Amrita School of Computing
Department of Computer Science & Engineering
19CSE304 Foundations of Data Science
Lab Sheet 5

Exercise 1 Data standardization and Normalization Practice

I. min-max normalization

a. Load the dataset

```
import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

 # Sklearn library

from sklearn import preprocessing

data_set = pd.read_csv('Iris.csv')

data_set.head()
```

b. Then we select the features that need to be scaled.

```
x = data_set.iloc[:, 1:5].values

print ("Original data values: \n",  x)
```

c. Perform min-max scaling on the selected features.

```
from sklearn import preprocessing

min_max_scaler = preprocessing.MinMaxScaler(feature_range =(0, 1))

# Scaled feature

scaled_x= min_max_scaler.fit_transform(x)

print ("After min max Scaling: \n", scaled_x)
```

d.

```
from sklearn import preprocessing

min_max_scaler = preprocessing.MinMaxScaler(feature_range =(0, 1))

# Scaled feature

scaled_x= min_max_scaler.fit_transform(x)

print ("After min max Scaling: \n", scaled_x)
```

II. Z-score normalization

```
from sklearn.datasets import load_iris
```

```
from sklearn.preprocessing import StandardScaler
import pandas as pd
# Load Iris dataset
iris = load_iris()
iris_df = pd.DataFrame(data=iris.data, columns=iris.feature_names)
# Display the original dataset
print("Original Iris dataset:")
print(iris_df.head())
# Initialize StandardScaler for Z-score normalization
scaler = StandardScaler()
# Z-score normalize the dataset
normalized_data = scaler.fit_transform(iris_df)
# Convert the normalized data back to a DataFrame
normalized_df = pd.DataFrame(normalized_data, columns=iris.feature_names)
# Display the normalized dataset
print("\nZ-score Normalized Iris dataset:")
print(normalized_df.head())
```

Exercise 2 Data standardization and Normalization Questions

Load wine quality dataset and do the following

a. Min max scaling
b. z-score normalization
c. Robust scaling
d. log1p transformation
e. Comment whether these data transformation useful in this data with the help of histogram plots of the data before and after transformation

Exercise 3 Principal Component Analysis Practice

a. Download breast cancer dataset
b. Practice from https://www.datacamp.com/tutorial/principal-component-analysis-in-python

Exercise 4 Principal Component Analysis Question

Execute the face recognition case study using pca given in the below link and write one page abstract on the implementation details.
https://github.com/geekquad/Facial-Recognition-with-PCA/blob/master/Face_Recognition_PCA.ipynb

Exercise 5 Synthetic dataset generation practice

a. **Use Faker package**

```
import pandas as pd

fake = Faker()

# Generating synthetic user data

num_users = 100
```

```
data = {

    'Name': [fake.name() for _ in range(num_users)],

    'Email': [fake.email() for _ in range(num_users)],

    'Address': [fake.address().replace('\n', ', ') for _ in range(num_users)],

    'Phone Number': [fake.phone_number() for _ in range(num_users)],

    'Date of Birth': [fake.date_of_birth(minimum_age=18, maximum_age=90) for _ in range(num_users)]

}

# Creating a Pandas DataFrame

df = pd.DataFrame(data)

# Displaying the synthetic data


print(df.head())
```

b. Use Pydbgen

c. Use Mimesis

```
from mimesis import Person

#Generating Fake Person Data:

person = Person()

print(person.full_name())

print(person.email(domains=['example.com', 'test.com']))

print(person.age())


print(person.gender())
```

Exercise 6 Synthetic dataset generation questions

a. Generate a data set with the following information

```
data = {
    'Course ID': course_ids,
    'Course Name': course_names,
    'Instructor': course_instructors,
    'Category': course_categories,
```

'Course Duration (hours)': course_durations,
'Students Enrolled': students_enrolled,
'Price': prices,
'Student CGPA': student_cgpa,
'Student Marks (out of 100)': student_marks
}

b. Generate data frame with the following information

data = {

‘Property ID’: property_ids,

‘Property Type’: property_type,

‘Price’: prices,

‘Area’: areas,

‘Number of Bedrooms’: bedrooms,

‘Number of Bathrooms’: bathrooms,

‘Agent ID’: agent_ids,

‘Owner Name’: owners,

‘Date Listed’: dates_listed

}

c. Generate a time series data