National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Sneha Ramesh Dharne
Student ID: x23195703

School of Computing
National College of Ireland

Supervisor: Vikas Tomer

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……Ms. Sneha Ramesh Dharne…… ……

Student ID: ……x23195703……

Programme:……Data Analytics…… **Year:** ……2024……..

Module: ……MSc Research Project……..……

Lecturer: …… Vikas Tomer ……..……

Submission Due Date: ……12/12/2024……..……

Project Title: Big Data-Powered Temperature Prediction Using PySpark and Time Series Machine Learning Techniques

Word Count: ……759…… **Page Count:** ……8……..……

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: ……SNEHA RAMESH DHARNE……

Date: ……08/12/2024……

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

Sneha Ramesh Dharne
Student ID: x23195703

1 Introduction

This configuration manual provides detail on the steps taken for data acquisition, the system specifications required for the project, the libraries utilized, and a code walkthrough used in the implementation.
The manual is organized into four sections: Section 2 outlines the system requirements, and Section 3 provides information on data collection and pre-processing, while the final section describes model training and evaluation.

2 System Configuration

2.1 Hardware Configuration

The computer used for this study has a 64-bit operating system running Windows 11, 16GB of RAM, and an INTEL i5 12th Generation processor running at 1.30 GHz.

2.2 Software Configuration

The tools required for this project are:

- Microsoft Power BI
- Anaconda Navigator (by default installs the python)
- Jupyter Notebook

3 Environment Setup

Steps to follow:
Downloading and installing anaconda from https://www.anaconda.com/

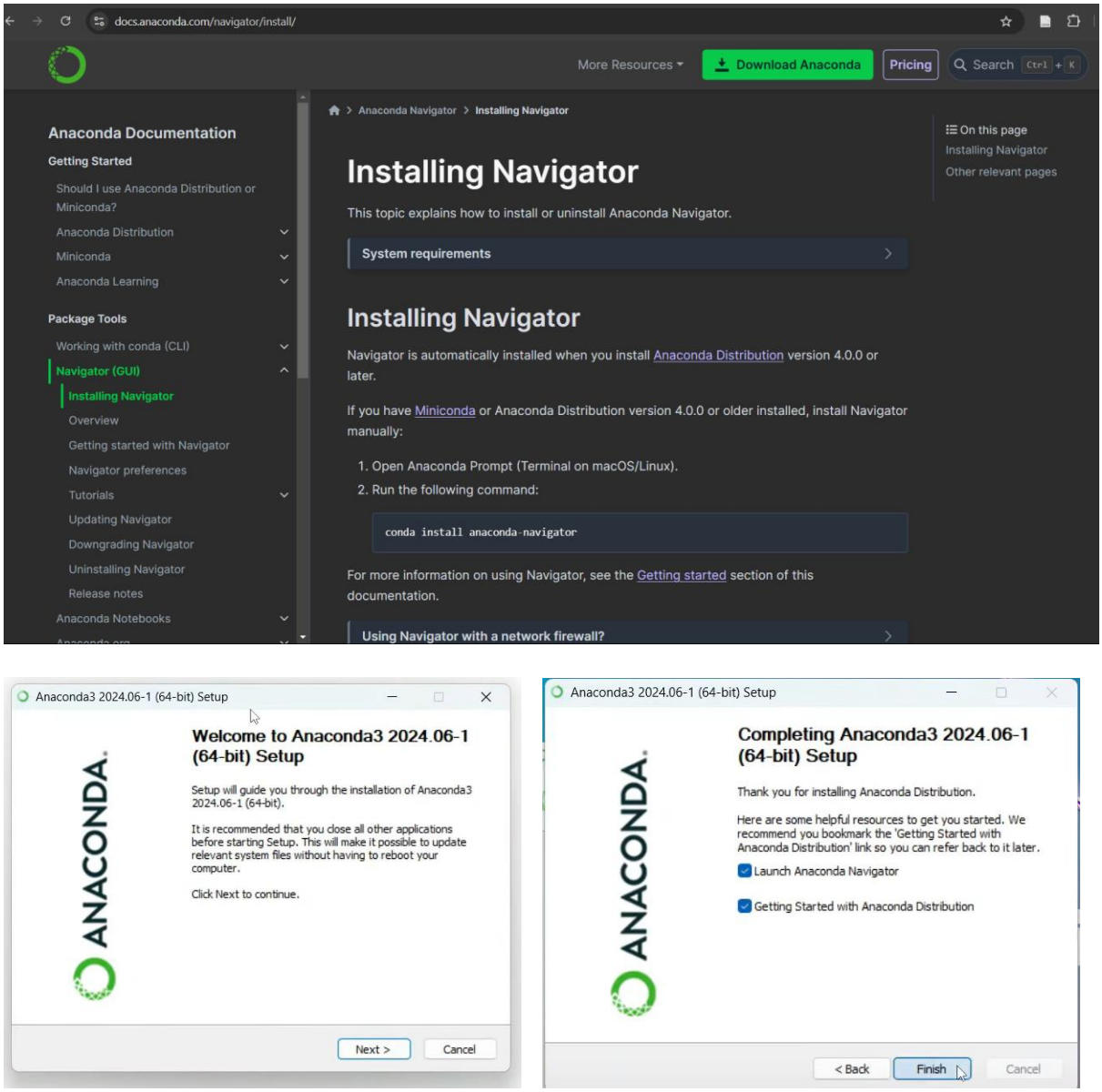

Fig. 1 Download and install anaconda Navigator

Download and install Microsoft Power Bi.

3.1 Data source

For this study, we use the Kaggle dataset "Climate Change: Earth Surface Temperature Data".
Dataset link- https://www.kaggle.com/datasets/berkeleyearth/climate-change-earth-surface-temperature-data

3.2 Installing Required Packages

Since PySpark is used in the processing, the project's packages are shown in Figure.
To install PySpark on the machine, follow these steps.

```
import pandas as pd
import os
import matplotlib.pyplot as plt

import findspark
from pyspark.sql import SparkSession
from pyspark.ml.classification import LogisticRegression

from pyspark.sql import functions as F
from pyspark.sql.window import Window
```

Fig. 2 Importing libraries

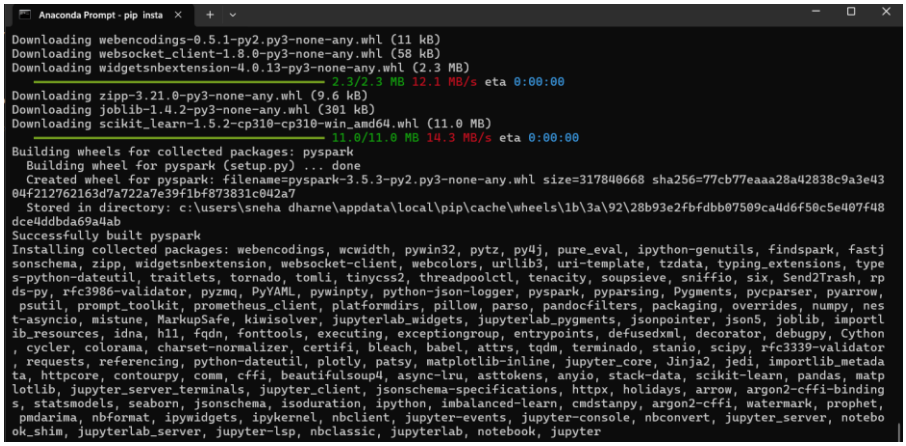

Fig. 3 Install Pyspark libraries

Download java 8 and Set up environment variable. check java version.

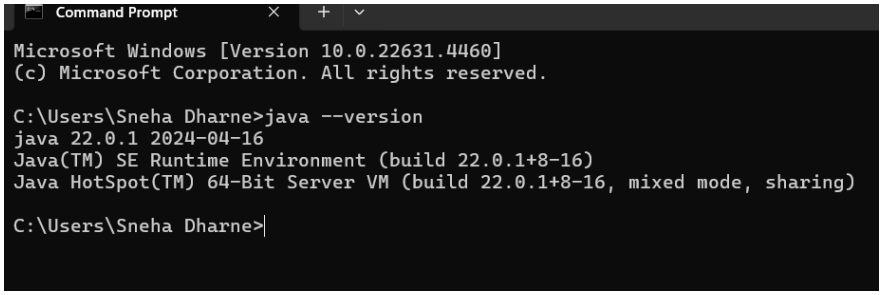

Fig.4 Download and install Java

The following packages were used :

NumPy
Sklearn
Pandas
Seaborn
Matplotlib

In [1]:
```
#Import Libraries
import pandas as pd
import os
import matplotlib.pyplot as plt

import findspark
from pyspark.sql import SparkSession
from pyspark.ml.classification import LogisticRegression
from lib_file import lib_path


from pyspark.sql import functions as F
from pyspark.sql.window import Window
```

In [2]:
```
import os
import matplotlib.pyplot as plt

import findspark
from pyspark.sql import SparkSession

import seaborn as sns
import pandas as pd
import numpy as np
```

Fig. 5 Install required packages and import library

Setup the base environment in anaconda navigator and open jupyter notebook. Run all the cells in notebook.

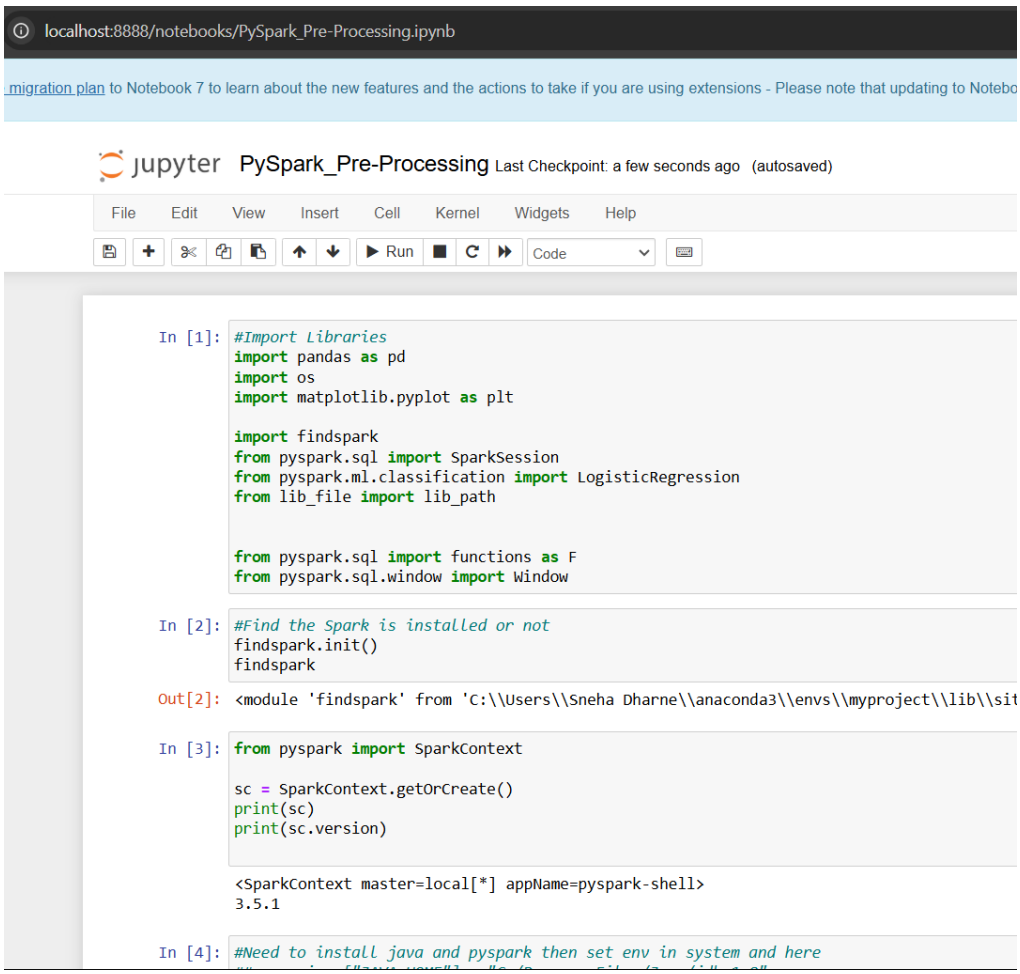

Fig. 6 Run cells in jupyter notebook

3.3 Pre-processing

- **Handling Missing Values:** Handling missing values by forward filling in the AverageTemperature column.

```
: data1.show(10)

+----------+-----------------+---------------------------+---
|        dt|AverageTemperature|AverageTemperatureUncertainty| Ci
+----------+-----------------+---------------------------+---
|1743-11-01|            6.068|          1.7369999999999999|Ârt
|1743-12-01|             NULL|                       NULL|Ârt
|1744-01-01|             NULL|                       NULL|Ârt
|1744-02-01|             NULL|                       NULL|Ârt
|1744-03-01|             NULL|                       NULL|Ârt
|1744-04-01|5.7879999999999985|          3.6239999999999997|Ârt
|1744-05-01|           10.644|          1.2830000000000001|Ârt
|1744-06-01|14.050999999999998|                      1.347|Ârt
|1744-07-01|           16.082|                      1.396|Ârt
|1744-08-01|             NULL|                       NULL|Ârt
+----------+-----------------+---------------------------+---
only showing top 10 rows
```

```
In [10]: # Sort the DataFrame by the 'dt' column
data1 = data1.orderBy('dt')

#Create a new column that represents the first day of the month for each row (monthly frequency)
data1 = data1.withColumn('month_start', F.trunc('dt', 'MM'))

#Group by the 'month_start' and calculate the average temperature for each month
data_monthly = data1.groupBy('month_start').agg(
    F.avg('AverageTemperature').alias('AverageTemperature')
)


window_spec = Window.orderBy('month_start')

data_monthly = data_monthly.withColumn(
    'AverageTemperature',
    F.last('AverageTemperature', True).over(window_spec)  # forward fill
)

# Show the result
data_monthly.show(20)
```

```
+-----------+------------------+
|month_start| AverageTemperature|
+-----------+------------------+
| 1743-11-01|  4.882423512747881|
| 1743-12-01|  4.882423512747881|
| 1744-01-01|  4.882423512747881|
| 1744-02-01|  4.882423512747881|
| 1744-03-01|  4.882423512747881|
| 1744-04-01|  9.737433427762947|
```

Fig. 7 Handle null values

- **Feature selection:** predictive variables are retrieved with the target variable, then the test train is split and fed into the models.

ARIMA Result

```
In [16]: from sklearn.metrics import mean_squared_error, r2_score

data_pandas = data_monthly.toPandas()

# keep only the numeric column for modeling
train_size = int(len(data_pandas) * 0.8)

train_data = data_pandas[:train_size]['AverageTemperature']
test_data = data_pandas[train_size:]['AverageTemperature']

# Fit the auto_arima model on the training data
auto_arima_model.fit(train_data)

# Forecast the test data
forecast = auto_arima_model.predict(n_periods=len(test_data))

# Calculate evaluation metrics
mse = mean_squared_error(test_data, forecast)
rmse = np.sqrt(mse)
r2 = r2_score(test_data, forecast)

# Display results
print(f"Mean Squared Error (MSE): {mse}")
print(f"Root Mean Squared Error (RMSE): {rmse}")
print(f"R-squared (R²): {r2}")
```

```
Mean Squared Error (MSE): 1.0359227561331057
Root Mean Squared Error (RMSE): 1.0178029063296614
R-squared (R²): 0.9462902030191617
```

Fig. 8 Feature selection

3.4 Power BI Visualizations

Power BI dashboards and charts were utilized for visualization.

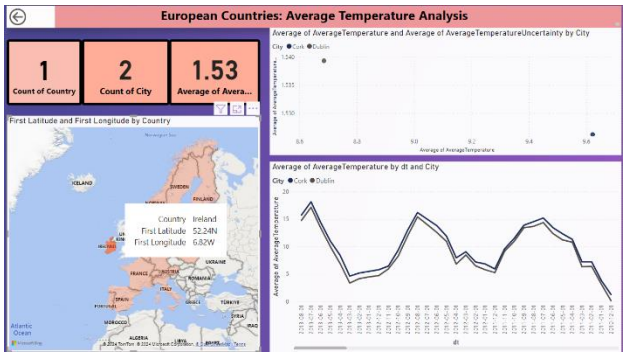

Fig.9 Visualization

4 Model Training and Evaluation

4.1 Implementation:

implementing the model to perform time series forecasting global temperatures.

It exploits seasonality and trends by using Auto ARIMA and SARIMA models. In order to forecast future values, it is applied to the temperature data**.**

A Seasonal ARIMA (SARIMA) model is trained to account for seasonal patterns (12-month seasonality).

4.2 Forecasting

we additionally use regression-based models like XGBoost and linear regression.

Data Splitting and Feature Engineering for XGBoost: Eighty percent of the dataset is used for training, while the remaining twenty percent is used for testing.

4.3 Evaluation and Plotting :

Evaluate each of the model's performance, i.e. ARIMA, SARIMA, XGBoost, and Linear Regression, using MSE, RMSE, and R^2 metrics.

References

Hewage, P., Trovati, M., Pereira, E. and Behera, A., 2021. Deep learning-based effective fine-grained weather forecasting model. *Pattern Analysis and Applications*, *24*(1), pp.343-366.

Abrahamsen, E.B., Brastein, O.M. and Lie, B., 2018. Machine learning in python for weather forecast based on freely available weather data.

Bahari, M. and Hamid, N.Z.A., 2019, June. Analysis and prediction of temperature time series using chaotic approach. In IOP Conference Series: Earth and Environmental Science (Vol. 286, No. 1, p. 012027). IOP Publ