=[early_stopping, lr_reduction],
             verbose=1
         )
```



```
In [37]: model.summary()
```

```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                Output Shape                 Param #     Connected to
==================================================================================================
 input_4 (InputLayer)        [(None, 48, 1)]              0           []

 conv1d_2 (Conv1D)           (None, 46, 64)               256         ['input_4[0][0]']

 max_pooling1d_2 (MaxPoolin  (None, 23, 64)               0           ['conv1d_2[0][0]']
 g1D)

 input_3 (InputLayer)        [(None, 8, 6)]               0           []

 conv1d_3 (Conv1D)           (None, 21, 32)               6176        ['max_pooling1d_2[0][0]']

 lstm_2 (LSTM)               (None, 8, 128)               69120       ['input_3[0][0]']

 max_pooling1d_3 (MaxPoolin  (None, 10, 32)               0           ['conv1d_3[0][0]']
 g1D)

 lstm_3 (LSTM)               (None, 64)                   49408       ['lstm_2[0][0]']

 flatten_1 (Flatten)         (None, 320)                  0           ['max_pooling1d_3[0][0]']

 dropout_2 (Dropout)         (None, 64)                   0           ['lstm_3[0][0]']

 dropout_3 (Dropout)         (None, 320)                  0           ['flatten_1[0][0]']

 concatenate_1 (Concatenate  (None, 384)                  0           ['dropout_2[0][0]',
 )                                                                     'dropout_3[0][0]']

 dense_2 (Dense)             (None, 64)                   24640       ['concatenate_1[0][0]']

 dense_3 (Dense)             (None, 2)                    130         ['dense_2[0][0]']

==================================================================================================
Total params: 149730 (584.88 KB)
Trainable params: 149730 (584.88 KB)
Non-trainable params: 0 (0.00 Byte)
```

11. Model Evaluation.



```
In [39]: # Evaluate the model
y_pred = model.predict([X_test_lstm, X_test_cnn])
y_pred_classes = np.argmax(y_pred, axis=1)
y_test_classes = np.argmax(y_test, axis=1)

accuracy = accuracy_score(y_test_classes, y_pred_classes)
precision = precision_score(y_test_classes, y_pred_classes, average='weighted')
recall = recall_score(y_test_classes, y_pred_classes, average='weighted')
f1 = f1_score(y_test_classes, y_pred_classes, average='weighted')
roc_auc = roc_auc_score(y_test, y_pred, multi_class='ovr')

print(f"Accuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1 Score: {f1}")
print(f"ROC-AUC: {roc_auc}")
```

```
637/637 [==============================] - 3s 5ms/step
Accuracy: 0.6332907536602143
Precision: 0.6339666651286364
Recall: 0.6332907536602143
F1 Score: 0.6249515607313394
ROC-AUC: 0.6838855326268334
```

References

Anaconda. 2023. Anaconda | The World's Most Popular Data Science Platform. [online] Available at: https://www.anaconda.com/.

Numpy.org. 2023. NumPy. [online] Available at: https://numpy.org/.

TensorFlow. 2023. TensorFlow | An end-to-end open source machine learning platform. [online] Available at: https://www.tensorflow.org/.

Pandas. 2023. pandas - Python Data Analysis Library. [online] Available at: https://pandas.pydata.org/.