NAME: SYED MUZZAMIL WASEEM

SID: 11067

CID: 113083

ASSIGNMENT # 03 (MACHINE LEARNING)

1. **Explain Neural Network (NN), the structure of a neural network. Basic definition and preliminaries.**

Neural Network and its Structure

A Neural Network is a core part of machine learning, especially in deep learning. It mimics how the brain works, inspired by neurons' communication. Imagine teaching a computer to tell cats from dogs in pictures that's where NNs come in.

i. **Basics and Detective Analogy:** Think of NNs as layers of detectives, each focused on a detail (node). They connect like phone calls, weighing clues (weights). Some clues are crucial, like spotting tails.

ii. **Smart Decisions and Layers:** Detectives in the input layer handle basics, passing info to the next. Hidden layers dive deeper. All work together for final decisions.

iii. **Activation and Choices:** Each detective decides cat or a dog. If confident (output above threshold), they tell the next layer. Otherwise, they're silent.

iv. **Learning via Examples:** Detectives start clueless. Showing pics and labeling trains them. They adjust weights based on feedback.

v. **Improvement Over Time:** Detectives get better by spotting patterns.

In short, neural networks are like detective teams with layers, connections, and learning, used to solve tasks like distinguishing cats from dogs in pictures.

2. **Compare Multi-layer perceptron and SLP.**

i. **Single-Layer Perceptron:**

Think of SLP as a basic neuron that can only manage straightforward tasks. It's designed for problems where two categories can be easily separated, like distinguishing between day and night based on brightness. Its output is like a light switch, either on (1) or off (0).

ii. Multi-Layer Perceptron:

MLP is a more advanced neuron. It's capable of handling complex tasks, like recognizing faces in images. It has hidden layers that process information in layers of abstraction, allowing it to capture intricate patterns. MLP learns by adjusting its internal settings through a process called backpropagation, much like a skilled detective refining their methods.

To sum it up, SLP is a basic decision-maker for simple problems, while MLP is a sophisticated thinker that excels in handling intricate tasks through its hidden layers and learning mechanisms.

3. Mention all the steps of PCA. Explain eigenvalues and eigenvectors.

PCA is a technique used to reduce the dimensionality of large datasets while retaining as much information as possible. It is a linear transformation technique that is widely used in machine learning for feature extraction and data compression.

The steps of PCA are as follows:

i. Standardize the data.

ii. Compute the covariance matrix.

iii. Compute the eigenvectors and eigenvalues of the covariance matrix.

iv. Sort the eigenvectors by decreasing eigenvalues.

v. Choose the first k eigenvectors.

vi. Transform the original data into the new k-dimensional space.

Eigenvalues and eigenvectors are important concepts in linear algebra that are used in PCA. Eigenvectors are vectors that do not change direction when a linear transformation is applied to them, while eigenvalues are scalars that represent how much an eigenvector is stretched or shrunk by a linear transformation.

4. Mention the characteristics of K means, SVM and PCA.

K-means Clustering

K-means clustering is an essential unsupervised machine learning algorithm used to group data points into clusters based on their similarity. This technique is known for its simplicity and efficiency, making it a popular choice in various fields such as data mining and image processing.

Characteristics:

i. Unsupervised Clustering:
   - K-means groups data without requiring prior class labels.
   - Data points are organized into clusters based on their likeness.

ii. Efficiency and Simplicity:
   - The algorithm is straightforward and computationally efficient.
   - Widely utilized in tasks like image processing and data analysis.

iii. Centroid-Based Approach:
   - Data points are assigned to clusters based on their proximity to cluster centroids.
   - Centroids are iteratively adjusted to enhance cluster quality.

iv. Initialization Impact:
   - Initial positions of cluster centroids can impact results.
   - Various techniques mitigate sensitivity to initializations.

Support Vector Machine (SVM)

SVM, or Support Vector Machine, stands as a versatile and powerful machine learning model with applications in linear and nonlinear classification, regression, and outlier detection.

Characteristics:

i. Versatile Learning Model:
   - SVM handles diverse tasks like classification, regression, and outlier detection.
   - It accommodates both linear and nonlinear data.

ii. Maximizing Margin:
   - SVM seeks hyperplanes with maximum margins between classes.
   - Enhanced margin aids generalization to unseen data.

iii. Kernel Trick for Nonlinearity:

- ❖ Kernel functions allow SVM to work in higher-dimensional spaces.
- ❖ This facilitates handling nonlinear patterns in data.

iv. Binary and Multi-Class Tasks:

- ❖ SVM is originally a binary classifier.
- ❖ Multi-class problems are tackled through strategies like One-vs-Rest or One-vs-One.

Principal Component Analysis (PCA)

Principal Component Analysis is a dimensionality reduction technique, vital for simplifying complex datasets while preserving essential information.

Characteristics:

i. Dimensionality Reduction:

- ❖ PCA reduces feature dimensions while retaining crucial data aspects.
- ❖ Applied to managing high-dimensional datasets.

ii. Orthogonal Transformation:

- ❖ Original features are transformed into uncorrelated principal components.
- ❖ Principal components are orthogonal, minimizing interdependence.

iii. Variance Maximization:

- ❖ PCA aims to maximize data variance among principal components.
- ❖ Components ranked by the variance they capture.

iv. Feature Interpretation:

- ❖ Principal components lack direct interpretation but represent significant patterns.

5. Implement K means clustering and PCA on any dataset from Kaggle and show the results.

K means clustering

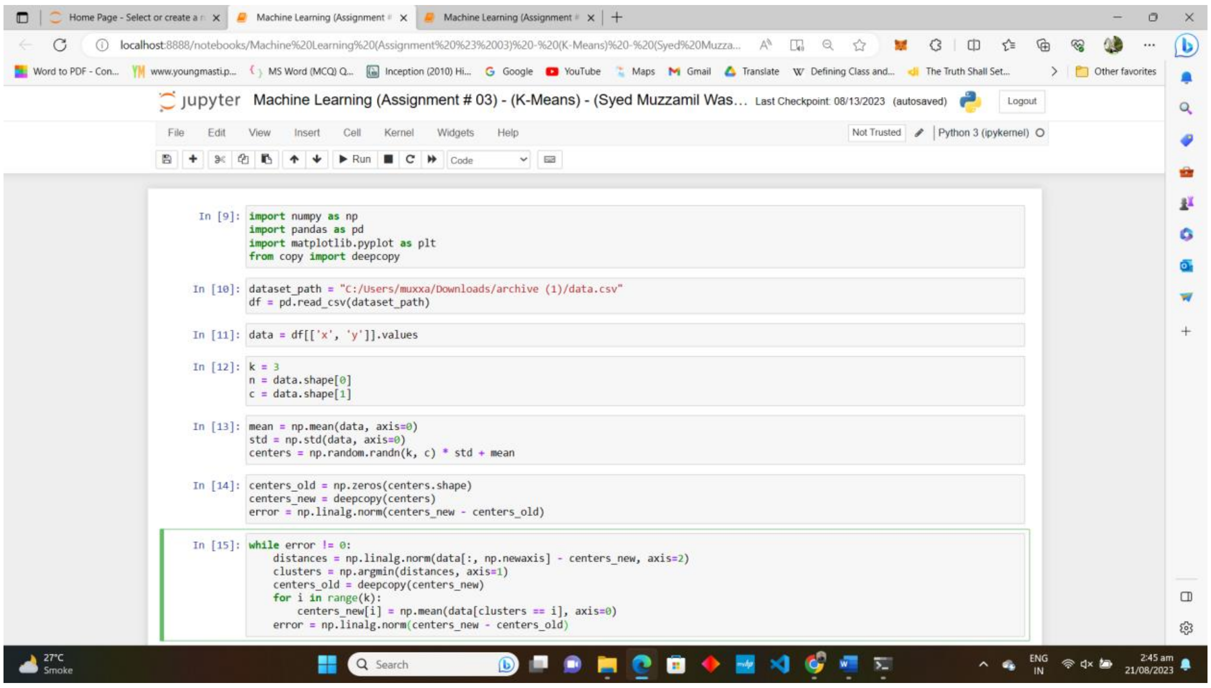
DATASET: 2D clustering data (https://www.kaggle.com/datasets/samuelcortinhas/2d-clustering-data)

CODE:

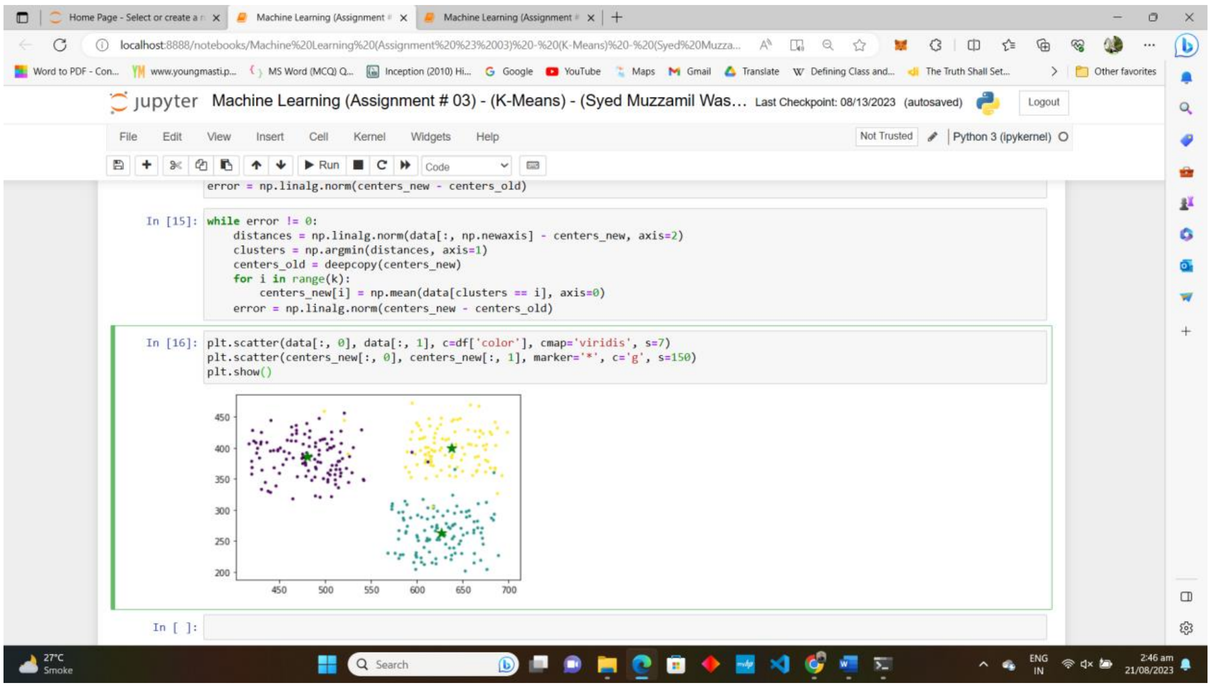
```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from copy import deepcopy

dataset_path = "C:/Users/muxxa/Downloads/archive (1)/data.csv"

df = pd.read_csv(dataset_path)

data = df[['x', 'y']].values

k = 3

n = data.shape[0]

c = data.shape[1]

mean = np.mean(data, axis=0)

std = np.std(data, axis=0)

centers = np.random.randn(k, c) * std + mean

centers_old = np.zeros(centers.shape)

centers_new = deepcopy(centers)

error = np.linalg.norm(centers_new - centers_old)

while error != 0:

    distances = np.linalg.norm(data[:, np.newaxis] - centers_new, axis=2)

    clusters = np.argmin(distances, axis=1)

    centers_old = deepcopy(centers_new)

    for i in range(k):

        centers_new[i] = np.mean(data[clusters == i], axis=0)

    error = np.linalg.norm(centers_new - centers_old)

plt.scatter(data[:, 0], data[:, 1], c=df['color'], cmap='viridis', s=7)

plt.scatter(centers_new[:, 0], centers_new[:, 1], marker='*', c='g', s=150)

plt.show()

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from copy import deepcopy

dataset_path = "C:/Users/muxxa/Downloads/archive (1)/data.csv"

df = pd.read_csv(dataset_path)
```

```
data = df[['x', 'y']].values

k = 3

n = data.shape[0]

c = data.shape[1]

mean = np.mean(data, axis=0)

std = np.std(data, axis=0)

centers = np.random.randn(k, c) * std + mean

centers_new = deepcopy(centers)

error = np.linalg.norm(centers_new - centers_old)

while error != 0:

    distances = np.linalg.norm(data[:, np.newaxis] - centers_new, axis=2)

    clusters = np.argmin(distances, axis=1)

    centers_old = deepcopy(centers_new)

    for i in range(k):

        centers_new[i] = np.mean(data[clusters == i], axis=0)

    error = np.linalg.norm(centers_new - centers_old)

plt.scatter(data[:, 0], data[:, 1], c=df['color'], cmap='viridis', s=7)

plt.scatter(centers_new[:, 0], centers_new[:, 1], marker='*', c='g', s=150)

plt.show()
```

CODE + OUTPUTS:


jupyter
Machine Learning (Assignment # 03) - (K-Means) - (Syed Muzzamil Was…
Last Checkpoint: 08/13/2023 (autosaved)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Not Trusted
Python 3 (ipykernel)
Run
Code

```
In [9]:  import numpy as np
         import pandas as pd
         import matplotlib.pyplot as plt
         from copy import deepcopy

In [10]: dataset_path = "C:/Users/muxxa/Downloads/archive (1)/data.csv"
         df = pd.read_csv(dataset_path)

In [11]: data = df[['x', 'y']].values

In [12]: k = 3
         n = data.shape[0]
         c = data.shape[1]

In [13]: mean = np.mean(data, axis=0)
         std = np.std(data, axis=0)
         centers = np.random.randn(k, c) * std + mean

In [14]: centers_old = np.zeros(centers.shape)
         centers_new = deepcopy(centers)
         error = np.linalg.norm(centers_new - centers_old)

In [15]: while error != 0:
             distances = np.linalg.norm(data[:, np.newaxis] - centers_new, axis=2)
             clusters = np.argmin(distances, axis=1)
             centers_old = deepcopy(centers_new)
             for i in range(k):
                 centers_new[i] = np.mean(data[clusters == i], axis=0)
             error = np.linalg.norm(centers_new - centers_old)
```


jupyter Machine Learning (Assignment # 03) - (K-Means) - (Syed Muzzamil Was... Last Checkpoint: 08/13/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
450
400
350
300
250
200
450 500 550 600 650 700

```
error = np.linalg.norm(centers_new - centers_old)
```

```
In [15]: while error != 0:
             distances = np.linalg.norm(data[:, np.newaxis] - centers_new, axis=2)
             clusters = np.argmin(distances, axis=1)
             centers_old = deepcopy(centers_new)
             for i in range(k):
                 centers_new[i] = np.mean(data[clusters == i], axis=0)
             error = np.linalg.norm(centers_new - centers_old)
```

```
In [16]: plt.scatter(data[:, 0], data[:, 1], c=df['color'], cmap='viridis', s=7)
         plt.scatter(centers_new[:, 0], centers_new[:, 1], marker='*', c='g', s=150)
         plt.show()
```

```
In [ ]:
```

PCA

CODE:

```
import pandas as pd

from sklearn.decomposition import PCA

import matplotlib.pyplot as plt

import warnings

warnings.filterwarnings('ignore')

df = pd.read_csv("C:/Users/muxxa/Downloads/archive (1)/data.csv")

X = df[['x', 'y']]

y = df['color']

X_std = (X - X.mean()) / X.std()

pca = PCA(n_components=2)

X_pca = pca.fit_transform(X_std)

pca_df = pd.DataFrame(data=X_pca, columns=['PC1', 'PC2'])

pca_df['color'] = y

explained_variance_ratio = pca.explained_variance_ratio_

plt.figure(figsize=(10, 6))

plt.scatter(pca_df['PC1'], pca_df['PC2'], c=pca_df['color'], cmap='viridis')

plt.xlabel('Principal Component 1')

plt.ylabel('Principal Component 2')
```

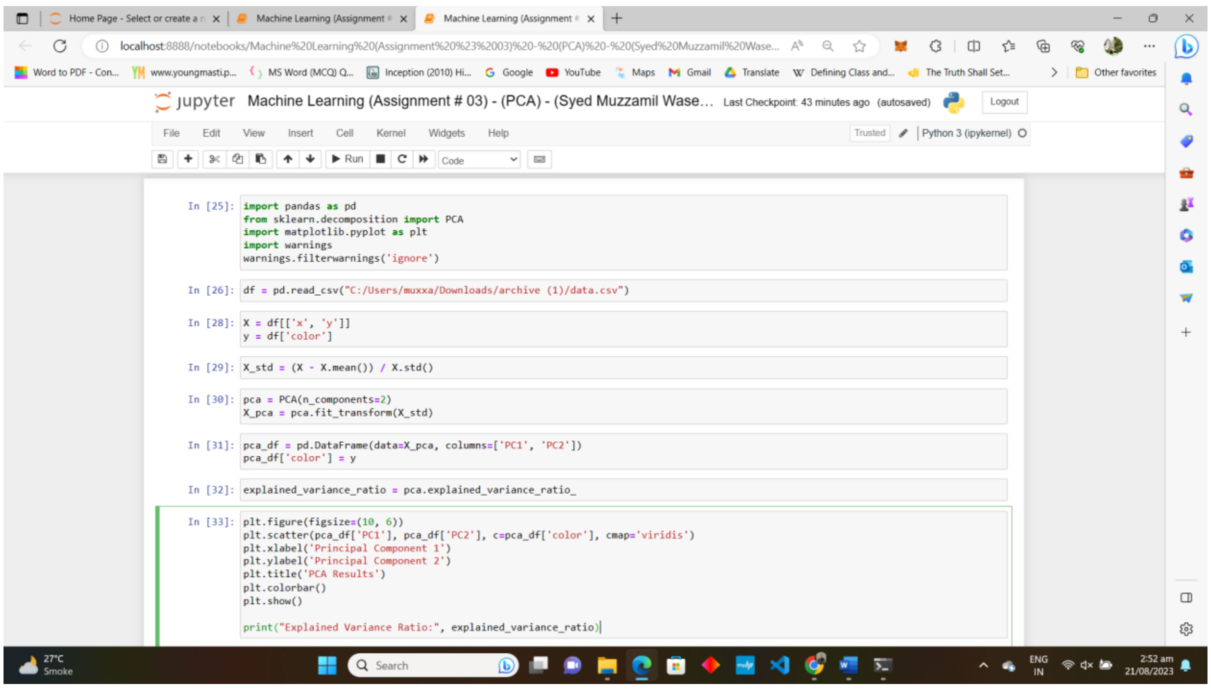
plt.title('PCA Results')

plt.colorbar()

plt.show()

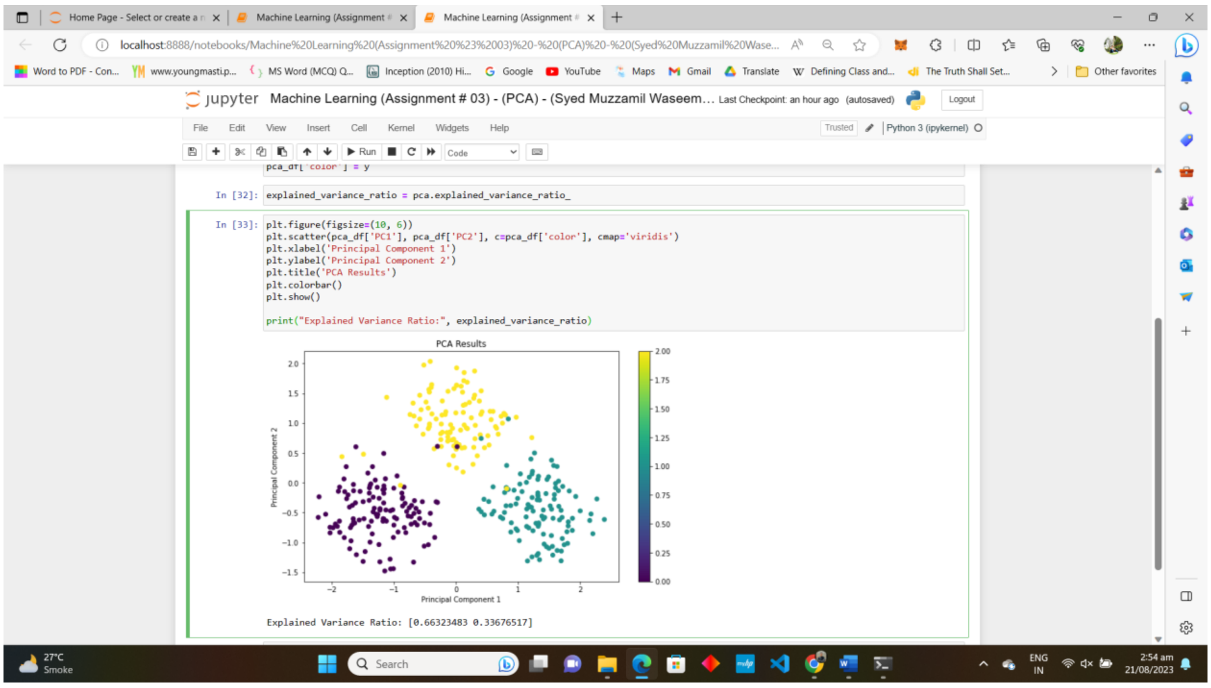
print("Explained Variance Ratio:", explained_variance_ratio)

CODE + OUTPUTS:



Machine Learning (Assignment # 03) - (PCA) - (Syed Muzzamil Wase... Last Checkpoint: 43 minutes ago (autosaved)

```
In [25]: import pandas as pd
         from sklearn.decomposition import PCA
         import matplotlib.pyplot as plt
         import warnings
         warnings.filterwarnings('ignore')

In [26]: df = pd.read_csv("C:/Users/muxxa/Downloads/archive (1)/data.csv")

In [28]: X = df[['x', 'y']]
         y = df['color']

In [29]: X_std = (X - X.mean()) / X.std()

In [30]: pca = PCA(n_components=2)
         X_pca = pca.fit_transform(X_std)

In [31]: pca_df = pd.DataFrame(data=X_pca, columns=['PC1', 'PC2'])
         pca_df['color'] = y

In [32]: explained_variance_ratio = pca.explained_variance_ratio_

In [33]: plt.figure(figsize=(10, 6))
         plt.scatter(pca_df['PC1'], pca_df['PC2'], c=pca_df['color'], cmap='viridis')
         plt.xlabel('Principal Component 1')
         plt.ylabel('Principal Component 2')
         plt.title('PCA Results')
         plt.colorbar()
         plt.show()

         print("Explained Variance Ratio:", explained_variance_ratio)
```



Machine Learning (Assignment # 03) - (PCA) - (Syed Muzzamil Waseem... Last Checkpoint: an hour ago (autosaved)

```
In [32]: explained_variance_ratio = pca.explained_variance_ratio_

In [33]: plt.figure(figsize=(10, 6))
         plt.scatter(pca_df['PC1'], pca_df['PC2'], c=pca_df['color'], cmap='viridis')
         plt.xlabel('Principal Component 1')
         plt.ylabel('Principal Component 2')
         plt.title('PCA Results')
         plt.colorbar()
         plt.show()

         print("Explained Variance Ratio:", explained_variance_ratio)
```

```
Explained Variance Ratio: [0.66323483 0.33676517]
```