**Arkansas Tech University**

**Graduate College, Information Technology**

**INFT 5603 Principles of Data Science, Fall 2023**

**Final / Due date: Monday, December 11th, 2023 before midnight. Total points: 100. Good luck!**

**1.** (*10 points*) Find the mean square error (MSE) for the following observation table.

|  |  |  |
| --- | --- | --- |
| Year | Actual output | Predicted output |
| 1 | 1 | 4 |
| 2 | 9 | 10 |
| 3 | 0 | -1 |
| 4 | -3 | -4 |

**A math problem with numbers and equations

Description automatically generated**

**2.** (*15 points*) Find the 2D convolution matrix using the following matrix *A* and the *template (mask)*.

*A* =

|  |  |  |  |
| --- | --- | --- | --- |
| -3 | 0 | 0 | 1 |
| 0 | 1 | 2 | 1 |
| 0 | 0 | 0 | 1 |
| -1 | 0 | 0 | 0 |

*Template (mask)* =

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 1 |
| 0 | 0 | 0 |
| 1 | 0 | 1 |

A white paper with black text and numbers

Description automatically generated

**3.** (*15 points*) Create the K = [7 -3 0 1 9 2 1] list in Python. Then, do the following actions.

Get the second element of the list.

Get the last three elements of the list.

Find the number of elements (size) of the list.

Insert 25 to end of the list.

Remove the last element of the list.

Concatenate the list K with the list L = [6 5 -2 9 0]

A screenshot of a computer

Description automatically generated

**4.** (*10 points*) Write a Python function to find average of two integer numbers in Python. Then, test the output

of the function by calling it and printing the result on the console (You do not need to take numbers from

end user).

A screenshot of a computer

Description automatically generated

**5.** (*10 points*) Find the confusion matrix of the following actual and predicted results/arrays in Python.

actual = [0, 1, 1, 1, 0, 0, 1, 1, 0, 1]

predicted = [0, 1, 0, 1, 0, 1, 0, 0, 1, 0]

A screenshot of a computer program

Description automatically generated

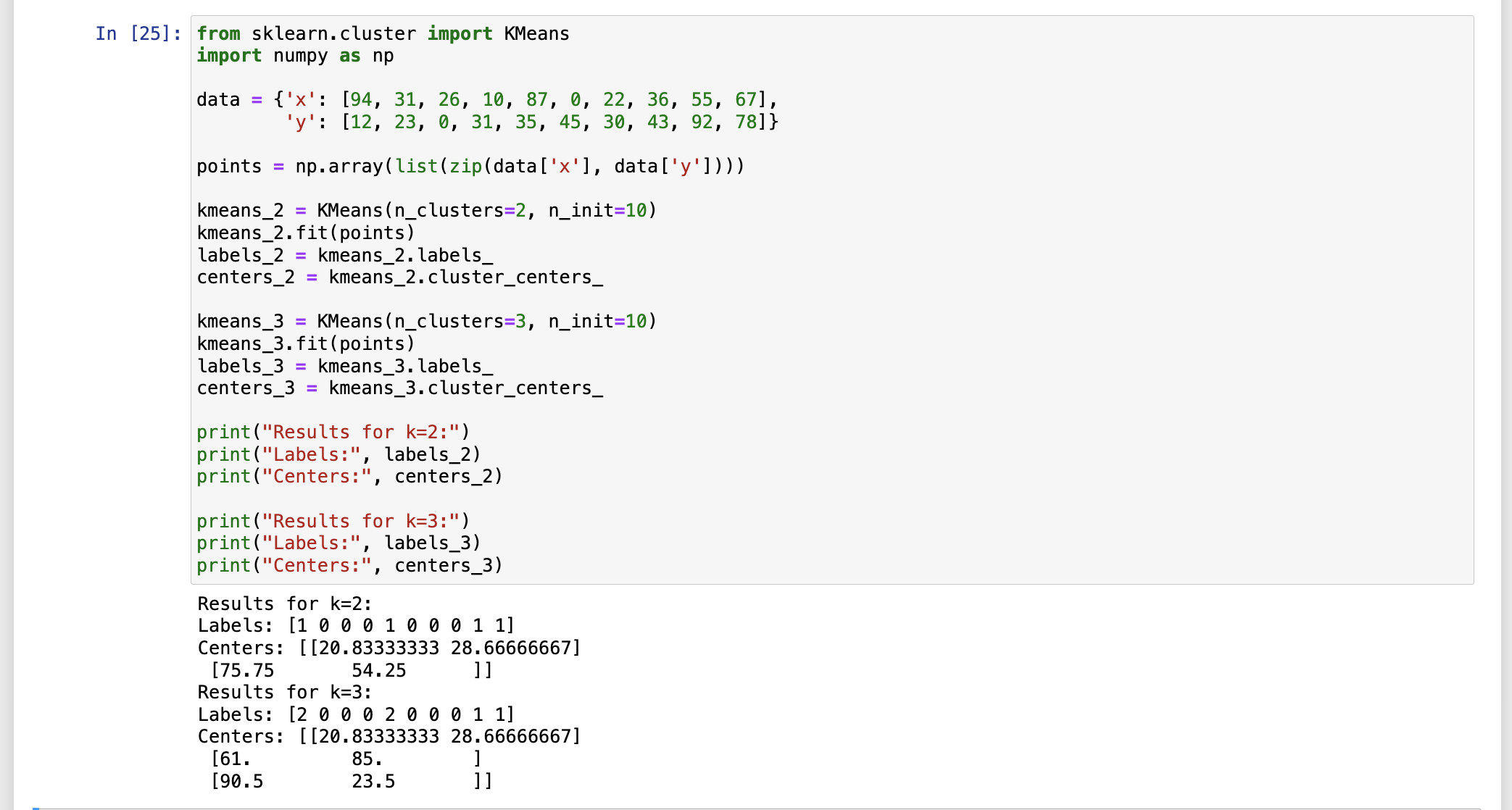
**6.** (*15 points*) Apply the K-means algorithm for the following data set in Python. Try the k values for k=2,

and k=3 and show the results.

Data = {‘x’: [94, 31, 26, 10, 87, 0, 22, 36, 55, 67],

‘y’: [12, 23, 0, 31, 35, 45, 30, 43, 92, 78]

}



**7.** (*15 points*) Design ANN using the Iris data set in Python. Set the hidden layers to 15, 20, and 15. Set the

max iteration number to 1,000. Then, predict the classes (labels) of the following new two tests using your

trained ANN.

test1 = [5.2, 4.9, 2.0, 5.3]

test2 = [4.8, 3.2, 5.7, 4.6]

*Note*: You can find more information about ScikitLearng library-based pre-defined MLPClassifier from

this link: <https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html>

A screenshot of a computer program

Description automatically generated

**8.** (*10 points*) What is an *Autoencoder*? Why do we need to use it?

An autoencoder is a type of artificial neural network used to learn efficient coding of unlabeled data, usually for the purpose of dimensionality reduction or feature learning. The basic idea of an autoencoder is to compress the input into a lower-dimensional code and then reconstruct the output from this representation to match the original input as closely as possible. This process forces the autoencoder to engage in what is effectively a form of lossy compression where the loss is measured by the difference between the original input and its reconstruction.

**Here is why we use autoencoders:**

* **Dimensionality Reduction**: Autoencoders can be used to reduce the dimensionality of the input data like Principal Component Analysis (PCA). This is useful for compressing the data and for visualizing high-dimensional data in 2D or 3D.
* **Feature Learning**: Autoencoders are good at learning representations (features) of the input data. These features can be useful for tasks like classification, where they can provide a set of more informative levels of abstraction than the raw data.
* **Denoising**: Denoising autoencoders are an extension of the basic autoencoder and are used to remove noise from data. They work by learning to ignore the "noise" in the data and recover the underlying clean data.
* **Data Compression**: When trained properly, autoencoders can learn to compress data in a way that is more specialized to the specifics of the data than standard compression algorithms.
* **Generative Models**: Variants like variational autoencoders (VAEs) can generate new data points that are like the input data. They can be used in generative tasks where new samples are needed, such as in image or text generation.
* **Anomaly Detection**: Autoencoders can be used to detect anomalies in data. If an autoencoder is trained on normal data, it will have a poor reconstruction of anomalies, which makes it possible to detect them.

**An autoencoder is composed of two parts:**

* **Encoder**: This part of the network compresses the input into a latent-space representation. It encodes the input data as a compressed representation in a reduced dimension.
* **Decoder**: This part aims to reconstruct the input data from the latent space representation. It decodes the encoded data back to an approximation of the original data.