

Configuration Manual

MSc Research Project
Data Analytics

Zainab Makrani
Student ID: 22190082

School of Computing
National College of Ireland

Supervisor: Jorge Basilio

National College of Ireland
Project Submission Sheet
School of Computing



Student Name:	Zainab Makrani
Student ID:	22190082
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Jorge Basilio
Submission Due Date:	15/12/2023
Project Title:	Configuration Manual
Word Count:	XXX
Page Count:	9

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:	
Date:	13th December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission , to each project (including multiple copies).	<input type="checkbox"/>
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.	<input type="checkbox"/>

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

Zainab Makrani
22190082

Contents

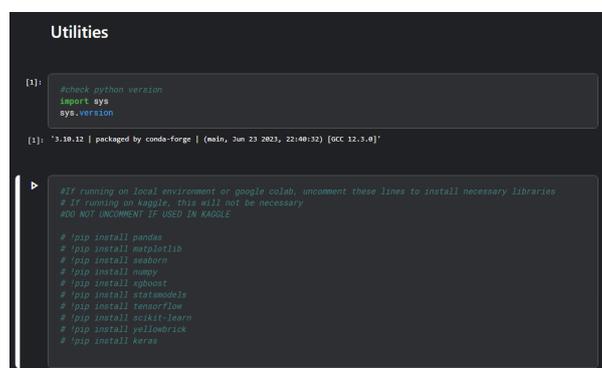
1	Introduction	2
2	Packages and Libraries	2
3	System and Software specification	3
3.1	Hardware specifications for local environment	3
4	Data Analysis	3
4.1	EDA	4
4.2	Pre-processing	5
4.3	Feature Engineering	6
4.4	Splitting data	7
5	Simple Machine learning models	7
6	Neural Networks	8
7	How to run the code	9
7.1	Kaggle Notebook	9
7.2	Jupyter Notebook	9

1 Introduction

This document discusses how the machine learning models are implemented for prediction of readmissions. The total code consists of 8 ML models which are differentiated between simple ML models and Neural networks. Each model is trained and used for prediction and then evaluated. The code is developed on kaggle IDE and is best to run it there, however, an alternative is to run it on local environment. Each of these ways is discussed in details in this manual.

2 Packages and Libraries

This is contained in the utilities section of the code which first checks the version of python implemented and necessary pip installments



```
Utilities

[1]: #check python version
import sys
sys.version

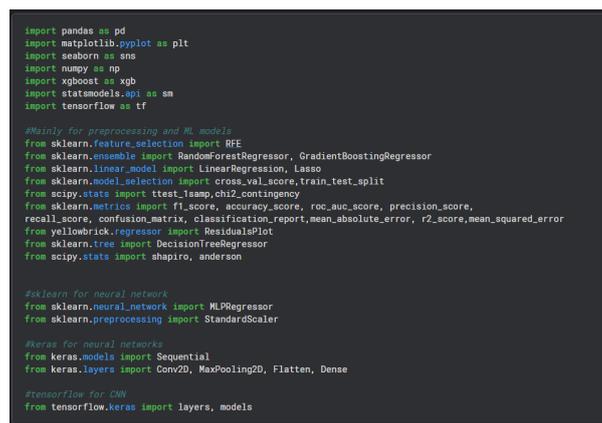
[1]: '3.10.12 | packaged by conda-forge | (main, Jun 23 2023, 22:40:32) [GCC 12.3.0]'

# If running on local environment or google colab, uncomment these lines to install necessary libraries
# If running on kaggle, this will not be necessary
# DO NOT UNCOMMENT IF USED IN KAGGLE

# !pip install pandas
# !pip install matplotlib
# !pip install seaborn
# !pip install numpy
# !pip install xgboost
# !pip install statsmodels
# !pip install tensorflow
# !pip install scikit-learn
# !pip install yellowbrick
# !pip install keras
```

Figure 1: Version and Pip install commands

The utilities also contain the libraries that are later implemented to create necessary machine learning methods.



```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import xgboost as xgb
import statsmodels.api as sm
import tensorflow as tf

#Mainly for preprocessing and ML models
from sklearn.feature_selection import RFECF
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.linear_model import LinearRegression, Lasso
from sklearn.model_selection import cross_val_score, train_test_split
from scipy.stats import ttest_ind, chi2_contingency
from sklearn.metrics import f1_score, accuracy_score, roc_auc_score, precision_score,
recall_score, confusion_matrix, classification_report, mean_absolute_error, r2_score, mean_squared_error
from yellowbrick.regressor import ResidualsPlot
from sklearn.tree import DecisionTreeRegressor
from scipy.stats import shapiro, anderson

#sklearn for neural network
from sklearn.neural_network import MLPRegressor
from sklearn.preprocessing import StandardScaler

#keras for neural networks
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

#tensorflow for CNN
from tensorflow.keras import layers, models
```

Figure 2: Python Libraries implemented

3 System and Software specification

To build this project on kaggle, u will need to create a kaggle account to run the kaggle notebook ¹. Based on their technical specifications, ², each notebook is allocated;

1. 12 hours of execution time for CPU and GPU ran sessions while providing 9hrs for TPU sessions.
2. 20GB automatically saving disk space
3. Scratchpad disk space which is not saved outside the current session.

To build this on a local environment, the prerequisites is that python 3 is installed and set up to the path ³.

3.1 Hardware specifications for local environment

These are the device specifications for the local environment in which the research was tested after compilation on kaggle

- Processor Intel(R) Core(TM) i5-8265U
- CPU @ 1.60GHz 1.80 GHz
- Installed RAM 8.00 GB (7.89 GB usable)
- Device ID 6DC9A036-AA3A-4B5D-ACCC-492EA2F9BF3E
- Product ID 00327-30666-87111-AAOEM
- System type 64-bit operating system, x64-based processor
- Pen and touch Touch support with 2 touch points

Windows specification include:

- Edition Windows 11 Home Single Language
- Version 22H2
- OS build 22621.2715
- Experience Windows Feature Experience Pack 1000.22677.1000.0

4 Data Analysis

In this section data is explored and prepared to be used for machine learning. In the EDA, data is loaded and thereafter graphs are made to understand the data. Not all codes are displayed here due to complexity of code but rather just snippets

¹How to run kaggle notebook. [<https://www.kaggle.com/docs/notebooks>]

²Technical specifications[<https://www.kaggle.com/docs/notebookstechnical-specifications>]

³How to install python 3 [<https://www.python.org/downloads/>]

4.1 EDA

```
# Display few rows of data
data.head()

[6]:
```

	race	gender	age	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	num_lab_procedures	num_procedures	num_medications
0	2	1	3	1	1	7	2	44	1	1
1	2	1	4	1	1	7	1	51	0	0
2	2	0	8	2	1	4	13	68	2	2
3	2	0	9	3	3	4	12	33	3	3
4	0	1	6	2	1	4	7	62	0	0

```
+ Code + Markdown

[7]:
```

```
# Display last few rows
data.tail()

[7]:
```

	race	gender	age	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	num_lab_procedures	num_procedures	num_medications
59552	2	1	8	1	1	7	1	1	1	0
59553	0	0	8	1	4	5	5	33	3	3
59554	2	1	7	1	1	7	1	53	0	0
59555	2	0	8	2	3	7	10	45	2	2
59556	2	1	7	1	1	7	6	13	3	3

```
[8]:
```

```
# Summary
data.describe()
```

Figure 3: Checking Loaded data in Kaggle

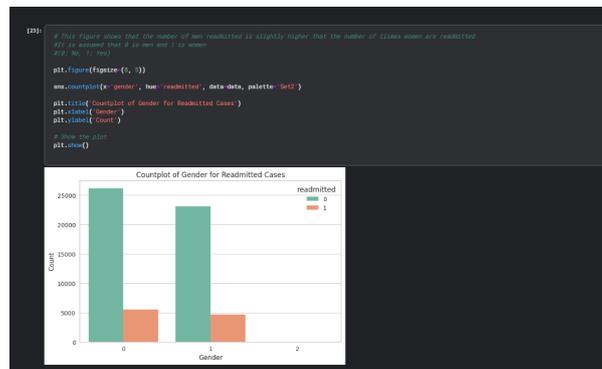


Figure 4: Counterplot to show gender relation to readmitted cases

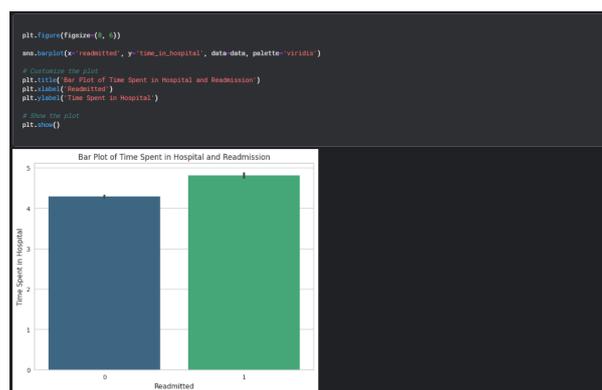


Figure 5: Barplot to show time spent in hospital vs readmission

4.2 Pre-processing

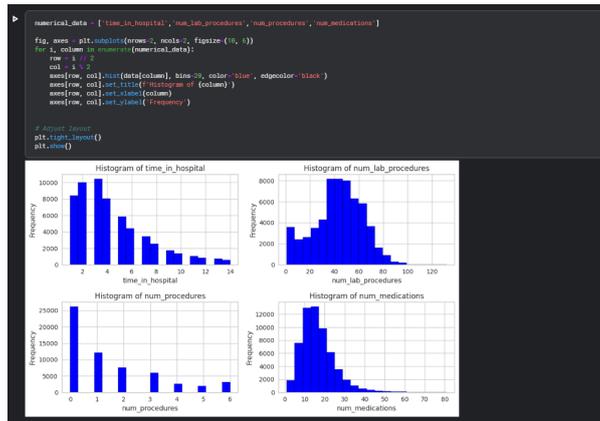


Figure 6: Histogram to check distribution

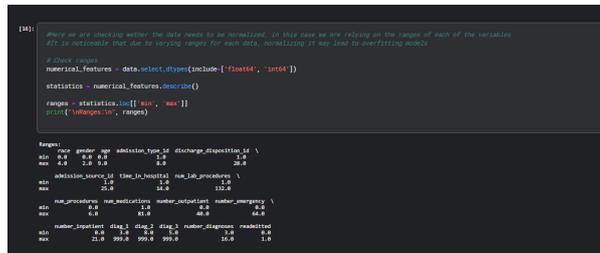


Figure 7: Check normalization of data

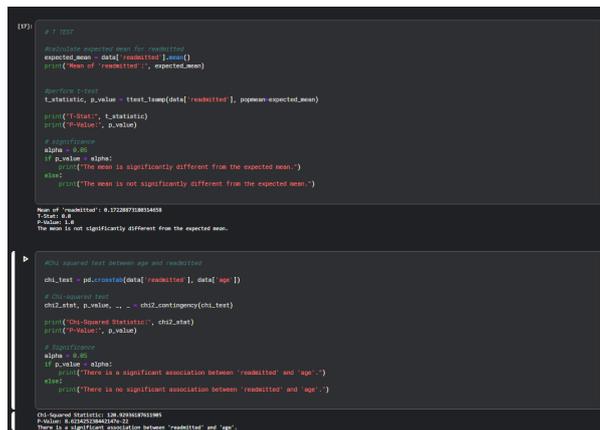


Figure 8: Enter Caption

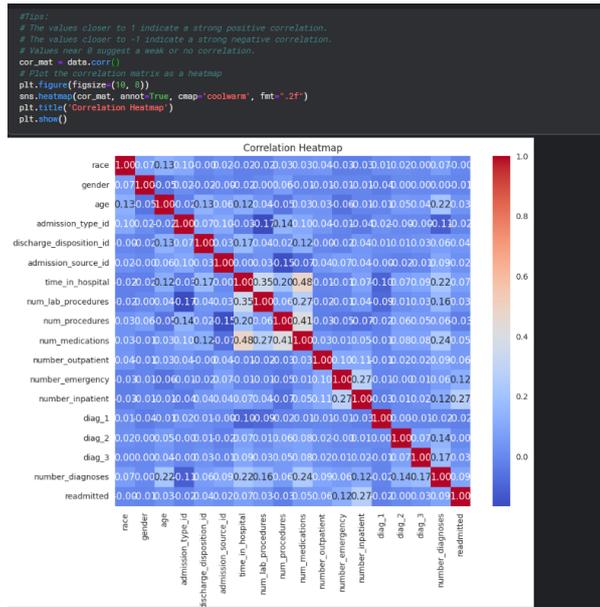


Figure 9: Cor relation matrix

4.3 Feature Engineering

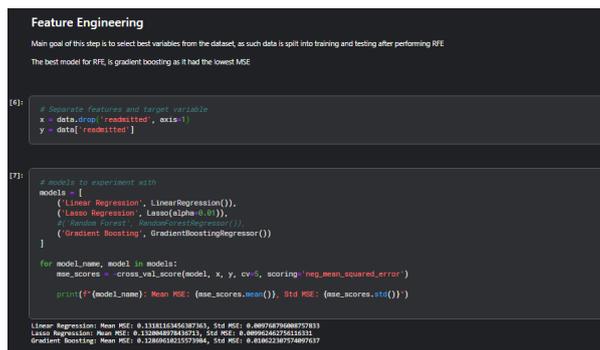


Figure 10: Experimenting with models for RFE

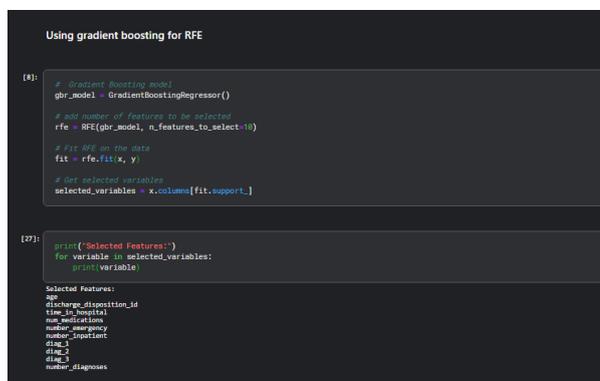


Figure 11: Using gradient boosting for RFE

4.4 Splitting data

```
Split data into training and testing

[17]: x = data_selected_variables
      y = data['number_of_tickets']
      x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)

[18]: x_train.head()

[19]:
   age  discharge_disposition_of_admission_to_hospital  num_medication  number_emergency  number_episodes  day_1  day_2  number_episodes
10077  4          1          2          10          0          0  1422  1423  21009          7
10078  7          1          5          17          0          1  1660  1661  14109          5
10084  7          4          3          21          0          1  1893  1820  42100          9
10085  7          4          4          14          0          1  1705  1706  15800          9
10086  7          2          1          10          0          0  7000  4110  12000          4

[19]: x_test.head()

[20]: (11912, 10)
```

Figure 12: Splitting data into training and testing

5 Simple Machine learning models

There are 5 simple models applied, however as a snippet to show the steps taken in the model creation the best performing from the 5 is displayed here which is random forest.

```
# Create and fit the Random Forest model
model_rf = RandomForestRegressor(n_estimators=100, random_state=42)
model_rf.fit(x_train, y_train)

# Make predictions on the testing data
rf_pred = model_rf.predict(x_test)
```

Figure 13: Creating model

```
threshold = 0.5 # Adjust the threshold as needed
rf_binary = np.where(y_test >= threshold, 1, 0)
rf_pred_binary = np.where(rf_pred >= threshold, 1, 0)

#classification evaluation
f1 = f1_score(rf_binary, rf_pred_binary)*100
accuracy = accuracy_score(rf_binary, rf_pred_binary)*100
auc = roc_auc_score(rf_binary, rf_pred_binary)
precision = precision_score(rf_binary, rf_pred_binary)*100

#Regression evaluation
mae = mean_absolute_error(y_test, rf_pred)
r2 = r2_score(y_test, rf_pred)

conf_matrix = confusion_matrix(rf_binary, rf_pred_binary)
class_report = classification_report(rf_binary, rf_pred_binary)

#print("Mean Squared Error:", mse)
print("Mean Absolute Error:", mae)
print("R-squared:", r2)
print("F1 Score:", f1)
print("Accuracy:", accuracy)
print("AUC:", auc)
print("Precision:", precision)
print("Confusion Matrix:", conf_matrix)
print("Class Report:", class_report)

Mean Absolute Error: 0.26678877186719115
R-squared: 0.04877163844595883
F1 Score: 22.18141755416582
Accuracy: 82.2112155809268
AUC: 0.5553842748579801
Precision: 47.63406940863091
Confusion Matrix: [[19491 332]
 [1787 3021]]
Class Report:
              precision    recall  f1-score   support
0             0.84       0.97       0.90       9823
1             0.48       0.14       0.22       2089

 accuracy         0.66         0.56         0.62       11912
  micro avg         0.66         0.56         0.62       11912
  weighted avg         0.78         0.82         0.78       11912
```

Figure 14: Evaluating Random Forest model

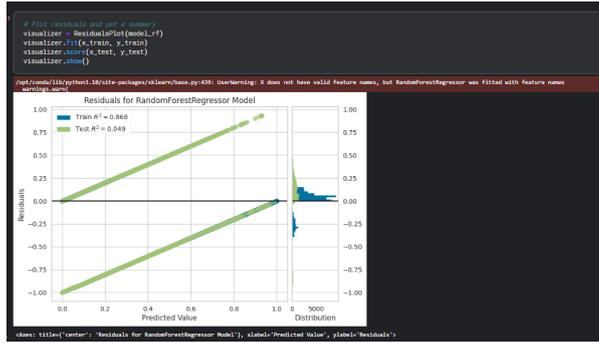


Figure 15: Visualizing the performance of random forest model predictions

6 Neural Networks

Just as done previously, here only the best performing model from the 3 neural networks is displayed which is dense neural network.

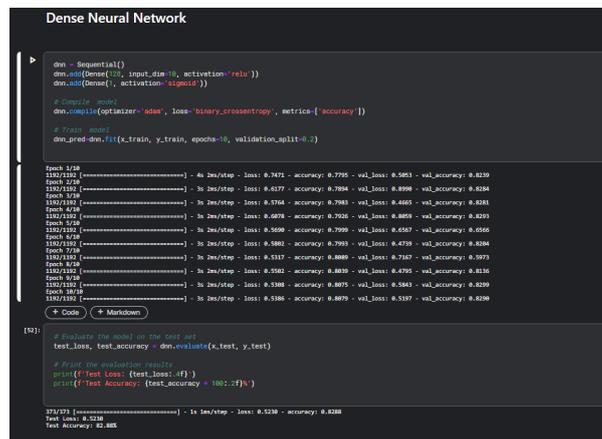


Figure 16: Creating and testing the dense neural network



Figure 17: Visualizing the performance of the dense neural network

7 How to run the code

The code itself contains comments of how to run and what to uncomment when running in different environments.

7.1 Kaggle Notebook

The pre requisites for this is that, you must create a kaggle account. Thereafter an easy way is to just follow the comments, but since the project is developed in kaggle notebook, you can opt to just hit the run all button after uploading it. It may take a while to run the whole code.

7.2 Jupyter Notebook

As for this, jupyter and python must be installed and correctly set up on your path. After this uncomment the pip install commands as shown below 18 to make sure that the libraries are imported correctly.

```
## If running in local environment or google colab, comment these lines to install necessary libraries
## If running on Kaggle, this is not necessary to uncomment
## Do not uncomment if used in Kaggle

# pip install pandas
# pip install matplotlib
# pip install seaborn
# pip install numpy
# pip install sklearn
# pip install keras
# pip install tensorflow
# pip install tensorflow-gpu
# pip install tensorflow-hub
# pip install keras-tqdm
# pip install keras
```

Figure 18: Pip install commands for local environment

After this follow the commented instructions and uncomment the first cell in the exploratory data 19. Before doing so make sure to download the dataset from kaggle ⁴ and that the jupyter notebook and dataset are in the same location.

```
## If running in local environment comment relevant lines
## Before this make sure to download the dataset from
# https://www.kaggle.com/datasets/omnamahshivai/dataset-hospital-readmissions-binary/

# # Replace your_dataset.csv with the dataset path
dataset = 'your_dataset.csv'

# Load the dataset into a Pandas DataFrame
data = pd.read_csv(dataset)
```

Figure 19: Uncomment the file path

Once that is done, comment this cell as it is only applicable if used on kaggle notebook to run the codes 20

```
## If using local environment comment these sections, this is only meant to be used for running code in Kaggle IDE

# Load the dataset
data = pd.read_csv('/kaggle/input/dataset-hospital-readmissions-binary/hospital-readmissions-orig.csv')
```

Figure 20: Comment for local Env

Once the relevant changes have been made you can opt to run all cells in the notebook.

References

⁴Dataset [<https://www.kaggle.com/datasets/omnamahshivai/dataset-hospital-readmissions-binary/>]