National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Benjamin Kelani
Student ID: 21226181

School of Computing
National College of Ireland

Supervisor: Prof. Cristina Hava Muntean

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Benjamin Kelani ……

Student ID: 21226181……..……

Programme: Data Analytics…… **Year:** 2023……..

Module: M.Sc Research Project……..……

Lecturer: ……..……

Submission Due Date: ……..……

Project Title: Earthquake Magnitude Modelling using Machine Learning technology

Word Count: 792…… **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: Benjamin Kelani……

Date: 14th December 2023……

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

Benjamin Kelani
21226181

1 Introduction

This configuration manual includes the hardware and software configurations, libraries and important snippets of codes utilized during the implementation process. The aim of the documentation is to facilitate the reciprocation of this research project which is 'Earthquake Magnitude Modelling Using Machine Learning Technology'.

2 System Requirements

2.1 Hardware Requirements

The system hardware requirements is shown in the Table 1 below.

Table 1: Hardware requirements

Processor	8th Gen intel® Core™ i3-8130U @2.20GHz 2.21 GHz
RAM	8.0 GB (7.87 GB usable)

2.2 Software Requirements

The python programming language is used in the Jupyter notebook and Tableau is also utilized for data visualizations. Software and versions used are shown in Table 2 below.

Table 2: Software and Versions

Software Type	Software Name	Version
IDE	Jupyter notebook	6.4.12
Programming Language	Python	3.9.9
Data Visualization Software	Tableau	2023.3
Productivty and task completion	MS Office	2018

2.2.1 Libraries and Packages

The table 3 below shows all the libraries and versions that were installed for research study.

Table 3: Libraries and versions

Library	Description	Version
Pandas	Data manipulation and analysis	2.1.4
Matplotlib	Data visualisation	2.0
Seaborns	Data Visualization	12.0b3
Warnings	Customizing error display	

Scikit-learn	Machine learning tasks	1.3

Table 4 contains the packages used for this research project.

Packages	Use
Train_test_split	To split dataset into trainset and test set
LinearRegression	To build a linear regression model
R2_score	To utilizer R-Squared as an evaluation algorithm
Mean_squared_error	To utilize mean squared error as an evaluation algorithm
RandomForestRegressor	To build a random forest model
SVR	To build a support vector machine model
Accuracy_score	To utilize accuracy as an evaluation algorithm

3 Research Implementation

3.1 Data Extraction

The earthquake dataset are obtained from NCEDC Change the header and label to something appropriate. The open-source data does not need a user account to log in. Figure shows the landing page of the NCEDC site.

Figure 1: The landing page to get the earthquake data.

To access the earthquake dataset, the query depicted in figure 2 are run.

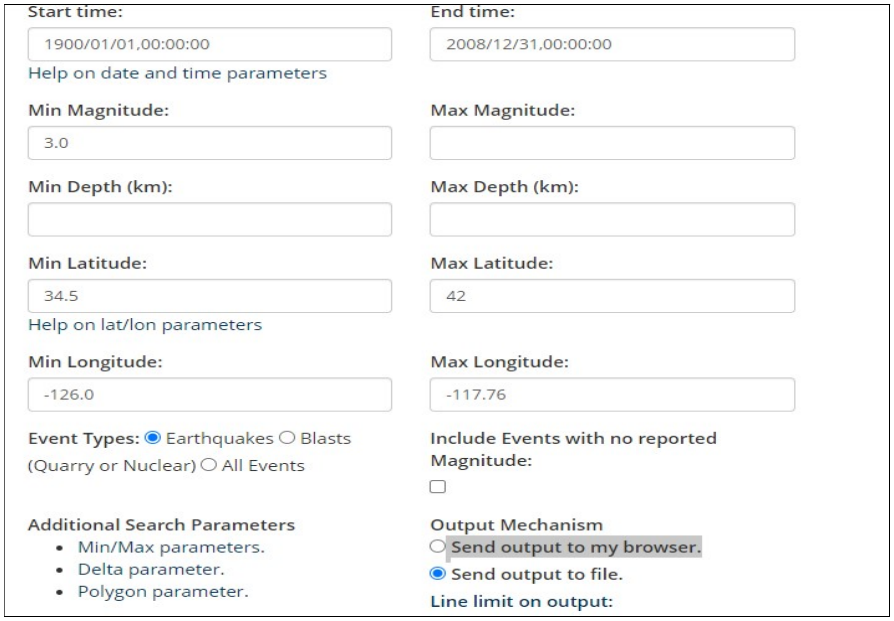

Figure 2: Search criteria for the earthquake dataset

To obtain the data, the file be downloaded as shown in figure 3.

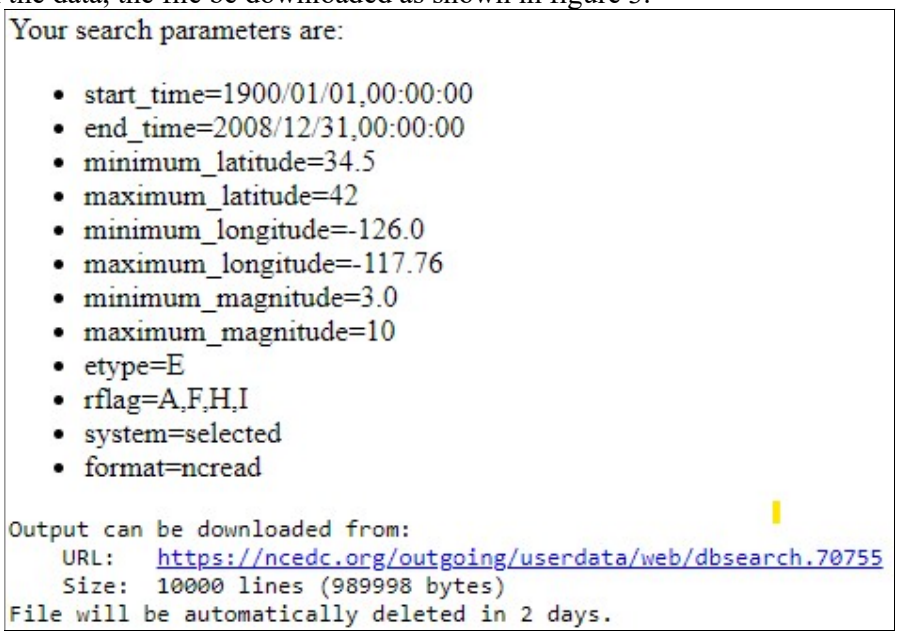

Figure 3: Page showing data file download

Sequel to downloading the file, it can be loaded and viewed in jupyter notebook environment as show in figure 4.

```
# Path to the CSV file
file_path = r'C:\Users\Kelani\Desktop\Project Data\Earthquake_Data.csv'

df = pd.read_csv(file_path, delimiter=r'\s+')

display(df)
```

	Date(YYYY/MM/DD)	Time	Latitude	Longitude	Depth	Mag	Magt	Nst	Gap	Clo	RM
0	1966/07/01	09:41:21.82	35.9463	-120.4700	12.26	3.20	Mx	7	171	20	0.0
1	1966/07/02	12:08:34.25	35.7867	-120.3265	8.99	3.70	Mx	8	86	3	0.0
2	1966/07/02	12:16:14.95	35.7928	-120.3353	9.88	3.40	Mx	8	89	2	0.0
3	1966/07/02	12:25:06.12	35.7970	-120.3282	9.09	3.10	Mx	8	101	3	0.0
4	1966/07/05	18:54:54.36	35.9223	-120.4585	7.86	3.10	Mx	9	161	14	0.0
...	...	...	...	...	...	...	...	...	...	...	.
18025	2007/12/19	12:14:09.62	34.1438	-116.9822	7.03	4.06	ML	10	73	14	0.0
18026	2007/12/21	12:14:56.45	37.3078	-121.6735	8.47	3.08	ML	114	45	5	0.1
18027	2007/12/23	21:43:43.54	37.2127	-117.8230	10.00	3.54	ML	45	176	40	0.0
18028	2007/12/28	01:59:42.40	36.5292	-121.1133	5.99	3.04	ML	70	45	4	0.0
18029	2007/12/28	23:20:28.12	38.7710	-122.7370	2.34	3.40	Mw	49	37	1	0.0

Figure 4 : Code snippet to view data information.

3.1.2 Missing Value Check

The data is checked for the presence of any missing value as shown in figure 5. There are no missing values identified.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 18030 entries, 1966-07-01 09:41:21.820000 to 2007-12-28 23:20:2
8.120000
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Latitude(deg)    18030 non-null  float64
 1   Longitude(deg)   18030 non-null  float64
 2   Depth(km)        18030 non-null  float64
 3   Magnitude(ergs)  18030 non-null  float64
 4   Magnitude_type   18030 non-null  object
 5   No_of_Stations   18030 non-null  int64
 6   Gap              18030 non-null  int64
 7   Close            18030 non-null  int64
 8   RMS              18030 non-null  float64
 9   SRC              18030 non-null  object
 10  EventID          18030 non-null  int64
dtypes: float64(5), int64(4), object(2)
memory usage: 1.7+ MB
```

Figure 5: Missing value check

3.2 Data Transformation

The earthquake data gets pre-processed by changing the column names and saving the pre-processed data in a .xslx format as shown in figure 6 and figure 7.

```
new_column_names = ["Date(YYYY/MM/DD)",  "Time(UTC)", "Latitude(deg)", "Longitude(deg)", "Depth(km)", "Magnitude(ergs)",
                    "Magnitude_type", "No_of_Stations", "Gap", "Close", "RMS", "SRC", "EventID"]

df.columns = new_column_names
ts = pd.to_datetime(df["Date(YYYY/MM/DD)"] + " " + df["Time(UTC)"])
df = df.drop(["Date(YYYY/MM/DD)", "Time(UTC)"], axis=1)
df.index = ts
display(df)
```

Figure 6: Code snippet of data transformation

```
In [10]: #Saving the file in the xlsx format
         file_name = 'Processed_Earthquakedata.xlsx'
         df.to_excel(file_name)
         print('Data is written to xlsx File successfully.')

         Data is written to xlsx File successfully.
```

Figure 7: Snippet showing saving of preprocessed data in .xlsx form.

3.3 Feature Selection and Data Split

Feature Selection is carried out to select important features for the model, this process is done using the correlation heatmap as shown in the figure 8.

```
correlation_matrix = df.corr()

# Create a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(data=correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```

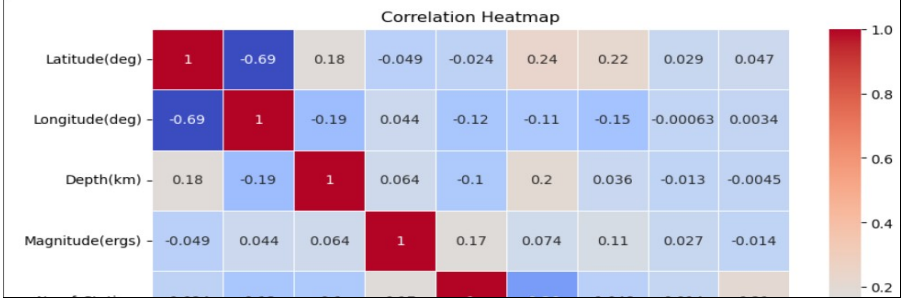

Figure 8: Correlation heatmap for feature selection

The most important features are selected, and the data is split into train and test as shown in figure 9.

```
from sklearn.model_selection import train_test_split

# Selecting the relevant columns through inference from the correlation table
X = df[['Latitude(deg)', 'Longitude(deg)', 'Depth(km)', 'No_of_Stations']]
y = df['Magnitude(ergs)']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

Figure 9: Feature Scaling and Data splitting

3.4 Model Building

Various statistical and machine learning models are utilized in this research, and they include multiple linear regression, support vector machines, random forest and naïve bayes model. Figure 10 shows the development of the random forest model.

```
from sklearn.ensemble import RandomForestRegressor

# Initializing a random forest regressor with 100 trees
rf = RandomForestRegressor(n_estimators=100, random_state=42)

# Fit the regressor to the training data
rf.fit(X_train, y_train)
```

```
RandomForestRegressor(random_state=42)
```

```
# Prediction of the target variable on the test data
y_prediction = rf.predict(X_test)
```

Figure 10: Random Forest Model Development

The SVM model is developed as shown in figure 11 below

```
from sklearn.svm import SVR

# Selecting the subset size of the training data
subset_size = 500
X_train_subset = X_train[:subset_size]
y_train_subset = y_train[:subset_size]

# Creating the SVM model
svm = SVR(kernel='rbf', C=1e3, gamma=0.1)

# Train the model on the subset of data
svm.fit(X_train_subset, y_train_subset)

# Evaluate the model on the test set
score = svm.score(X_test, y_test)
print("Test score:", score)
```

```
Test score: -2.62732111259047
```

Figure 11: Support Vector Machines Model Development

The snippet below shows the development of the naïve bayes model.

```
import pandas as pd
import numpy as np
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.preprocessing import LabelEncoder, MinMaxScaler
import matplotlib.pyplot as plt
import seaborn as sns

# Convert the magnitude column to categorical data
df['Magnitude_Category'] = pd.cut(df['Magnitude(ergs)'], bins=[0, 5, 6, 7, np.inf], labels=['Minor', 'Moderate', 'Strong', 'Majo
# Encode Magnitude Category
le = LabelEncoder()
df['Magnitude_Category_Encoded'] = le.fit_transform(df['Magnitude_Category'])

# Normalize latitude and longitude values
scaler = MinMaxScaler()
df[['Latitude(deg)', 'Longitude(deg)']] = scaler.fit_transform(df[['Latitude(deg)', 'Longitude(deg)']])

# Select features
X = df[['Latitude(deg)', 'Longitude(deg)', 'No_of_Stations']]
y = df['Magnitude_Category_Encoded']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Train the Gaussian Naive Bayes model on the training data
gnb = GaussianNB()
gnb.fit(X_train, y_train)
```

Figure 12: Naïve Bayes Model Development

The mulitple linear regression model is built as shown in the figure 13 below.

MULTIPLE LINEAR REGRESSION

```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
LinearRegression()
```

```
LinearRegression()
```

```
LinearRegression()
```

Figure 13: Linear Regression Model Development

3.5 Model Evaluation

The most popular evaluation metrics for regression tasks includes Mean Square Error and R-Square Chico D (2021). These metrics are utilized coupled with accuracy as a metric.
Figure 14 shows the evaluation of the linear regression model, figure 15, figure 16 and figure 17 shows the evaluation of the random forest model, support vector machine model and naïve bayes model respectively

```
# Predict on the testing set
y_pred = regressor.predict(X_test)

# Compute R^2 and MSE
r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)

scores['mse'].append(mse)
scores['R^2'].append(r2)

print("R^2: {:.2f}, MSE: {:.2f}".format(r2, mse))
```

Figure 14: Evaluation of Linear Regression Model

```
# Prediction of the target variable on the test data
y_prediction = rf.predict(X_test)

# Evaluation of the performance of the model using R^2 score and mean squared error
mse = mean_squared_error(y_test, y_prediction)
r2 = r2_score(y_test, y_prediction)

scores['mse'].append(mse)
scores['R^2'].append(r2)

print('Mean Squared Error is: ', mse)
print('R^2 Score is: ', r2)
```

Figure 15: Evaluation of Random Forest Model

```
# Prediction on the test dataset
y_pred_svm = svm.predict(X_test)

# Compute the R^2 and MSE
r2_svm = r2_score(y_test, y_pred_svm)
mse_svm = mean_squared_error(y_test, y_pred_svm)

scores['mse'].append(mse_svm)
scores['R^2'].append(r2_svm)

print("SVM R^2: {:.2f}, MSE: {:.2f}".format(r2_svm, mse_svm))
```

Figure 16: Evaluation of Support Vector Machine Model

```
#GAUSSIAN ACCURACY
accuracy = accuracy_score(y_test, y1_pred)
print('Accuracy:', accuracy)
```

Figure 17: Evaluation of Naïve-Bayes Model

The RSE values of the model is as a result of the correlation between the independent variables of the dataset and the dependent variables (Magnitude) Figure 18 shows the correlation matrix of the data.

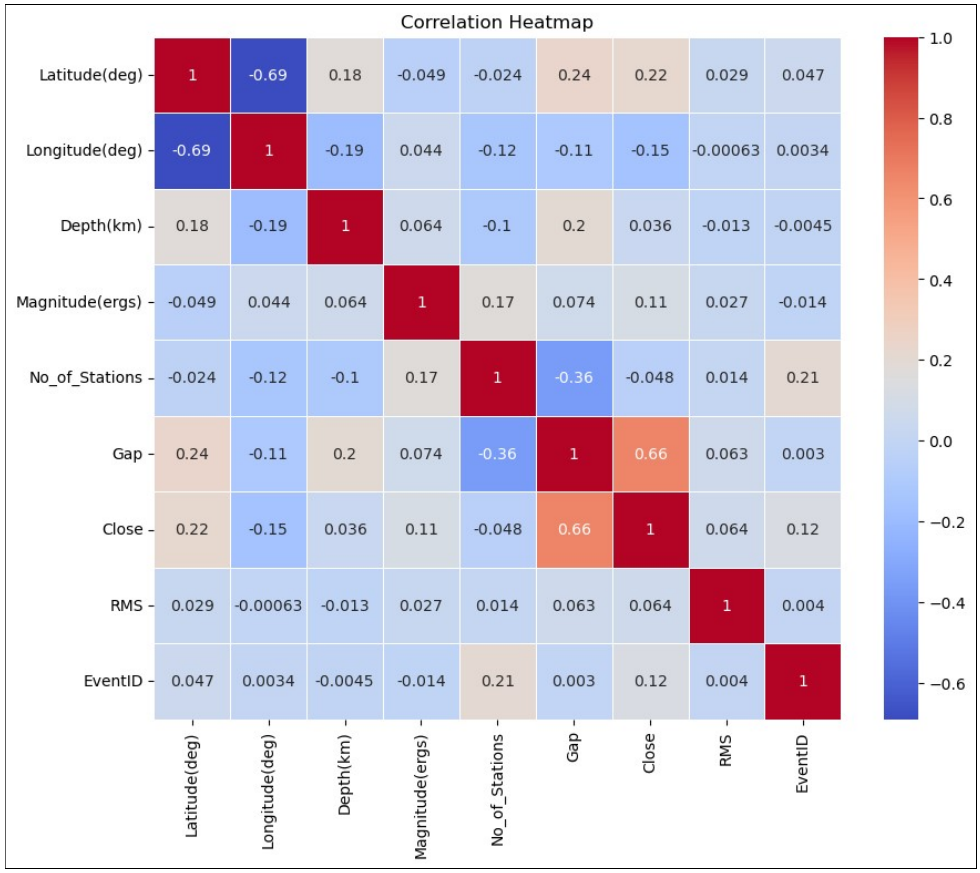

Figure 18: Correlation matrix.

References

Chicco, D., Warrens, M.J. and Jurman, G., 2021. The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. *PeerJ Computer Science*, *7*, p.e623.