Earthquake Prediction

Abstract:

It is well known that if a disaster occurs in one region, it is likely to happen again. With the increase in the use of technology, many seismic monitoring stations have increased, so we can use machine learning and other data-driven methods like linear regression to predict earthquakes. In this project, we are going to create a model for the task of Earthquake Prediction using Machine Learning and Python programming language based on parameters such as longitude, latitude, duration, country, depth etc. Index Terms-earthquake, magnitude, seismic, regression, machine learning.

I. INTRODUCTION:

There exist many events in the modern world that cannot be influenced by human beings. Such events include natural disasters like tsunamis, tornadoes, floods, volcanic eruptions, etc. Human beings cannot stop the impending threat, rather, we can only take precautionary measures and rapid response in order to minimize the economical and human losses. However, not all natural disasters are equally well studied and predicted. Earthquakes are one of the most dangerous and destructive catastrophes. Firstly, they often occur without explicit warning and therefore do not leave enough time for people to take measures. In addition, the situation is compounded by the fact that earthquakes often lead to other natural hazards such as tsunamis and landslides. However, rapid development of machine learning methods and successful application of these methods to various kinds of problems indicates that these technologies could help in making accurate predictions

II. PROBLEM DEFINITION:

Earthquake prediction using machine learning (ML) is a task of developing a model that can accurately predict the magnitude of an earthquake, given parameters such as longitude, latitude,

depth, duration magnitude, country etc. We use various prediction models to determine this.

III. DATASETS:

When a specific field is researched in terms of machine learning, the first question is where to find data. As for earthquake datasets, various organizations and research institutions are constantly monitoring seismic activity of all over the world. The structure of earthquake datasets are usually presented in the form of a table, each record of which corresponds to a certain seismic event. The sets of attributes are different for data published in different datasets. In this project we make use of two datasets.

Link for datasets: https://drive.google.com/drive/folders/1b4IyWfUzbDgOv uW_W7rOSEeTpBN53Mlf?usp= share_link

DATASET 1:

earthquake1.csv Contains all the recorded earthquakes in Turkey between 1910-2017 larger than 3.5 intensity. The parameters are:

• Id: order number of the earthquake

• Date: earthquake occurrence date

• Time: time of the earthquake

• Lat: latitude of the earthquake epicenter

• Long: longitude of the earthquake epicenter

• Country: country of the earthquake epicenter

• City: province of the occurred earthquake

• Area: region of the occurred earthquake

• Direction: direction of the earthquake signa

l • Dist: distance of direction in km

• Depth: depth of the occurred earthquake (distance from the surface)

• Xm: the largest of the given magnitude values

• Md: magnitude depending on time

• Richter: Richter magnitude or the local magnitude (ML)

• Mw: moment magnitude

• Ms: surface wave magnitude

• Mb: body wave magnitude

DATASET 2:

earthquake2.csv Contains all Significant Earthquakes that occurred in the period of 1965-2016.The parameters are:

• Time: Time of the earthquake

• Latitude: latitude of the earthquake epicenter

• Longitude: longitude of the earthquake epicenter

• Depth: depth of the occurred earthquake (distance from the surface)

• mag: size of the earthquake

• magType: type of magnitude occurred (ml or ms)

• Nst: Number of seismic stations

• Gap: Region along an active fault where stress is accumulating because no

• earthquakes have occurred there recently

• D-min: Horizontal distance from the epicenter to the nearest station

• RMS: In general, the smaller this number, the more reliable is the calculated depth of the earthquake

• Id: order number of the earthquake

• Updated: updation in sequence of earthquake

• Place: place where earthquake occurred

• Type: type of earthquake occurred

• Horizontal error: error in range

• Depth error: error occurred in calculation of depth of earthquake

• magError: error occurred in calculation of magnitude of earthquake

• magNst: error occurred in calculation of number of seismic stations

• Status: current status of earthquake

• Location source: source of location where earthquake occurred

• magSource: source where magnitude is occurred

IV. METHODS :

To begin with, the dataset is pre-processed to make it suitable for modelling. Second, the dataset is partitioned into training and testing data-set and third, different models are

constructed and fitted to the training data-set. The accuracy of the system is obtained by testing the system using the testing data. In this study, we explore, evaluate and analyze dataset

characteristics using different machine learning algorithms like linear regression, decision tree and KNN.

A. Linear Regression:

Linear Regression is a supervised Machine Learning model which assumes a linear relationship between the input variables (x) and the single output variable (y). For linear regression, we will use only numerical data (float). This form of analysis estimates the coefficients of the linear equation, involving one or more independent variables that best predict the value of the dependent variable. For Earthquake prediction, we use various input variables (X) such as earthquake occurrence date, surface wave magnitude, body wave magnitude, Richter magnitude or the local magnitude (ML) for the first dataset to predict the output variable (Y) - Xm, the largest of the given magnitude values. For the 2nd dataset, we predict the value of magnitude (Y) using variables such as the number of seismic stations, type of magnitude that occurred (ml or ms), gap etc.

B. Decision Tree:

Decision-tree algorithm falls under the category of supervised learning algorithms. It works for both continuous as well as categorical output variables. Decision tree regression observes the features of an object and trains a model in the structure of a tree to predict data in the future to produce meaningful continuous output. The goal of a decision tree is to learn a model that predicts the value of the magnitude of the earthquake by learning simple decision rules inferred from the data features of the given dataset. For example, if the Richter value > 0.49, then XM value will be approx. 0.07. For our decision tree, the first node is the column with the highest information gain i.e., X[9] which is md - magnitude depending on time.

C. K-Nearest Neighbor:

K nearest neighbours is a simple algorithm that uses ‘feature similarity’ to predict the values of any new data points. This means that the new point is assigned a value based on how closely it resembles the points in the training set. the KNN algorithm would find the k-nearest data points to a new data point and predict the continuous output based on the average of the output values of the nearest neighbours. For earthquake prediction, we use k-Nearest Neighbours Regressor. Even though we had multiple features, the importance of each feature in the Euclidean distance calculation can vary. To handle this, we performed normalization before calculating the distance, so that all features are on the same scale and have an equal contribution to the distance

V. EVALUATION AND DISCUSSIONS DATA PRE-PROCESSING :

For both the datasets the following pre-processing steps were taken:

• First, we dropped the ‘id’ column since it is not useful for prediction.

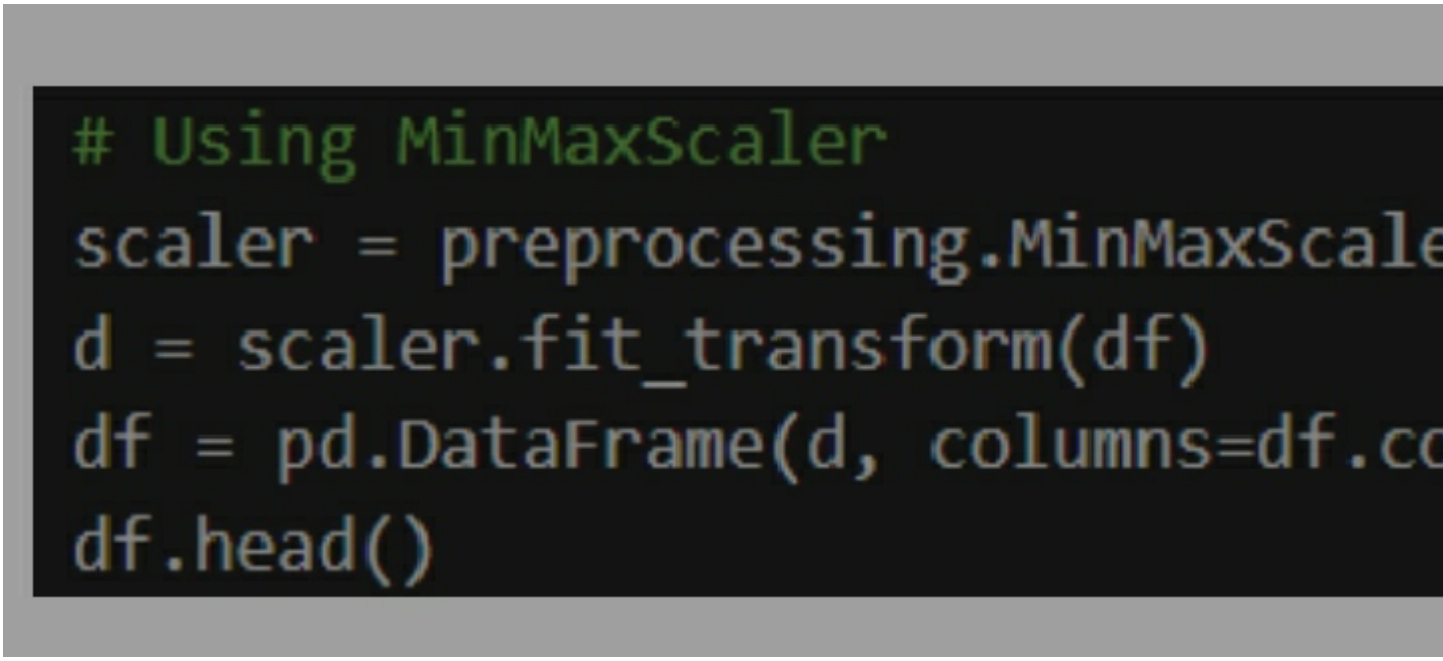
• Next we converted categorical variables into numerical values so that it could be easily fitted to a machine learning model using a label encoder.

```
# Data Encoding
label_encoder = preprocessing.LabelE
for col in df.columns:
    if df[col].dtype == 'object':
        label_encoder.fit(df[col])
        df[col] = label_encoder.transf
df.dtypes
```

• Imputation with SimpleImputer which replaced the missing data with mean was done to handle null values.

```
# Imputing Missing Values with Mean
si=SimpleImputer(missing_values = np.na
si.fit(df[["dist","mw"]])
df[["dist","mw"]] = si.transform(df[["d
df.isnull().sum()
```

• Normalization of the data was done using MinMax Scaler. The Min-Max scaling method helps the dataset to shift and rescale the values of their attributes, so they end up ranging between 0 and 1.



```
# Using MinMaxScaler
scaler = preprocessing.MinMaxScale
d = scaler.fit_transform(df)
df = pd.DataFrame(d, columns=df.co
df.head()
```

• Dataset1 had two columns ‘date’ and ‘time’. It was converted to another column ‘timestamp’ using packages such as ‘datetime’ and ‘time’.

```
import datetime
import time

timestamp = []
for d, t in zip(df['date'], df['time']):
  ts = datetime.datetime.strptime(d+' '+t, '%Y
  timestamp.append(time.mktime(ts.timetuple())
timeStamp = pd.Series(timestamp)
df['Timestamp'] = timeStamp.values
final_data = df.drop(['date', 'time'], axis=1)
final_data = final_data[final_data.Timestamp !
df = final_data
df.head()
```

SPLITTING THE DATASET:

The datasets are split into training and testing datasets. The dataset 1 is split into 80% training and 20% testing, with a random state of 2, meaning that the same random sample will be used each time the code is run.

```
y=np.array(df['xm'])
X=np.array(df.drop('xm',axis=1))
from sklearn.model_selection import train
X_train,X_test,y_train,y_test=train_test_
_size=0.2,random_state=2)
```

The dataset 2 is split into 75% and 25% for training and testing respectively with a random state of 2.

```
y=np.array(df['xm'])
X=np.array(df.drop('xm',axis=1))
from sklearn.model_selection import trai
X_train,X_test,y_train,y_test=train_test_
_size=0.25,random_state=2)
```

The resulting training sets are stored in variables X_train, y_train, while the testing sets are

stored in X_test, y_test

CREATING MODELS :

A. Linear Regression :

```
from sklearn.linear_model import Line
start1 = time.time()
linear=LinearRegression()
linear.fit(X_train,y_train)
ans1 = linear.predict(X_test)
end1 = time.time()
t1 = end1-start1
```

B. Decision Tree :

```
from sklearn.tree import DecisionTreeReg
start2 = time.time()
regressor = DecisionTreeRegressor(random
regressor.fit(X_train,y_train)
ans2 = regressor.predict(X_test)
end2 = time.time()
t2 = end2-start2
```

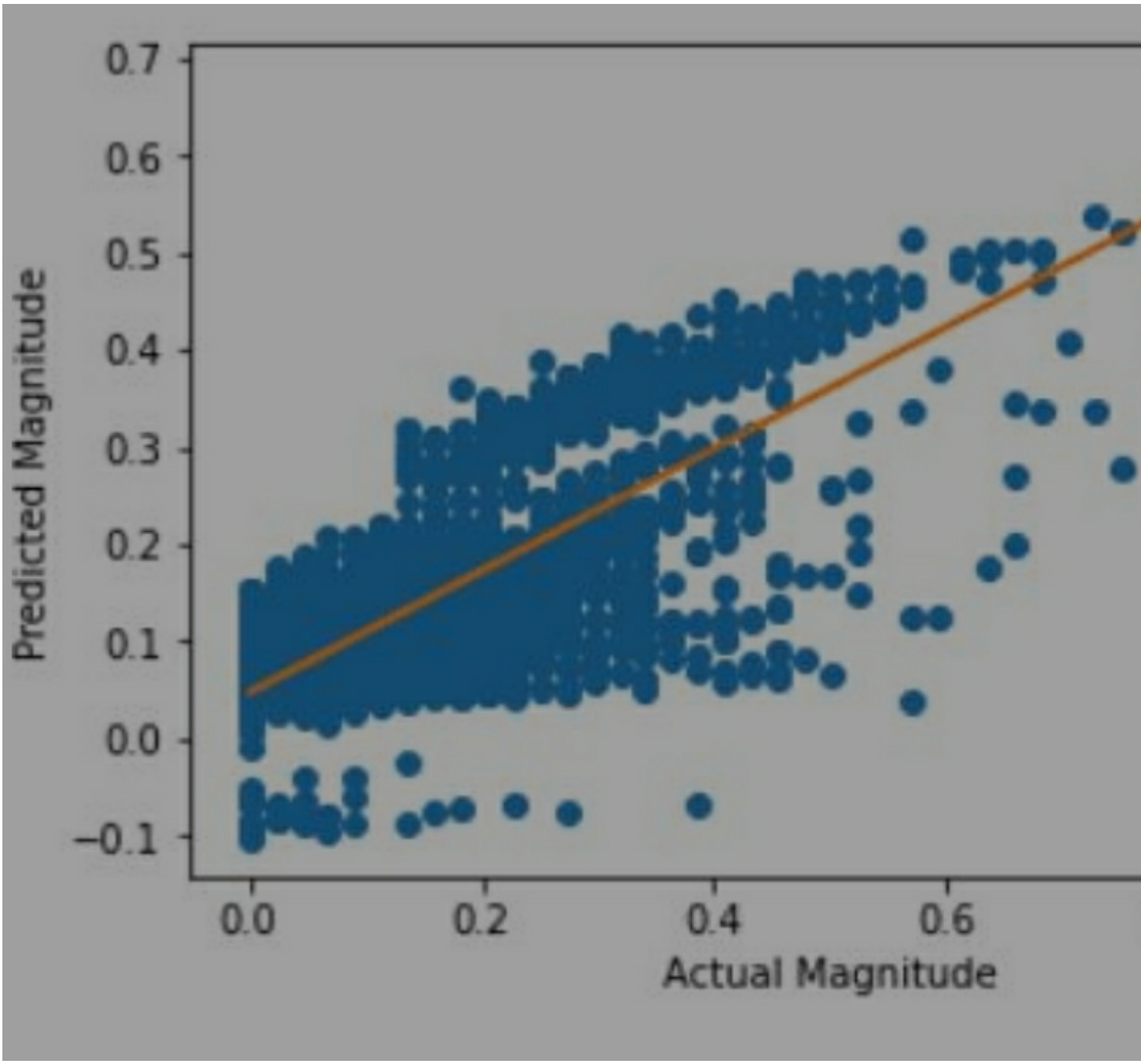
C. KNN:

```
from sklearn.neighbors import KNeighb
start3 = time.time()
knn = KNeighborsRegressor(n_neighbors
knn.fit(X_train, y_train)
ans3 = knn.predict(X_test)
end3 = time.time()
t3 = end3-start3
```

RESULTS DATASET 1:

earthquake1.csv Linear Regression Model

- Accuracy of Linear Regression model is: 0.63134131503029
- Mean Absolute Error: 0.05878246463205686
- Mean Squared Error: 0.00625827169726636
- Root Mean Squared Error: 0.07910923901331854



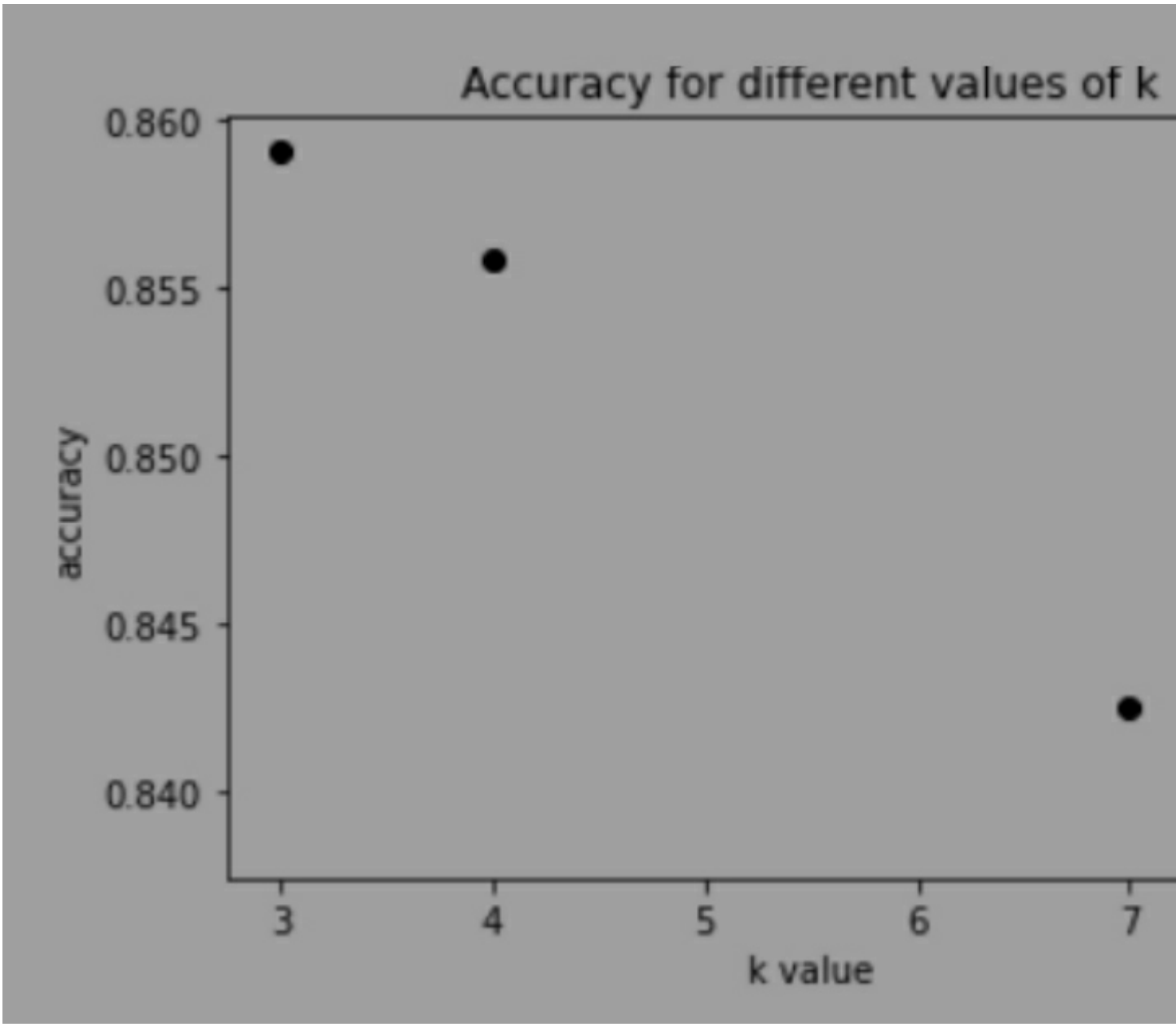
Decision Tree Model :

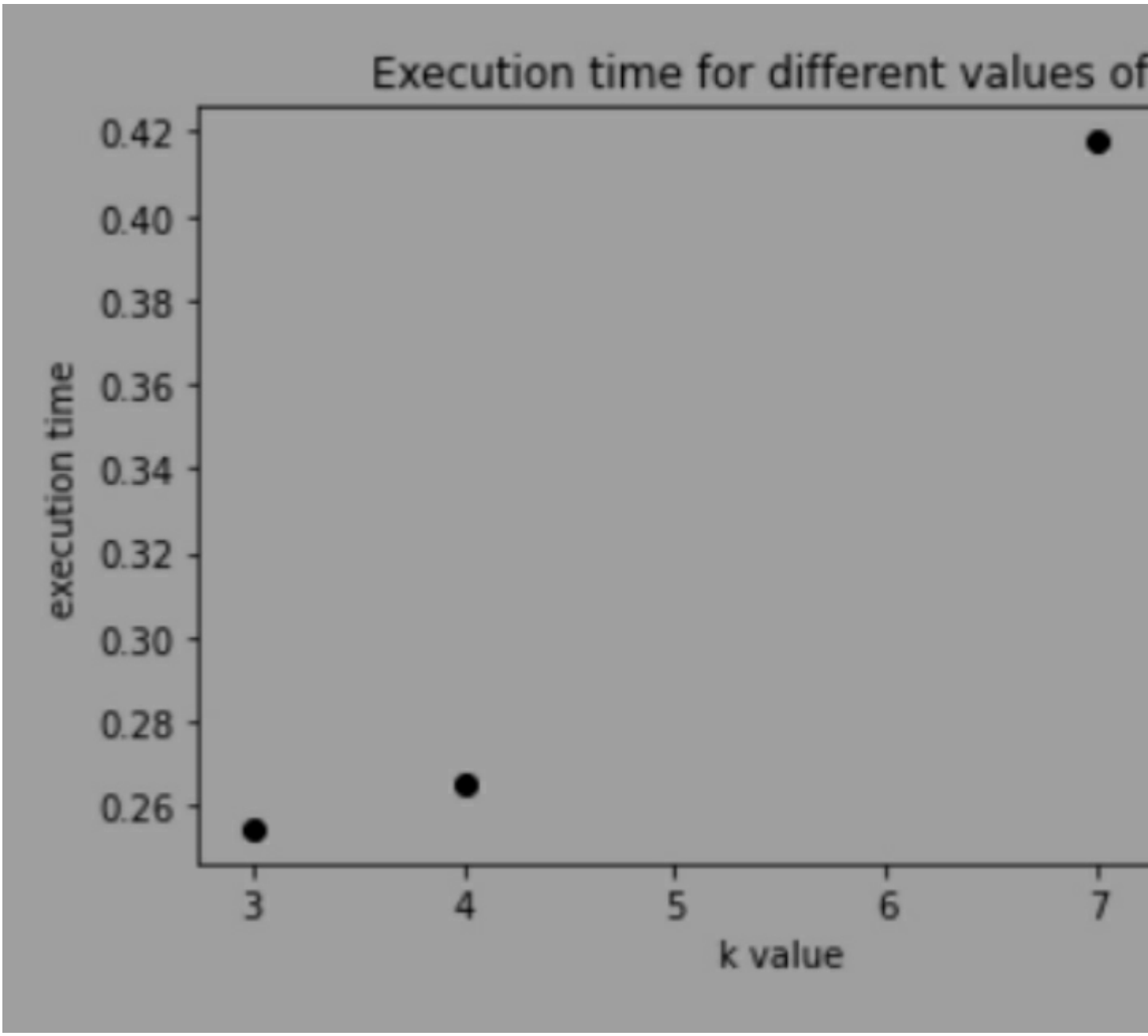
• Accuracy of Decision Tree model is: 0.9932960893884235.

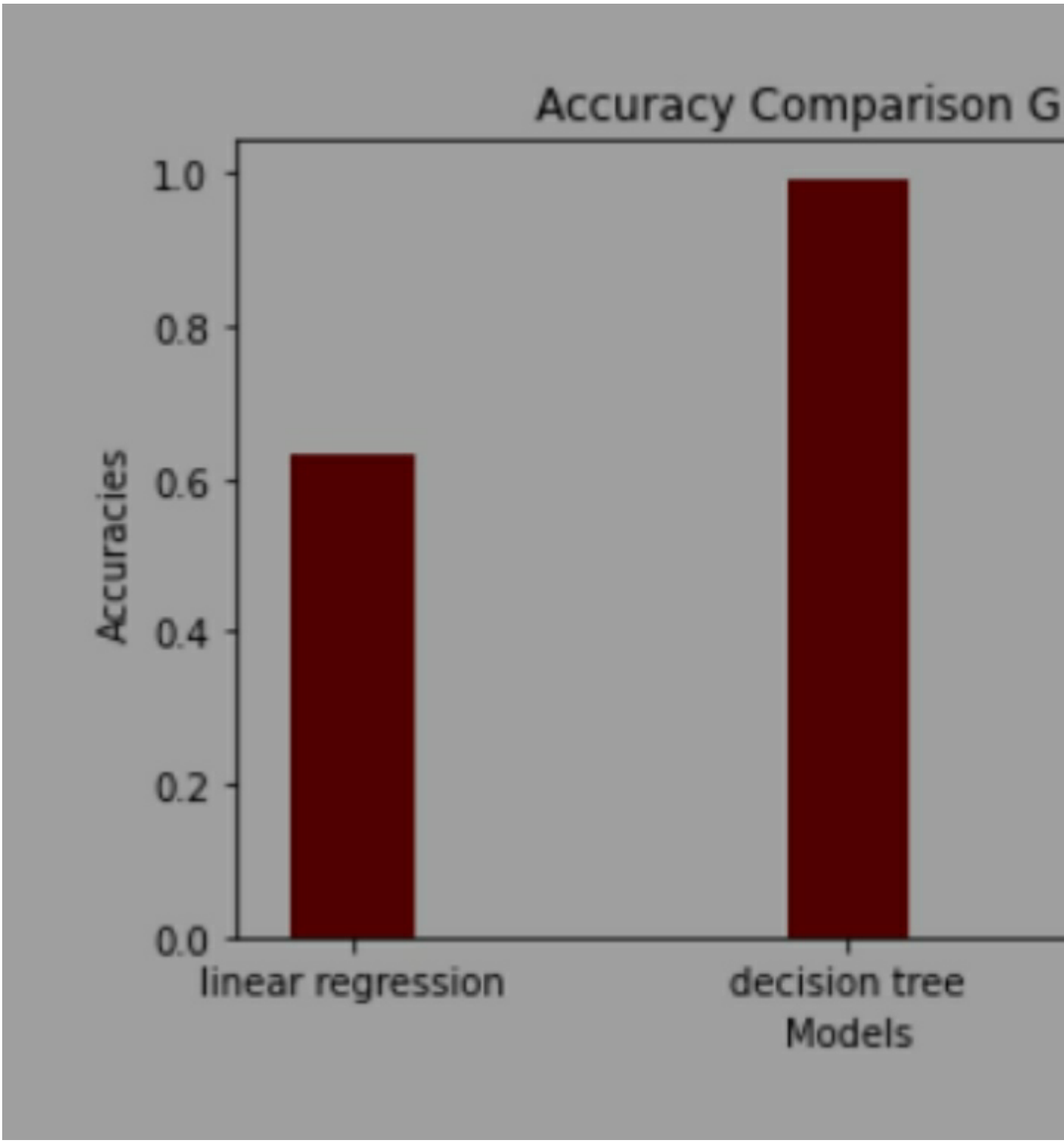
- Mean Absolute Error: 0.0006909999621372331
- Mean Squared Error: 0.00011380416561969702
- Root Mean Squared Error: 0.010667903525046383

KNN Model:

- Accuracy of KNN model is: 0.8457466919393031
- Mean Absolute Error: 0.03305598677318794
- Mean Squared Error: 0.002618571462992348
- Root Mean Squared Error: 0.051171979275696854


Accuracy for different values of k
0.860
0.855
0.850
0.845
0.840
accuracy
3
4
5
6
7
k value


Execution time for different values of
0.42
0.40
0.38
0.36
0.34
0.32
0.30
0.28
0.26
execution time
3
4
5
6
7
k value


Accuracy Comparison G
1.0
0.8
0.6
0.4
0.2
0.0
Accuracies
linear regression
decision tree
Models

DATASET 2:

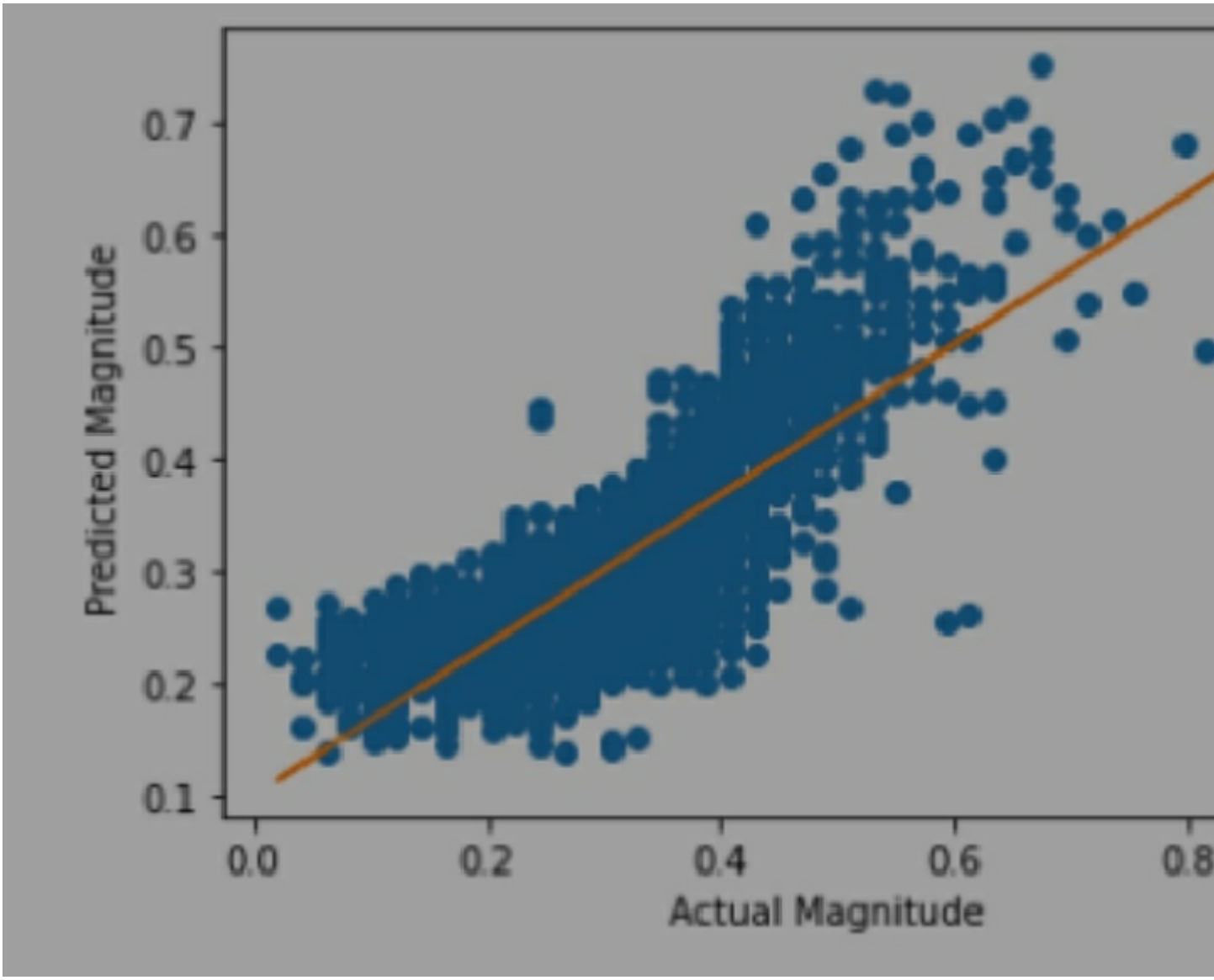
earthquake2.csv Linear Regression Model

• Accuracy of Linear Regression model is: 0.6520026760336074

• Mean Absolute Error: 0.0447298826759214

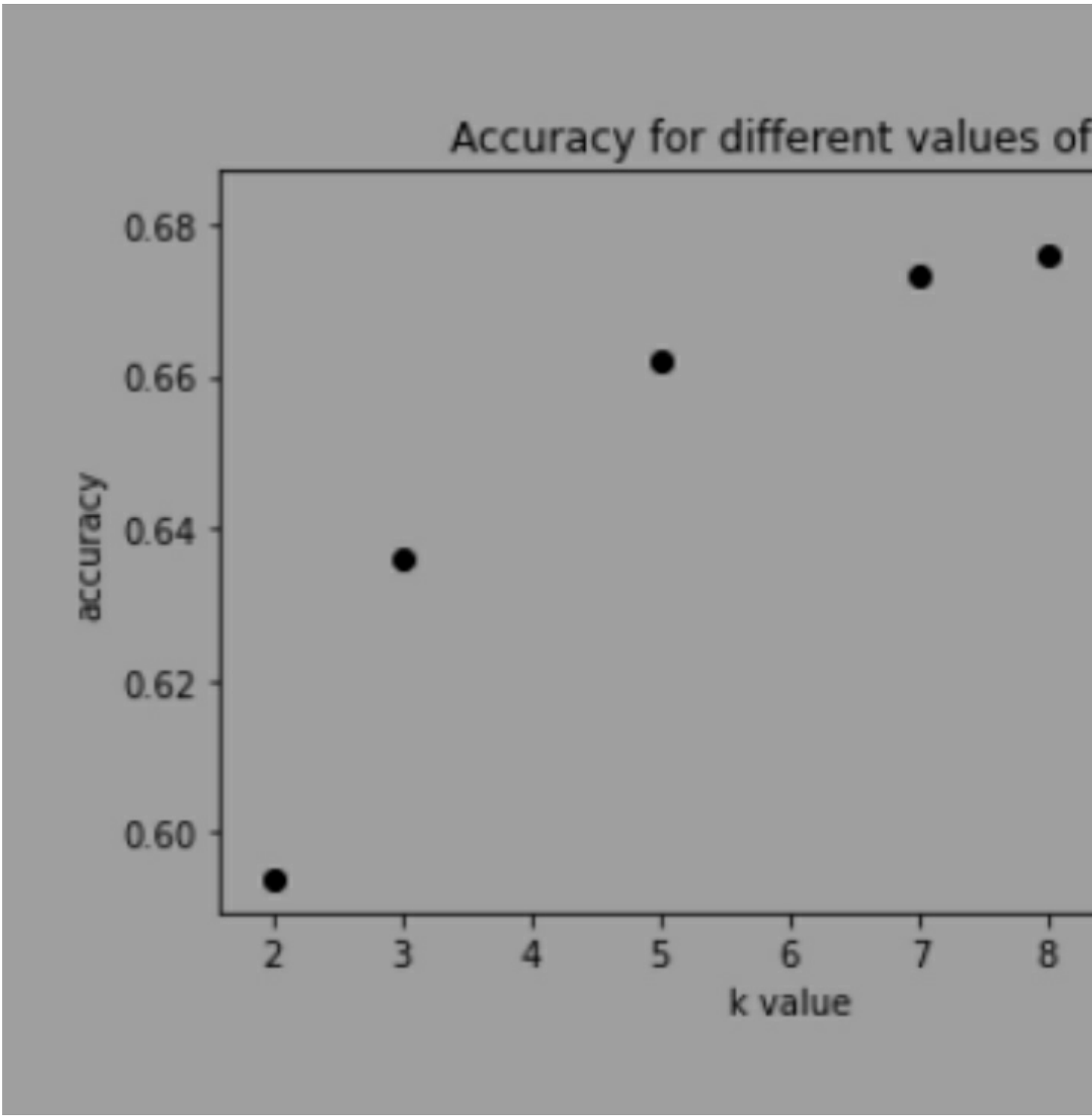
• Mean Squared Error: 0.0034475163509273773

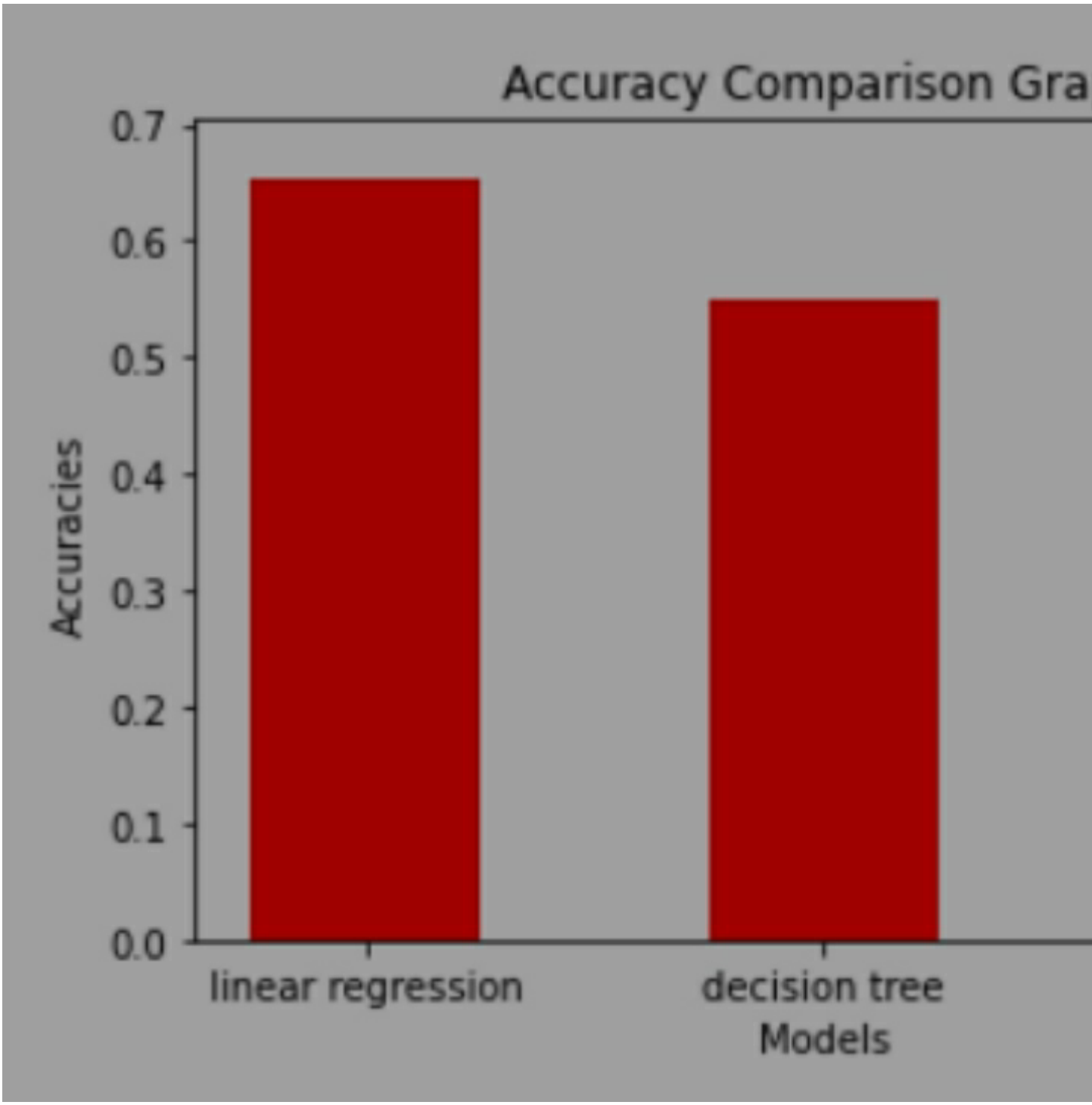
• Root Mean Squared Error: 0.058715554590988726



Decision Tree Model:

- Accuracy of Decision Tree model is: 0.5488812291089398
- Mean Absolute Error: 0.049579341940857974
- Mean Squared Error: 0.004469112926303383
- Root Mean Squared Error: 0.06685142426533172 KNN Model
- Accuracy of KNN model is: 0.6828276373795497
- Mean Absolute Error: 0.04780785783701236
- Mean Squared Error: 0.004023436946623714
- Root Mean Squared Error: 0.06343056791976337


Accuracy for different values of
0.68
0.66
0.64
0.62
0.60
accuracy
2
3
4
5
6
7
8
k value


Accuracy Comparison Gra
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Accuracies
linear regression
decision tree
Models


Execution Time Compariso
Execution Time
0.0
0.1
0.2
0.3
0.4
0.5
0.6
linear regression
decision tree
Models

VI. FINDINGS AND FUTURE DIRECTIONS:

Comparing the performances of different ML models, we find that, For Dataset1, Minimum error is obtained in the Decision Tree with the maximum accuracy of 99.4 per cent, whereas, the error is maximum for linear regression which has the minimum accuracy amongst the 3 algorithms with 63.1 percent. For Dataset2, Minimum error is obtained in the linear regression model, however KNN has the maximum accuracy of 68.2 per cent. Error is maximum for decision tree which has lowest accuracy among the three models. In both the datasets, KNN model took the most time for executing whereas linear regression took the least amount of time. The field of earthquake prediction is an active area of research and there have been significant advances in recent years. One of the most promising areas of research is the use of machine learning and artificial intelligence to analyze seismic data and identify patterns that may indicate an impending earthquake. Other areas of research include the use of satellite data, remote sensing, and geodetic measurements to detect precursory signs of an earthquake. While we do not yet have the capability to predict earthquakes with 100% accuracy, continued research and advancements in technology may bring us closer to this goalin the future.

VII. CONCLUSION :

In this project we used two datasets containing information about earthquakes which happened around the world to predict the earthquake magnitude. We trained the models using various machine learning algorithms to find which will give us the most accurate result. We compared the results of Linear regression, Decision Tree and K-Nearest Neighbor.

Earthquake prediction is challenging for several reasons. First, the processes that lead to earthquakes are not fully understood. Second, the data that is used for prediction is limited and may not be reliable. Third, there are many variables that can affect earthquakes, making it difficult to predict their occurrence with a high degree of accuracy. To effectively utilize the datasets obtained, pre-processing and data visualization of the data were performed.

VIII.APPENDIX:

NumPy:

NumPy is a Python library for working with large, multi-dimensional arrays and matrices of numerical data, as well as a large collection of high-level mathematical functions to operate on these arrays.

Pandas:

Pandas is a library for data manipulation and analysis in Python, providing efficient data structures and powerful manipulation capabilities.

Matplotlib:

Matplotlib is a plotting library for creating static, animated, and interactive visualizations in Python.

Seaborn:

Seaborn is a data visualization library for creating attractive and informative statistical graphics in Python, built on top of Matplotlib. Scikit-learn: Scikit-learn is a machine learning library for Python, providing simple and efficient tools for data mining and data analysis.

ACKNOWLEDGMENT:

We would like to express our sincere gratitude to Dr. Aiswarya S. Kumar for her guidance throughout this project. Our appreciation to our friends and family for their input and thoughts on this study. Furthermore, we would like to express our sincere gratitude to Amrita University for allowing us to perform this study and Chancellor Amma for her presence and grace

REFERENCES:

[1] scikit-learn developers Decision Tree Regression

https://scikitlearn.org/stable/auto_examples/tree/plot_tree_regression.html

[2] Aishwarya Singh A Practical Introduction to K-Nearest Neighbors Algorithm for Regression

https://www.analyticsvidhya.com/blog/2018/08/k-nearestneighbor-introduction-regression-python/

[3] Mahadev Maitri Earthquake Prediction

https://www.kaggle.com/code/mahadevmm9/earthquakeprediction

[4] Caganseval Earthquakes in 1910-2017, Turkey

https://www.kaggle.com/datasets/caganseval/earthquake

[5] Python Software Foundation datetime -- Basic date and time types

https://docs.python.org/3/library/datetime.html