Software Engineering Tools Lab

Assignment No-2

(Module 2- Software Development Frameworks)

2019BTECS00058 & 2019BTECS00061

Devang Kamble and Rohit Biradar

1. Give answers for questions for the Android SDK

a. **Original Author(s)** - Andy Rubin, Rich Miner, Nick Sears, and Chris White
b. **Developers** - Google, JetBrains
c. **Initial release** - September 23, 2008
d. **Stable release -** Android 12 / October 4, 2021
e. **Preview release -** Android 13 developer previews throughout February 2022 and March 2022. It shall move to beta releases around April 2022.
f. **Repository (with cloud support)** – GitHub Link of Latest Release
g. **Written in (Languages) –** Java
h. **Operating System support -** *OS* X, Linux or Windows
i. **Platform portability –** There is one SDK Platform available for each version of Android. It includes an android.jar file with a fully compliant Android library. In order to build an Android app, you must specify an SDK platform as your build target. This build is independent of the platform and can be used as an engine to build through any OS using a software like Android Studio.
j. **Available in (Total languages) -** Kotlin, Java, and C++
k. **List of languages supported** - Kotlin, Java
l. **Type (Programming tool, integrated development environment etc.)** - An IDE like Android Studio or Visual Studio Code with Terminal support.
m. **Website** - developer.android.com

n. **Features:**
   a. Beautiful UI
   b. Connectivity
   c. Storage, Media support
   d. Messaging
   e. Web Browser
   f. Multi-touch
   g. Multi-tasking
   h. Resizable widgets
   i. Multi-Language

o. **Size (in MB, GB etc.)** - 872 MB for the current release

p. **Privacy and Security** - Android is private by design owned by Google. As the Android platform evolves, it continues to provide tools and guidance to help developers design apps that minimize the amount of data that is used and give users control and transparency so that users can stay informed and decide what data to share. The codebase hence is Open Source to give developers total access.

q. **Type of software (Open source/License)** - Open Source

r. **If License- Provide details** – SDK License by Google (Link)

s. **Latest version** – Android 12

t. **Cloud Support** – Yes, many under GCP (Link)

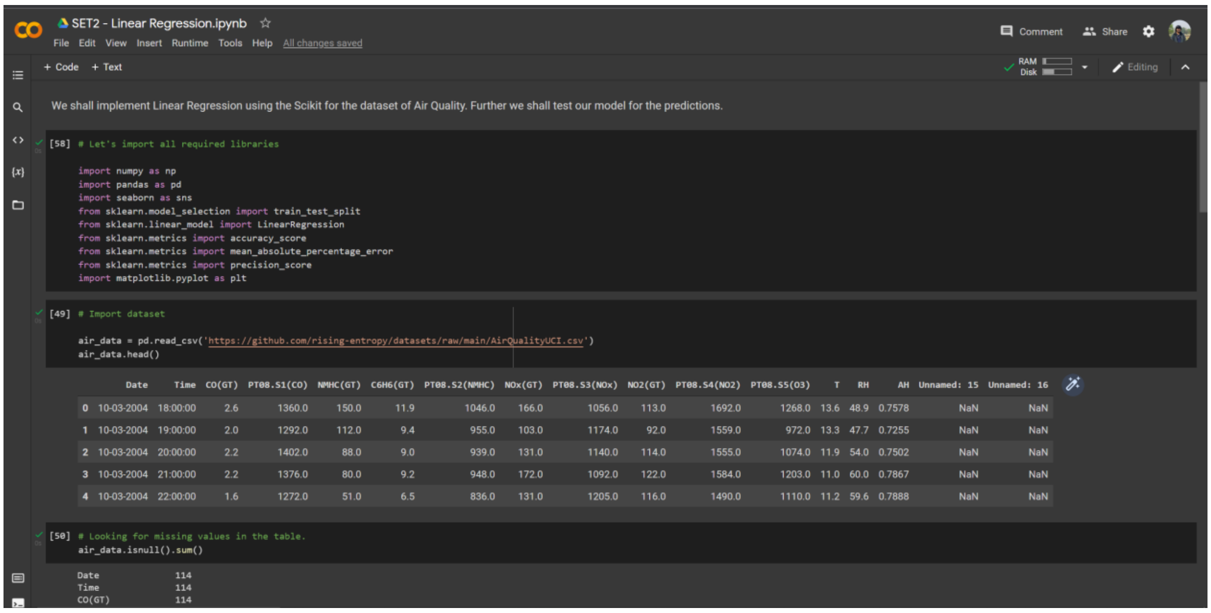
u. **Applicability** – Build, Debug, Run and Test Android applications which then run on any device with Android OS.

v. **Drawbacks (if any)** – Careful programming necessary for all devices and SDK versions for Android. Limitations with XML architecture. Computation intensive development required.

2. Implement linear regression problem using Google Colab (Perform pre-processing, training and testing)

   Dataset: Air Quality

We build on Google Colab:


SET2 - Linear Regression.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing

We shall implement Linear Regression using the Scikit for the dataset of Air Quality. Further we shall test our model for the predictions.

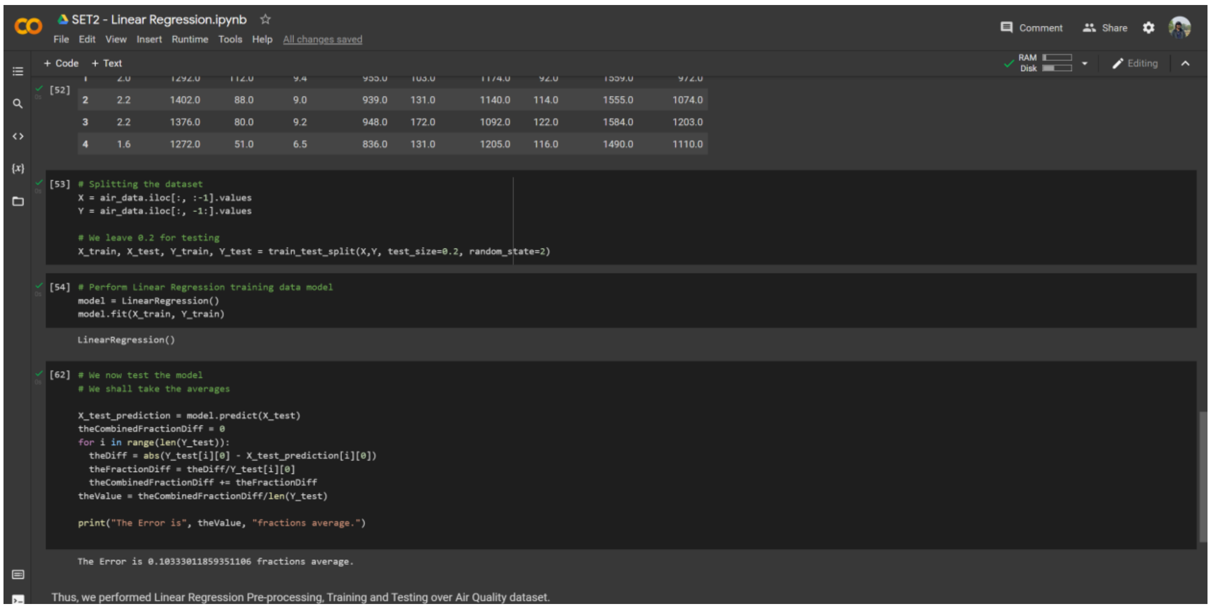
```
[58] # Let's import all required libraries

     import numpy as np
     import pandas as pd
     import seaborn as sns
     from sklearn.model_selection import train_test_split
     from sklearn.linear_model import LinearRegression
     from sklearn.metrics import accuracy_score
     from sklearn.metrics import mean_absolute_percentage_error
     from sklearn.metrics import precision_score
     import matplotlib.pyplot as plt
```

```
[49] # Import dataset

     air_data = pd.read_csv('https://github.com/rising-entropy/datasets/raw/main/AirQualityUCI.csv')
     air_data.head()
```

	Date	Time	CO(GT)	PT08.S1(CO)	NMHC(GT)	C6H6(GT)	PT08.S2(NMHC)	NOx(GT)	PT08.S3(NOx)	NO2(GT)	PT08.S4(NO2)	PT08.S5(O3)	T	RH	AH	Unnamed: 15	Unnamed: 16
0	10-03-2004	18:00:00	2.6	1360.0	150.0	11.9	1046.0	166.0	1056.0	113.0	1692.0	1268.0	13.6	48.9	0.7578	NaN	NaN
1	10-03-2004	19:00:00	2.0	1292.0	112.0	9.4	955.0	103.0	1174.0	92.0	1559.0	972.0	13.3	47.7	0.7255	NaN	NaN
2	10-03-2004	20:00:00	2.2	1402.0	88.0	9.0	939.0	131.0	1140.0	114.0	1555.0	1074.0	11.9	54.0	0.7502	NaN	NaN
3	10-03-2004	21:00:00	2.2	1376.0	80.0	9.2	948.0	172.0	1092.0	122.0	1584.0	1203.0	11.0	60.0	0.7867	NaN	NaN
4	10-03-2004	22:00:00	1.6	1272.0	51.0	6.5	836.0	131.0	1205.0	116.0	1490.0	1110.0	11.2	59.6	0.7888	NaN	NaN

```
[50] # Looking for missing values in the table.
     air_data.isnull().sum()

     Date              114
     Time              114
     CO(GT)            114
```

Our simple Linear Regression Model gave approximately 90% accurate value (fractional mean).


SET2 - Linear Regression.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing

[52]

1	2.0	1292.0	112.0	9.4	955.0	103.0	1174.0	92.0	1559.0	972.0
2	2.2	1402.0	88.0	9.0	939.0	131.0	1140.0	114.0	1555.0	1074.0
3	2.2	1376.0	80.0	9.2	948.0	172.0	1092.0	122.0	1584.0	1203.0
4	1.6	1272.0	51.0	6.5	836.0	131.0	1205.0	116.0	1490.0	1110.0

```
[53] # Splitting the dataset
     X = air_data.iloc[:, :-1].values
     Y = air_data.iloc[:, -1:].values

     # We leave 0.2 for testing
     X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.2, random_state=2)
```

```
[54] # Perform Linear Regression training data model
     model = LinearRegression()
     model.fit(X_train, Y_train)

     LinearRegression()
```

```
[62] # We now test the model
     # We shall take the averages

     X_test_prediction = model.predict(X_test)
     theCombinedFractionDiff = 0
     for i in range(len(Y_test)):
       theDiff = abs(Y_test[i][0] - X_test_prediction[i][0])
       theFractionDiff = theDiff/Y_test[i][0]
       theCombinedFractionDiff += theFractionDiff
     theValue = theCombinedFractionDiff/len(Y_test)

     print("The Error is", theValue, "fractions average.")

     The Error is 0.10333011859351106 fractions average.
```

Thus, we performed Linear Regression Pre-processing, Training and Testing over Air Quality dataset.

Google Colab Link: Link

Jupyter Notebook File: Link

The Code:

```
# Let's import all required libraries

import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import accuracy_score
from sklearn.metrics import mean_absolute_percentage_error
from sklearn.metrics import precision_score
import matplotlib.pyplot as plt

# Import dataset

air_data = pd.read_csv('https://github.com/rising-
entropy/datasets/raw/main/AirQualityUCI.csv')
air_data.head()

# Looking for missing values in the table.
air_data.isnull().sum()

# Handling the missing values

# Drop Date and Time
air_data = air_data.drop(columns='Date', axis=1)
air_data = air_data.drop(columns='Time', axis=1)
air_data = air_data.drop(columns='Unnamed: 15', axis=1)
air_data = air_data.drop(columns='Unnamed: 16', axis=1)
air_data = air_data.drop(columns='RH', axis=1)
air_data = air_data.drop(columns='AH', axis=1)
air_data = air_data.drop(columns='T', axis=1)

# Replacing missing values with mean
air_data['CO(GT)'].fillna(air_data['CO(GT)'].mean(), inplace=True)
air_data['PT08.S1(CO)'].fillna(air_data['PT08.S1(CO)'].mean(), inplace=True)
air_data['NMHC(GT)'].fillna(air_data['NMHC(GT)'].mean(), inplace=True)
air_data['C6H6(GT)'].fillna(air_data['C6H6(GT)'].mean(), inplace=True)
air_data['PT08.S2(NMHC)'].fillna(air_data['PT08.S2(NMHC)'].mean(),
inplace=True)
air_data['NOx(GT)'].fillna(air_data['NOx(GT)'].mean(), inplace=True)
air_data['PT08.S3(NOx)'].fillna(air_data['PT08.S3(NOx)'].mean(), inplace=True)
air_data['NO2(GT)'].fillna(air_data['NO2(GT)'].mean(), inplace=True)
air_data['PT08.S4(NO2)'].fillna(air_data['PT08.S4(NO2)'].mean(), inplace=True)
air_data['PT08.S5(O3)'].fillna(air_data['PT08.S5(O3)'].mean(), inplace=True)

air_data.isnull().sum()
```

```python
# Get dataset head

air_data.head()

# Splitting the dataset
X = air_data.iloc[:, :-1].values
Y = air_data.iloc[:, -1:].values

# We leave 0.2 for testing
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.2,
random_state=2)

# Perform Linear Regression training data model
model = LinearRegression()
model.fit(X_train, Y_train)

# We now test the model
# We shall take the averages

X_test_prediction = model.predict(X_test)
theCombinedFractionDiff = 0
for i in range(len(Y_test)):
  theDiff = abs(Y_test[i][0] - X_test_prediction[i][0])
  theFractionDiff = theDiff/Y_test[i][0]
  theCombinedFractionDiff += theFractionDiff
theValue = theCombinedFractionDiff/len(Y_test)

print("The Error is", theValue, "fractions average.")

"""Thus, we performed Linear Regression Pre-processing, Training and Testing
over Air Quality dataset."""
```