Configuration Manual for Research Project on Predicting Customer Lifetime Value: A Comprehensive Approach with Machine Learning and Deep Learning Models

MSc Research Project
Data Analytics

Arun Joy
Student ID: x23174994

School of Computing
National College of Ireland

Supervisor: Jaswinder Singh

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Arun Joy
Student ID:	x23174994
Programme:	Data Analytics **Year:** 2024
Module:	MSc Research Project
Lecturer:	Jaswinder Singh
Submission Due Date:	12/12/2024
Project Title:	Configuration Manual for Research Project on Predicting Customer Lifetime Value: A Comprehensive Approach with Machine Learning and Deep Learning Models
Word Count:	510 **Page Count:** 7

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: Arun Joy

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 for Research Project on Predicting Customer Lifetime Value: A Comprehensive Approach with Machine Learning and Deep Learning Models

Arun Joy
Student ID: x23174994

1 Introduction

The main objective of this project is to forecast Customer Lifetime Value (CLV) using both machine learning and deep learning models, including Decision Tree Regression, Random Forest Regression, Gradient Boosting Regression, GRU-LSTM and Bidirectional LSTM with Attention Mechanism. Finally, Bidirectional LSTM with the Attention Mechanism was seen to be most accurate than other models making it a strong tool in customer retention techniques.
This configuration manual guides through the different requirements and steps to be followed to successfully replicate and study the implementation of this project.

2 System Requirements

2.1 Hardware Specification

Operating System	Windows 11 Home Single Language
Processor	Processor 12th Gen Intel(R) Core(TM) i7-12700H, 2300 Mhz, 14 Core(s), 20 Logical Processor(s)
RAM	16.0 GB
System Type	64-bit operating system, x64-based processor

2.2 Software Specification

Programming Language	Python
Tools	Jupyter Notebook, Visual Studio Code

3 Software Installation

Link to download Jupyter Notebook: https://jupyter.org/install

Link to download Visual Studio Code: https://code.visualstudio.com/

4 Configuration Settings

Python Version – 3.11.5

4.1 Imported Libraries

Figure 1 shows the list of libraries imported to run the implementation of this project.

In [1]:
```
import sklearn
import numpy as np
import pandas as pd
import seaborn as sns
import plotly.express as px
from sklearn import metrics
from keras.layers import GRU
from keras.models import Model
import matplotlib.pyplot as plt
from tensorflow.keras import backend as K
from sklearn.tree import DecisionTreeRegressor
from tensorflow.keras.models import Sequential
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler, LabelEncoder
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor, ExtraTreesRegressor
from keras.layers import Input, Dense, LSTM, Dropout, Bidirectional, Multiply, Flatten, Permute, RepeatVector, Lambda
```

In [2]:
```
#disabling warnings
import warnings
warnings.filterwarnings('ignore')
```

Figure 1: Imported Libraries.

Python libraries and their versions can be seen in the following table.

scikit-learn	1.5.1
NumPy	1.26.4
Pandas	2.0.3
Seaborn	0.12.2
Plotly	5.9.0
Keras	3.4.1
Matplotlib	3.7.2
TensorFlow	2.17.0

Figure 2: Code Folder.

As seen in Figure 2, the code folder has the dataset, 'CLVDataset', taken from Kaggle and the implementation .ipynb file, 'Code'. The 'FlaskGui' folder has the application .py notebook which is the web application to predict CLV. The best performing model, BiLSTM with Attention Mechanism was stored as a JSON file, 'bilstmatt.json' and its weights were saved as an HDF5 file, 'bilstmatt_model.weights.h5' for easy loading and use in a production environment.

5 Data Preparation

The dataset, 'IBM Watson Marketing Customer Value Data' was taken from Kaggle and the link to that dataset is: https://www.kaggle.com/datasets/pankajjsh06/ibm-watson-marketing-customer-value-data/data

Figures 3 to 9 demonstrate different steps in data preparation.

Loading Data

```
In [3]: 1 data=pd.read_csv('CLVDataset.csv')
In [4]: 1 data.head() #first 5 rows of the dataset
```

Out[4]:

	Customer	State	Customer Lifetime Value	Response	Coverage	Education	Effective To Date	EmploymentStatus	Gender	Income	...	Months Since Policy Inception	Number of Open Complaints	Number of Policies	
0	BU79786	Washington	2763.519279	No	Basic	Bachelor	2/24/11	Employed	F	56274	...	5	0	1	Co
1	QZ44356	Arizona	6979.535903	No	Extended	Bachelor	1/31/11	Unemployed	F	0	...	42	0	8	P
2	AI49188	Nevada	12887.431650	No	Premium	Bachelor	2/19/11	Employed	F	48767	...	38	0	2	P
3	WW63253	California	7645.861827	No	Basic	Bachelor	1/20/11	Unemployed	M	0	...	65	0	7	Co
4	HB64268	Washington	2813.692575	No	Basic	Bachelor	2/3/11	Employed	M	43836	...	44	0	1	P

5 rows × 24 columns

Figure 3: Loading Dataset.

```
In [10]: 1 data=data.drop(['Customer', 'Effective To Date'], axis=1) #removing unnecessary columns
```

Figure 4: Removing Unwanted Columns.

```
In [13]: 1 data=data[data['Customer Lifetime Value'] < 15000] #removing outliers
```

Figure 5: Removing Outliers.

```
In [15]: 1 #rounding the values
         2 data['Customer Lifetime Value']=data['Customer Lifetime Value'].round(2)
         3 data['Total Claim Amount']=data['Total Claim Amount'].round(2)
```

Figure 6: Rounding Decimal Values in the Dataset.

```
In [31]: 1 categorical_list=categorical_data.columns.tolist() #converting categorical columns in the data to a list

In [32]: 1 print(categorical_list)

['State', 'Response', 'Coverage', 'Education', 'EmploymentStatus', 'Gender', 'Location Code', 'Marital Status', 'Policy Type',
'Policy', 'Renew Offer Type', 'Sales Channel', 'Vehicle Class', 'Vehicle Size']

In [33]: 1 le=LabelEncoder()
         2 data[categorical_list]=data[categorical_list].apply(le.fit_transform) #converting categorical values to numeric values
```

Figure 7: Converting Categorical to Numeric Values.

```
In [36]: 1 X=data.drop(['Customer Lifetime Value'], axis=1) #X has all the features except 'Customer Lifetime Value'
         2 y=data['Customer Lifetime Value'] #y has only 'Customer Lifetime Value'

In [37]: 1 #splitting both the dataset X and y into training and testing in the ratio 80:20
         2 X_train, X_test, y_train, y_test=train_test_split(X, y, test_size=0.2)
```

Figure 8: Splitting the Dataset.

```
In [44]:
#scaling to a range between 0 and 1

y_train_scaler=MinMaxScaler()
y_test_scaler=MinMaxScaler()

#fitting the scaler for training data and scaling it
y_train_scaler.fit(pd.DataFrame(y_train))
y_train=y_train_scaler.transform(pd.DataFrame(y_train))

#fitting the scaler for testing data and scaling it
y_test_scaler.fit(pd.DataFrame(y_test))
y_test=y_test_scaler.transform(pd.DataFrame(y_test))
```

Figure 9: Scaling Numeric Values.

6 Running the Experiment

The 'Code' file seen in Figure 2 can be run in the Jupyter Notebook by selecting 'Restart and Run All' in the Kernel section as seen in Figure 10.

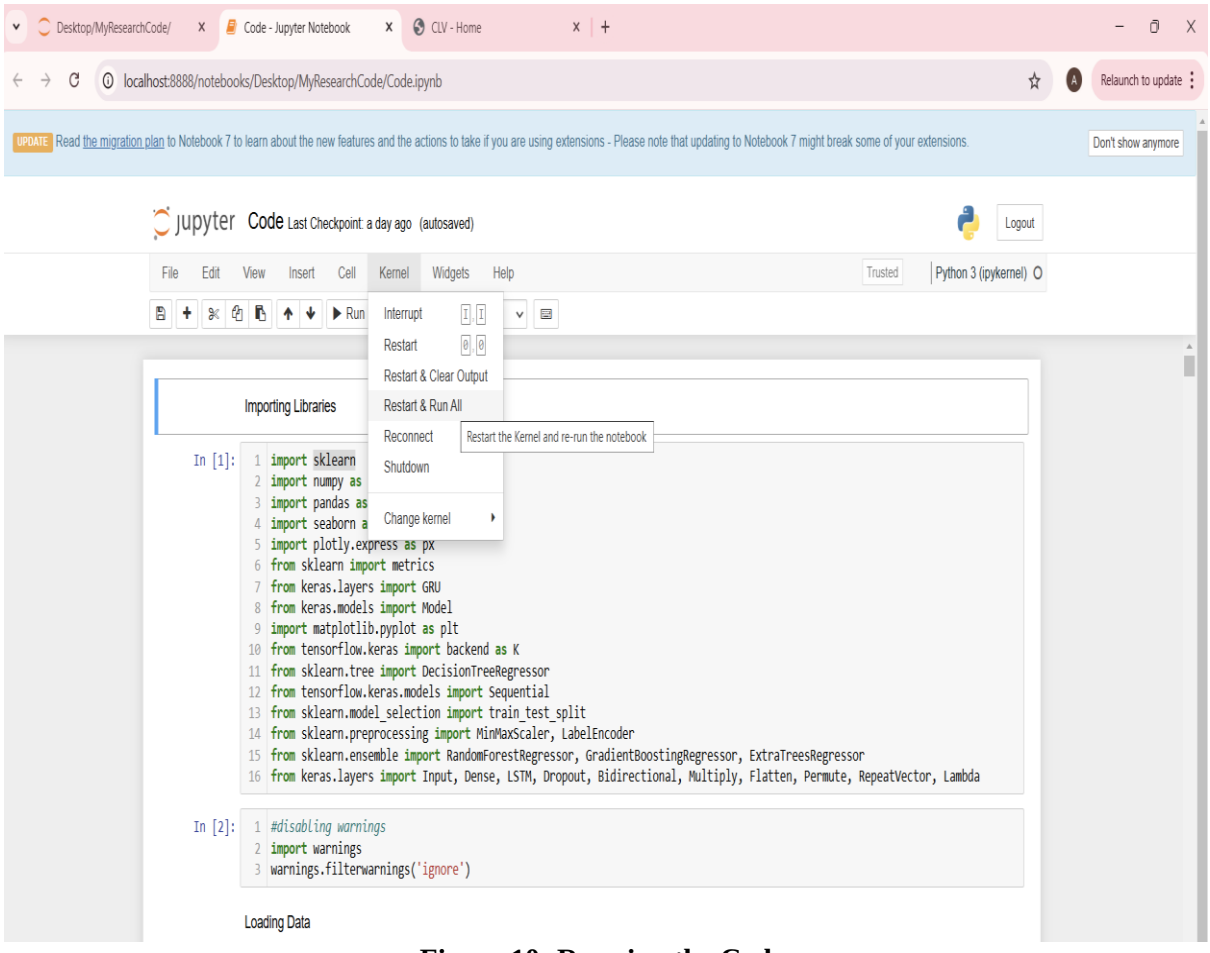

Figure 10: Running the Code.

To run the application, select the ‘FlaskGui’ folder in Visual Studio Code and then run the app.py file, by clicking the play button as seen in Figure 11. That will generate a http link to run in the local host, which can be followed to get the application to predict CLV as seen in Figure 12.

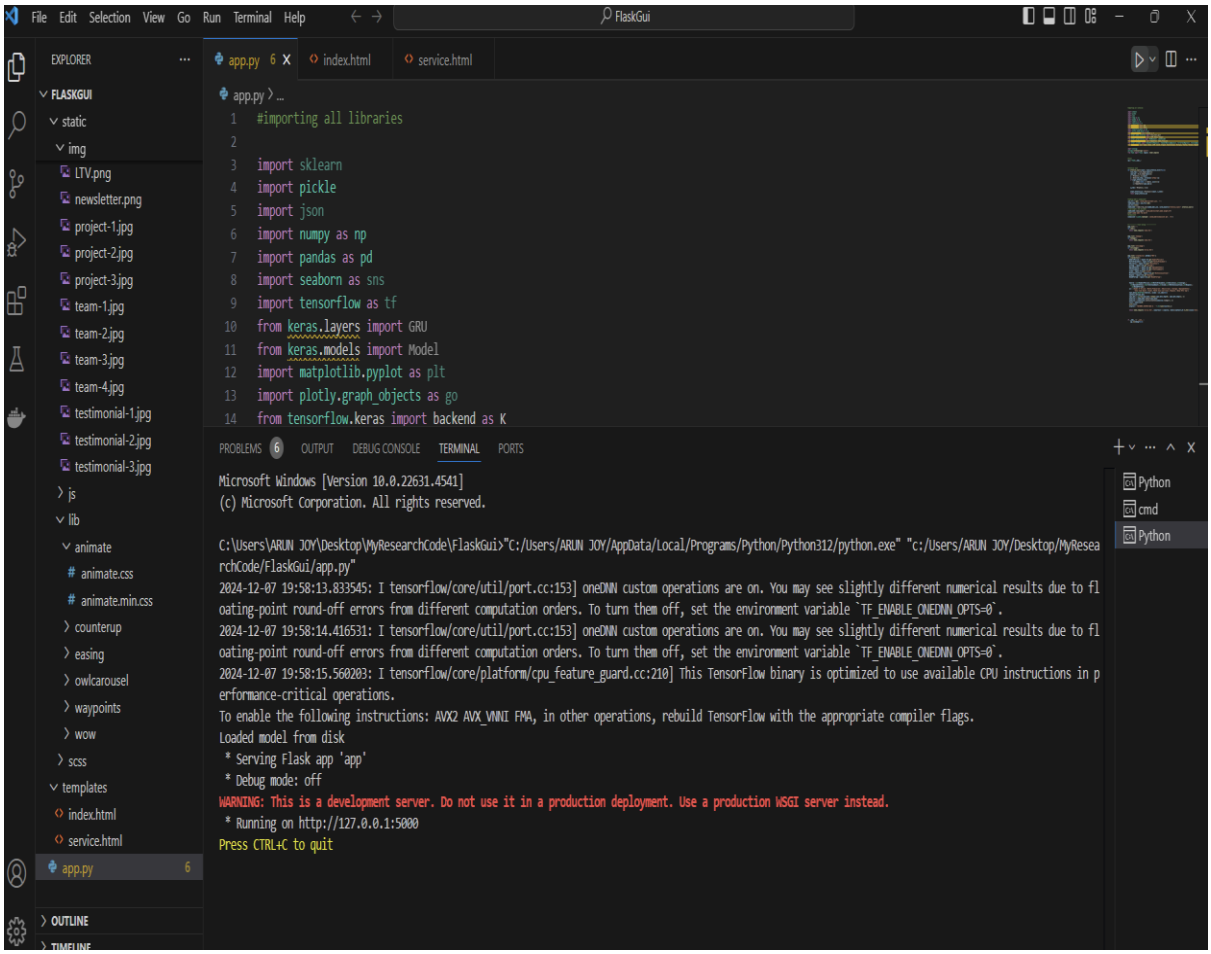

Figure 11: Running the application.

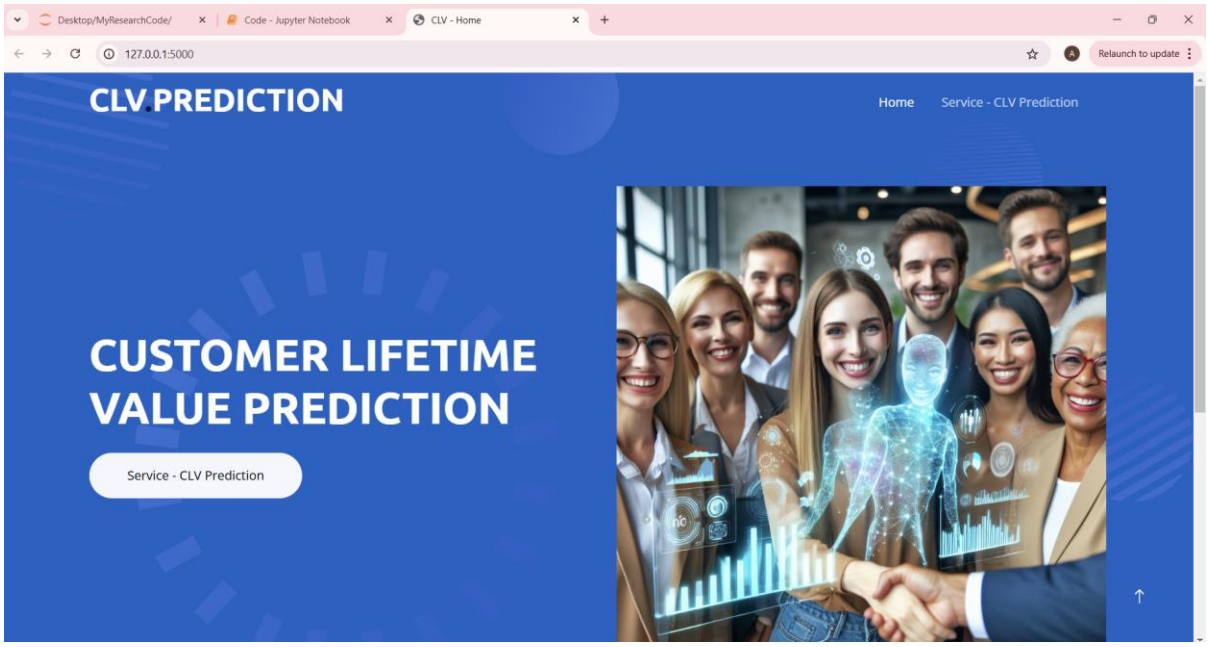

Figure 12: CLV Prediction Application.

The ‘Service-CLV Prediction’ has the page to predict CLV based on different attributes, and it can be seen in Figure 13.

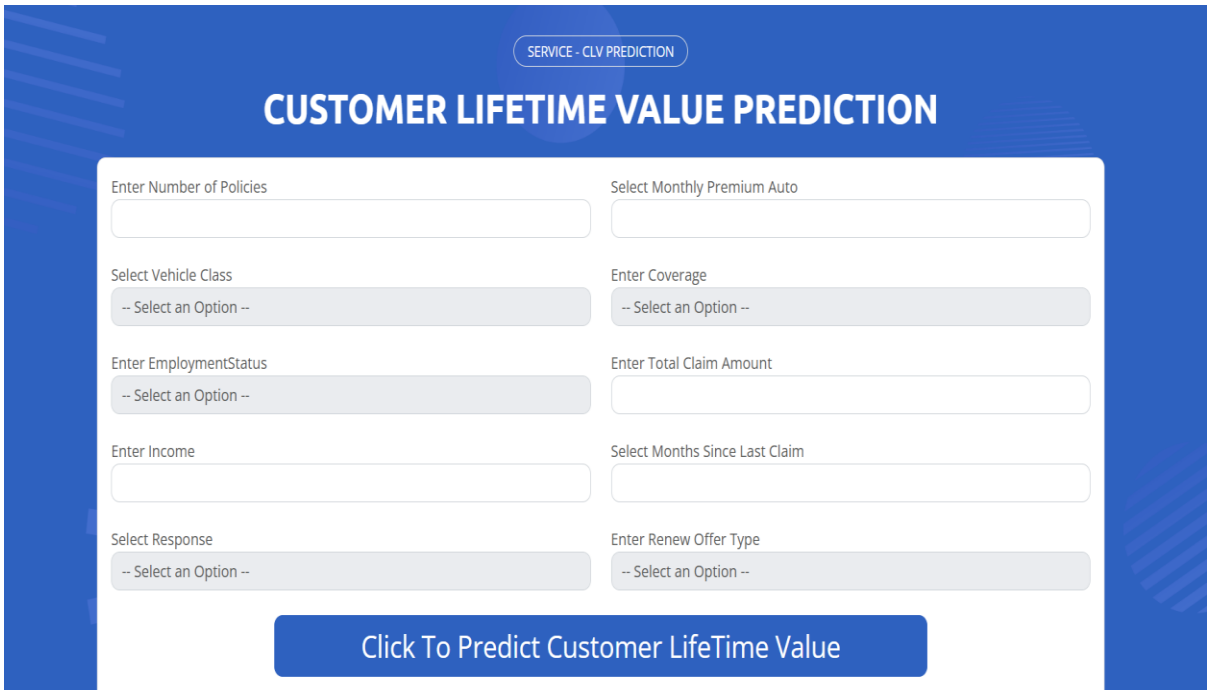

Figure 13: CLV Prediction Page.