|  |
| --- |
| Name: Group 54 |
| Student Reference Number: 10818465 1082055 , 10819477 |



|  |  |  |
| --- | --- | --- |
| Module Code: PUSL3123 | Module Name: AI and Machine Learning | |
| Coursework Title: Group coursework | | |
| Deadline Date: 3/01/2024 | | Member of staff responsible for coursework:  Dr Neamah Al-Naffakh |
| Programme: | | |
| Please note that University Academic Regulations are available under Rules and Regulations on the University website [www.plymouth.ac.uk/studenthandbook](http://www.plymouth.ac.uk/studenthandbook). | | |
| Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.  Randunu Kumara 10818465  Wishmi Wedagedara 10820855  Athukoralalage Athukorala 10819477  ***We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.***  Signed on behalf of the group: Randunu Kumara | | |
| Individual assignment: ***I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.***  Signed : | | |
| Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.  I \*have used/not used translation software.  If used, please state name of software………………………………………………………………… | | |
| **Overall mark \_\_\_\_\_% Assessors Initials \_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_** | | |

**Table of contents**

1. **Group members**
2. **Task 1**
   1. **Introduction**
   2. **Review**
   3. **Discussion**
   4. **Conclusion**
   5. **References and Bibliography**
3. **Task 2**
   1. **Task 2.1**
   2. **Task 2.2**
   3. **Task 2.3**
   4. **Task 2.4**

**Group Members**

|  |  |  |
| --- | --- | --- |
| Name | Index | Contribution |
| Randunu Kumara | 10818465 | * Literature review (Research paper 1,2,3,4) * Task 2.2 * Task 2.3 * Task 2.4 |
| Wishmi Wedagedara | 10820855 | * Literature review (Research paper 5,6,7) * Task 2.1 * Task 2.4 |
| Athukoralalage Athukorala | 10819477 | * Literature review (Research paper 7,8) * Task 2.1 * Task 2.4 |

**Task 1 - Literature Review**

**Introduction**

When it comes security and privacy in technological field securing and authenticating with biometric plays a vital role . out of the main biometric like fingerprint, face ,iris,gait,voice ,keystroke behavioral profiling . the selected topic for this specific literature review is face, facial recognition is the main bio metrics that has been using for nearly a decade in apple iphone devices.c Manufacuters Trusting on the facial recognition for the security and trust of their customers , it shows how the facial recognition plays a key role. Though it seems like its not that of a deep field , the depth of this fields depicts it real nature when discovering about its implementations and testing. For facial recognition there are lots of things needed like machine learning models, datasets for training and testing , classifiers , machine learning algorthims, matrices . and also accuracy of the systems and the error rates. Those are just some of the very essential things needed for a successful face recognition system. Those features will be discussed in the literature review later. Mainly the facial recognition can be divided in to two main parts

1. Face detection
2. Face recognition

Thse are two different scenarios . for the beginning of this review when considering about the history of facial recognition it goes back to around early 1960’s . back in those days techniques like Eigenfaces and principal component analysis were the main technologies. Those early innovators relied on extracting facial features from pictures , features like the distance between eyes , jawline shape. With the development of technologies those features were not enough to keep up with the latest. So as of that CNN also known as Convolution Neural Networks made the entrance in early 2010’s . those features could extract features like wrinkles ,subtle expressions other than geometry

When it comes to new research areas in face recognition feature extraction comes more often using this emotions and the key features of the face can be recognized and extracted. Also there are considerable amount of classification algorithms too . there are very capable classification algorithms like triplet loss function and Siamese network. Its not just enough by just implementing the system the accuracy of the system is also very important for that performance evaluation , standard matrices like accuracy, recognition rate , false acceptance rate. Another thing that has emerged recenty is masked face recognition, in 2020 as of the covid 19 pandemic as everyone is wearing a face mask its important to have face recognition models which identify the person even the mask is on there are research studies in this review also about masked face recognition

For developers who are going to implement face recognition to their application or systems . there are no of efficient and accurate face recognition api available . those models are already trained and using api endpoints those models can be used. Azure Face API , Google ML kit are some examples for that.

So in this review there are research papers about already existing datasets , new synthetic datasets, new face recognition models which are developed . new models to identify masked faces . researches about low quality image recognition etc. lets dive in to more details in the next chapter

**Literature Review**

In this review there will be a paragraph for each selected research paper and they will in order as in the references

In this research the main focus is about the history of face recognition technology . , it’s a biometric technology which is used for identifying people separately using facial features . in the beginning face geometry was the main focus . PCA and LDA was introduced between 1960s and 1990s. classifeirs like SVM,Adaboost , neural networks were invented during 1990s and 2010s .Deep learning technology has been in the action since 2010 .though these new technologies made huge difference to industry there were some challenges like pose , 3d modelling , expression changing .another thing was manual vs learned features .key technologies were PCA,LDA for facial feature extraction.SVM ,adaboost were the classifiers for face verification . when it comes evaluations benchmark datasets like LFW,CAS-PEAL-R1, megaface .accuaracy, rov curve auc measure discrimination ability were the matrices . through out this research paper it shows the development and the variation of the face recognition through out the years

when it comes to facial bio metrics it can be divided into two main parts face recognition and face detection .there are different types of developments, researches about these two methods . accuracies of different face recognition and detection techques were discussed in this research . Goofle and amazon are the leading companies which are researching for face recognition and detection. Datasets are the key thing to face recognition . in 2001 object detection using HAAR feature using cascade classifer was proposed. Its machine learning model which uses negative and positive images. Another important library is MTCNN which was written by github and ipacz. There are two methods for face detection in face recognition package hogg method and cnn method. There are perks and conns in these two while cnn method is more accurate but it take higher computational power . while hogg method is less accurate but its quick. Local Binary pattern (LBPH) also simple yet accurate method for facial recognition

in this research it shows accuracy about facial recognition techniques and introduction of a new dataset . the best face recognition models show a great accuracy of more than 99.8% on labeled faces in wild (LFW) dataset. To get such an accuracy these models were trained using large dataset which consists of millions of faces. In this research it discuss about a new dataset which has a it own rendering pipeline .this dataset has reduced the error rate by 52%. 256 per pixel were used to render these pictures. This syncface dataset has achieved 91.93 of accuracy and when it was tested with after mixing up with 2000 real pictures accuracy was 97.23%. with this new dataset synthetic to real domain gap has been clearly reduced error rate on LFW by 52.5% . it shows that this new data set is way better than GAN generated face to learn face recognition

This research is about a new open source framework called FaceX-Zoo .it provides various supervisory heads . also highly modular and scalable design is another feature in this framework. Another key feature is that a face sdk is provide with trained models for fully functional face validations this face sdk provides validation for masks faces to and allows to train face recognition networks with different backbones and supervisory heads. When it comes non masked face recognition face detection model was trained by retinal face on widerface dataset. This framework provides the solution for masked face recognition and semisiamese training for shallow face learning for the evaluation developers have synthesized a masked facial dataset based on MEGAFACE . for that FMA3D was used . also another important which has been implemented in this is they have given users a option of masked face recognition using 3D virtual mask adding technique using this feature they can train the module .from this new sdk , this can be used as the baseline for new developments .

Getting the statics about face recognition systems which was tested using small and large datasets. The mean absolute criterion was names as faster and simpler for face recognition . using a large dataset leads to low error rates. The research was done using ORL database with specific conditions. DET curve is used to evaluate the performance of the system. Classifier is a template matching method with a nearest neighbor . for large datasets Pin less mode is not accurate as its false matching rate is high . when using large datasets the results are more trustworthy as the error rates are low. The minimum size of the test dataset is N and estimating the N risk of α being wrong and error rate p does not exceed the estimated from the test set . there are two methods that a biometric system can operate those are identification and verification .AS the end conclusion of this study it shows that the MAD criterion Is faster and simpler for working.

The next research study is about a new face recognition technology GoogLeNet-M . as this is on the basis of streamlining the network. This improves the performance. Experiments shows that GooGleNet-M has the best performance with a recall rate of 0.97 and 0.98 of an accuracy. Also its concluded that out of other networks on the dataset googlenet-M has the best performance. In this the inception V2 layer has the batch normalization layer while the inception -V3 layer replaces the convolution kernel of two dimensions while the inception layer-4 comes with the idea of residual network . for this googlenet inception-v4 network has been used . googlenet using the channel shuffle is the improvement idea of this study. Here the nonlinearity of he network is increased as that leads to a meaningful depth of the network. Due to previous version’s activation function’s shortcomings those were improved continuously. This new technology has addressed the issue of high training accuracy and low test accuracy . with the adjustments to deep learning parameters learning rate will be high

In this research its about a research about synthetic data set and its experiments .with the controllable synthesis model without risking any privacy issues , non existing identities are taken for the datasets . the main focus of this study is observing the performance gap between the real images and synthetic image trained models there are two ways of narrowing the process effectively by enlarging the intra class variations and leveraging few real human faces for adaption of domain. The process can be briefly mentioned like this first the 10k identities were generated after the testing is finished there’s a gap of 88.98% vs 99.18% when the test is done on LFW and also syncface has more accuracy than the realface .these stats clearly depicts a considerable gap between synface and real face

When It comes facial recognition the according to the faces and identities MefaFace2 is the first and MS!M takes the second place large companies also use private datasets like Facenet by google, and the Facebook also trains their own models using 500M faces of 10M identities. With considerably large set of data training framework can be developed efficiently optimized. Also can be performed at a linear acceleration without dropping the performance. When it comes to earlier research studies three benchmarking tasks which are standard, masked, unbiased facial recognition are detailed including background, test sets and metrices those are compared with popular benchmarks and overlaps were removed .what’s difference in this is when it comes to masked face recognition performance of the algorithms are evaluated under three categories. For the metrices skewed error ration and standard deviation have been adopted. According to SER and STD scores training data reduces the biasness of recognition to a certain extent which surpasses the Webface4M and MS1MV2 .

In this study the main focus is about the lightning , facial expressions. With these parameters its bit hard for the trained models to make the recognition. When the quality of the image is low recognition become hardly possible depending on the degree. The experiment in this study is to prioritizing the hard samples for high quality images and easy sameples for low quality images. From this the loss function will be changed adoptively. First the images are introduced to data collection processes, from the difference in recognizability the problem is tackled using a feature norm as proxy for the image quality margin function is changed adoptively based on the feature gradient scale is assigned . then the efficiency of the proposed adaptive loss on various qualities of dataset and achieve low quality face datasets.

**Discussion**

In this chapter, the discussion about all the research papers which were selected will be mentioned in order of the references. In this chapter , the details about data collection, accuracy , efficiency and future trends will be discussed in each paragraph for each research paper

Though the development of face recognition systems is a huge impact to the technology fields but it also may raise privacy concerns . with this highly developed technologies it may lead to sexual violations . could even lead to judging someone’s race, cast ,sexual orientation . there must be some algorithms to avoid these types of problems too

In this study a dataset about Bollywood celebrities has been used after the implementation this was the result , TP=2747 FN=11 FP=63 TN=11 and the final conclusion of this study is best combination for face recognition and detection is Face-Recognition Module and MTCNN

WebFace260M is the one of dataset which has been used in this. With the experiment testing using 120k real images they have achieved 99.33% accuracy on LFW and 93.61 on average across the five benchmarks.

The conclusion of this research is , a whole pipeline will be provided to modules by the facesdk. Other features that has been provided by the sdk are face detection, face landmark localization ,and face feature extraction . the dataset which has been used in this study is widerface dataset,MegaFace

Dataset for this research is Feret which is a larger dataset. Here the DET curve gives uniform treatment to both types of error, and uses a logarithm scale for both axes, which spreads out the plot and better distinguishes different well performing systems and usually produces plots that are close to linear. Also the multi layer perceptron can perform as a classifier

In this study it describes about the application of deep learning models in facial recognition. It improves the Googlenet and then to improve the grouping convolution method under multi gpu applications. The future step from this point will be testing the generalization ability of the network model using a larger dataset.

According to this research study CASIA-WebFace and LFW dataset has been employed for training and testing. And also a synthetic version also generated. As of the conclusion , the main purpose was to explore the potentials of synthetic data . performance has been improved constantly when enlarging the intra – clas variation of synthetic data. There’s a big impact on the performance from width and depth of the synthetic data.

For this research Megaface2 and mS!M datasets were used . as of the conclusion of this study . while the webFace260M dataset preparation , the main concern was about privacy issues . due to that fact all the images were collected from public internet sources for testing and training. This dataset consist of diverse set of birthdates , ages and poses. For the evaluation process Unbiased facial recognition was designed specially. As the end conclusion this model reduces failure rate significantly from 40%.

In this research study for training datasets MS1MV2 , MS1mV3 and webface4m were used. Those datasets consists of millions of data each. the testing has been done on 9 different datasets. Datasets have been divided into to three groups High quality, Mixed Quality and low quality and for the training process dataset has been cropped and aligned faces with five different landmarks. Since the trained method is expected be trained better with the unidentifiable data three on-the-fly augmentations have been introduced which are cropping,rescalling and photometric jittering as of the end conclusion , the problem which was addressed in this research is unidentifiable pictures in datasets. as a solution problem was approached in two ways first one is using feature norm as proxy for the image qyality and changing the margin function adaptively based on feature norm. when it comes to limitations the problem is here the loss function doesn’t give special treatment to mislabeled samples . as in the ending part it says the its better to always move on to new datasets.

**Conclusion**

In this chapter all the data collection, accuracy , testing details , social impact and the future implementation about the all the research papers are mentioned .

Through out the above chapters all the details about the 9 different researches about the facial recognition was discussed . beginning from the first research the first one is about the evolution of face recgnition through out the years. Then there’s the new synthetic and highly accurate dataset .the new pytorch tool which allows to train models. So as of the end of this literature review the end conclusion is face recognition is still evolving field which needs constant researches and updates about the latest technologies. And regarding the datasets and model training the more data more accuracy it can achieve

**Reference And Bibliography**

Lixiang Li, X.M. (2020) *A review of Face Recognition Technology - IEEE Xplore*. Available at: https://ieeexplore.ieee.org/abstract/document/9145558/ (Accessed: 17 December 2023).

Muhammad Usama, Y.N.K., Ali Ahmad (2021) *[PDF] Face Recognition: A literature survey | semantic scholar*. Available at: https://www.semanticscholar.org/paper/Face-recognition%3A-A-literature-survey-Zhao-Chellappa/28312c3a47c1be3a67365700744d3d6665b86f22 (Accessed: 17 December 2023).

Bae, G. *et al.* (2022) *Digiface-1M: 1 million digital face images for face recognition*, *arXiv.org*. Available at: https://arxiv.org/abs/2210.02579 (Accessed: 18 December 2023).

Yinglu Liu, , J.W. (2021) *FaceX-Zoo: A pytorch toolbox for face recognition*. Available at: https://www.semanticscholar.org/paper/FaceX-Zoo%3A-A-PyTorch-Toolbox-for-Face-Recognition-Wang-Liu/8956aac5a7c44067cc4801ce42fedfed5eb8f182 (Accessed: 17 December 2023).

J. Alcobé, M.F.-Z. (2022) *Face recognition with small and large size databases - semantic scholar*. Available at: https://www.semanticscholar.org/paper/Face-recognition-with-small-and-large-size-Alcob%C3%A9-Fa%C3%BAndez-Zanuy/de72d2563277eed2a9cee2b91b72dc3c462cf5fe (Accessed: 17 December 2023).

Zhigang Yu, Y.D. (2022) *[PDF] research on face recognition classification based on improved ...* Available at: https://www.semanticscholar.org/paper/Research-on-Face-Recognition-Classification-Based-Yu-Dong/5253f3442156e0728f097cfdafb43fa414f68bbb (Accessed: 17 December 2023).

Haibo Qiu, B.Y. (2021) *Synface: Face recognition with synthetic data - semantic scholar*. Available at: https://www.semanticscholar.org/paper/SynFace%3A-Face-Recognition-with-Synthetic-Data-Qiu-Yu/52acfaafe3e064df47bf168f588179425c0097c6 (Accessed: 17 December 2023).

Zheng Hua Zhu, G.H. (2022) *WebFace260M: A benchmark for million-scale deep face recognition*. Available at: https://www.semanticscholar.org/paper/WebFace260M%3A-A-Benchmark-for-Million-Scale-Deep-Zhu-Huang/975cd1cad1eaf664bc07dbfab14c3c60715038d6 (Accessed: 17 December 2023).

Minchul Kim, A.K.J., Xiaoming Liu (2022) *Adaface: Quality Adaptive Margin for face recognition - semantic scholar*. Available at: https://www.semanticscholar.org/paper/AdaFace%3A-Quality-Adaptive-Margin-for-Face-Kim-Jain/549572ffc0f5adb6640a8d8fc93c8d7e62008985 (Accessed: 17 December 2023).

**Task 2**

**Task 2.1**

a)

%fisherirs dataset is loaded

load fisheriris

b)

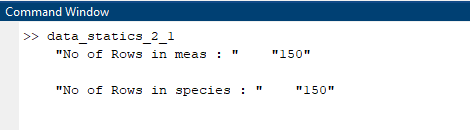
% part B

N = size(meas,1);

disp(["No of Rows in meas : ",num2str(N)]);

N2 = size(species,1);

disp(["No of Rows in species : ",num2str(N2)]);

****

c)

% part c

%disp(meas)

% to get the featurs from 1-4

%creating the list for attritbutes

atr = {"sepal length","sepal width","petal length","petal width"}

for x =1:4

%assigning values

meas\_data = meas(:,x);

meas\_mean = mean(meas\_data);

meas\_st\_deviation = std(meas\_data);

meas\_max = max(meas\_data);

meas\_min = min(meas\_data);

meas\_root\_sq = sqrt(mean(meas\_data.^2));

%displaying

%column

disp(["Attribute ",atr(x)]);

%mean display

disp(["mean : " , num2str(meas\_mean) ]);

%standard deviation

disp(["standard deviation : " , num2str(meas\_st\_deviation) ]);

%maximum

disp(["Maximum : " , num2str(meas\_max) ]);

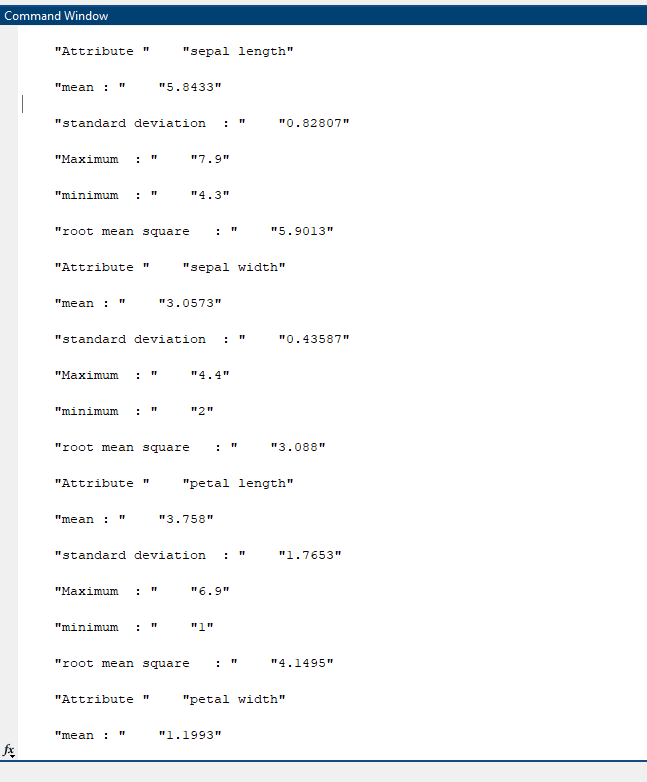
%minimum

disp(["minimum : " , num2str(meas\_min) ]);

%root mean square

disp(["root mean square : " , num2str(meas\_root\_sq) ]);

end



**Task 2.2**

**1)**

% Loading dataset

load fisheriris;

% Shuffling the dataset

rng("default");

indx = randperm(size(meas,1));

new\_meas = meas(indx,:);

new\_species = species(indx);

% Training percentage

trn\_p = 0.6;

% Spliting dataset into test and train

training\_count = floor(trn\_p \* size(new\_meas,1));

% Data

training\_data = new\_meas(1:training\_count,:);

testing\_data = new\_meas(training\_count+1:end,:);

% Target

training\_target = new\_species(1:training\_count);

testing\_target = new\_species(training\_count+1:end);

% Converting species labels to numerical values for the neural network

species\_labels = unique(new\_species);

num\_species = length(species\_labels);

new\_training\_targets = zeros(num\_species, length(training\_target));

for i = 1:length(training\_target)

species\_index = find(strcmp(species\_labels, training\_target{i}));

new\_training\_targets(species\_index, i) = 1;

end

%for testing target

new\_testing\_targets = zeros(num\_species,length(testing\_target));

for i = 1:length(testing\_target)

species\_index = find(strcmp(species\_labels, testing\_target{i}));

new\_testing\_targets(species\_index, i) = 1;

end

2)

%this is the code for 2 part

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%Creating the neural network

hidden\_layer\_size1 = 10;

nett = feedforwardnet(hidden\_layer\_size1);

% Training the neural network

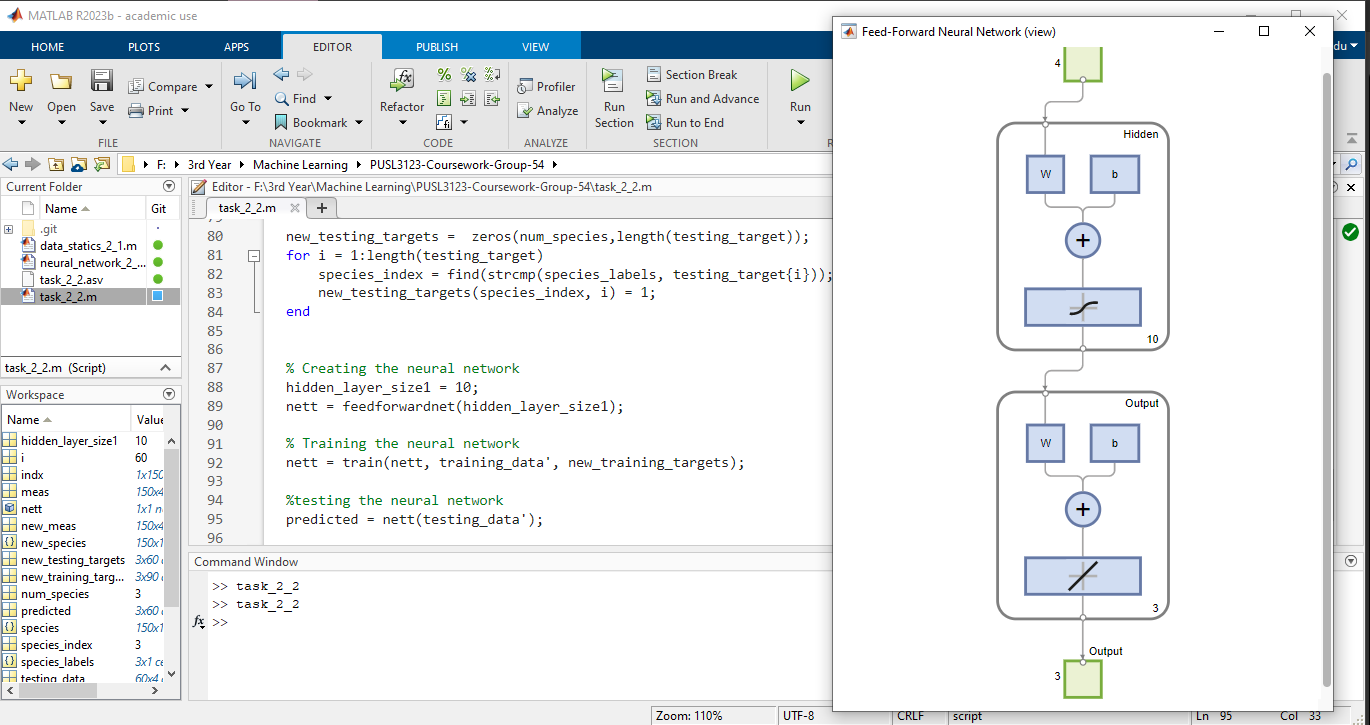
nett = train(nett, training\_data', new\_training\_targets);

%testing the neural network

predicted = nett(testing\_data');

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%end of the 2 part



3)

% prt 3

%defining different hidden layer sizes

hidden\_layers = [10,15,20];

%defining no of exprmnt times

no\_exps = 3;

%code for the part 3

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

for layer\_size = hidden\_layers

for exp = 1:no\_exps

%nn

nett =feedforwardnet(layer\_size);

%training

nett = train(nett,training\_data',new\_training\_targets);

%testing nn

predicted = nett(testing\_data');

%displaying

view(nett);

end

end

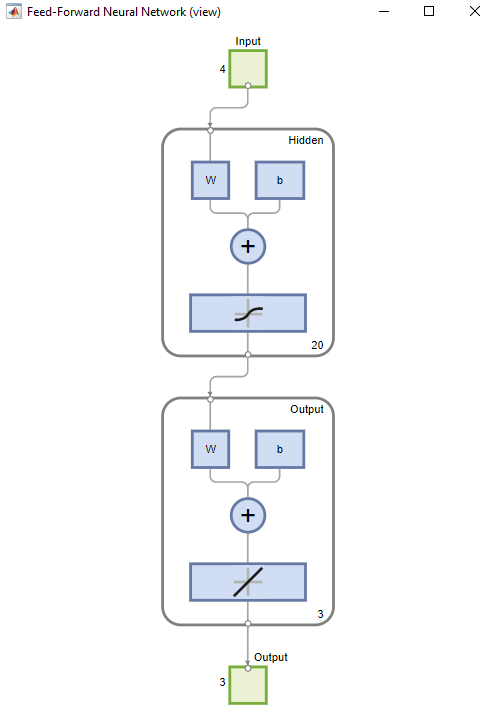
%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%end of the part 3

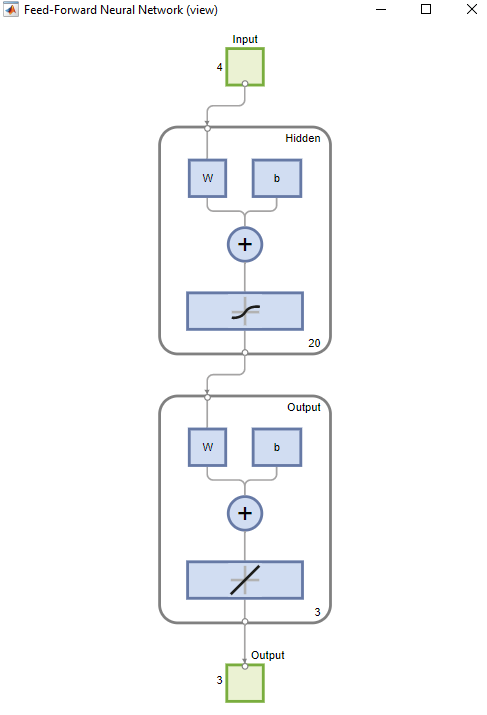
A screenshot of a computer

Description automatically generatedA diagram of a diagram

Description automatically generated



4)



5)

%part 5

no\_runs = 5;

%to get the accuaracy

accrcy = zeros(length(hidden\_layers),no\_exps,no\_runs);

for layer\_indx = 1:length(hidden\_layers)

layer\_size = hidden\_layers(layer\_indx);

for exp = 1:no\_exps

for run = 1:no\_runs

%creating the nn

nett = feedforwardnet(layer\_size);

%traing the nn

nett = train(nett,training\_data',new\_training\_targets);

%testing the nn using testing data

predicted = nett(testing\_data');

%getting the percantage of correct classification

% correct\_prdctns = sum(strcmp(species\_labels(argmax(predicted)),testing\_target/length(testing\_target)));

% accrcy(layer\_indx,exp,run) = correct\_prdctns;

[~, predicted\_idx] = max(predicted);

predicted\_labels = species\_labels(predicted\_idx);

% Calculating accuracy by comparing predicted labels to actual testing labels

correct\_predictions = sum(strcmp(predicted\_labels, testing\_target)) / length(testing\_target);

accrcy(layer\_indx, exp, run) = correct\_predictions;

end

end

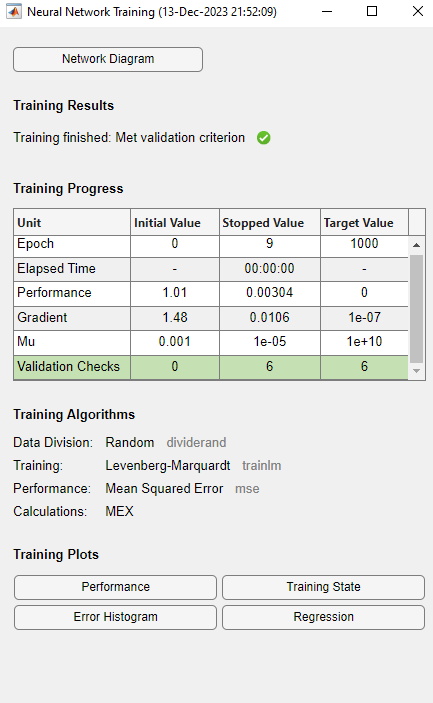
end

%calculating averge performance accros runs

average\_accuracy = mean(accrcy,3);

disp("average accuaracy =");

disp(average\_accuracy);



Full code for the task 2.2

% Loading dataset

load fisheriris;

% Shuffling the dataset

rng("default");

indx = randperm(size(meas,1));

new\_meas = meas(indx,:);

new\_species = species(indx);

% Training percentage

trn\_p = 0.6;

% Spliting dataset into test and train

training\_count = floor(trn\_p \* size(new\_meas,1));

% Data

training\_data = new\_meas(1:training\_count,:);

testing\_data = new\_meas(training\_count+1:end,:);

% Target

training\_target = new\_species(1:training\_count);

testing\_target = new\_species(training\_count+1:end);

% Converting species labels to numerical values for the neural network

species\_labels = unique(new\_species);

num\_species = length(species\_labels);

new\_training\_targets = zeros(num\_species, length(training\_target));

for i = 1:length(training\_target)

species\_index = find(strcmp(species\_labels, training\_target{i}));

new\_training\_targets(species\_index, i) = 1;

end

%for testing target

new\_testing\_targets = zeros(num\_species,length(testing\_target));

for i = 1:length(testing\_target)

species\_index = find(strcmp(species\_labels, testing\_target{i}));

new\_testing\_targets(species\_index, i) = 1;

end

%this is the code for 2 part

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%Creating the neural network

hidden\_layer\_size1 = 10;

nett = feedforwardnet(hidden\_layer\_size1);

% Training the neural network

nett = train(nett, training\_data', new\_training\_targets);

%testing the neural network

predicted = nett(testing\_data');

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%end of the 2 part

% prt 3

%defining different hidden layer sizes

hidden\_layers = [10,15,20];

%defining no of exprmnt times

no\_exps = 3;

%code for the part 3

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

for layer\_size = hidden\_layers

for exp = 1:no\_exps

%nn

nett =feedforwardnet(layer\_size);

%training

nett = train(nett,training\_data',new\_training\_targets);

%testing nn

predicted = nett(testing\_data');

%displaying

view(nett);

end

end

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%end of the part 3

%part 5

no\_runs = 5;

%to get the accuaracy

accrcy = zeros(length(hidden\_layers),no\_exps,no\_runs);

for layer\_indx = 1:length(hidden\_layers)

layer\_size = hidden\_layers(layer\_indx);

for exp = 1:no\_exps

for run = 1:no\_runs

%creating the nn

nett = feedforwardnet(layer\_size);

%traing the nn

nett = train(nett,training\_data',new\_training\_targets);

%testing the nn using testing data

predicted = nett(testing\_data');

%getting the percantage of correct classification

% correct\_prdctns = sum(strcmp(species\_labels(argmax(predicted)),testing\_target/length(testing\_target)));

% accrcy(layer\_indx,exp,run) = correct\_prdctns;

[~, predicted\_idx] = max(predicted);

predicted\_labels = species\_labels(predicted\_idx);

% Calculating accuracy by comparing predicted labels to actual testing labels

correct\_predictions = sum(strcmp(predicted\_labels, testing\_target)) / length(testing\_target);

accrcy(layer\_indx, exp, run) = correct\_predictions;

end

end

end

%calculating averge performance accros runs

average\_accuracy = mean(accrcy,3);

disp("average accuaracy =");

disp(average\_accuracy);

**Task 2.3**

1)

%loading kmeans

load kmeansdata;

2)

%assing data to a variable

new\_data = X;

%defining k values

k\_values = [3,4,5];

%part 2 begins

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% %for evaluating matrices

evl\_mtrcs = zeros(length(k\_values),1);

for i =1:length(k\_values)

%k means clustering for every k value

k2 = k\_values(i);

[indx,c] = kmeans(new\_data,k2);

%evaluating the clutering perfomance

evl\_mtrcs(i) = sum(sum(new\_data-c(indx,:)));

%visualizing

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% figure;

% gscatter(new\_data(:,1),new\_data(:,2),indx);

% hold on;

% %ploting centroids

% scatter(c(:,1),c(:,2),100,'k','filled');

% title("cluster results");

% legend('cluser1','cluster2','cluster3');

% hold off;

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if size(new\_data,2)<= 3

figure;

scatter3(new\_data(:,1),new\_data(:,2),new\_data(:,3),[],indx,'filled');

title(["k means clustering with k = ",num2str(k2)]);

xlabel('Feature 1');

ylabel('Feature 2');

zlabel('Feature 3');

drawnow;

end

end

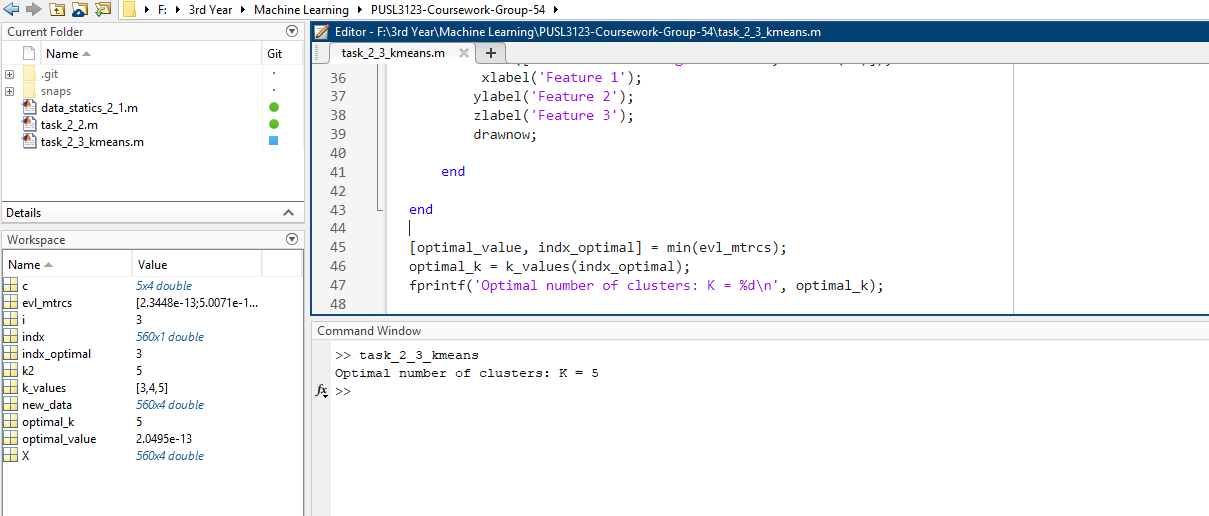
[optimal\_value, indx\_optimal] = min(evl\_mtrcs);

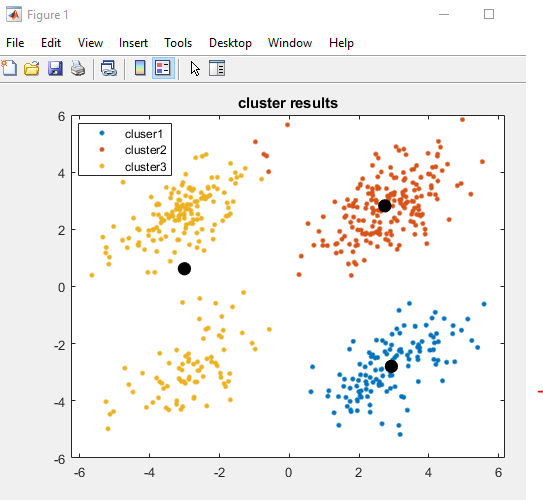
optimal\_k = k\_values(indx\_optimal);

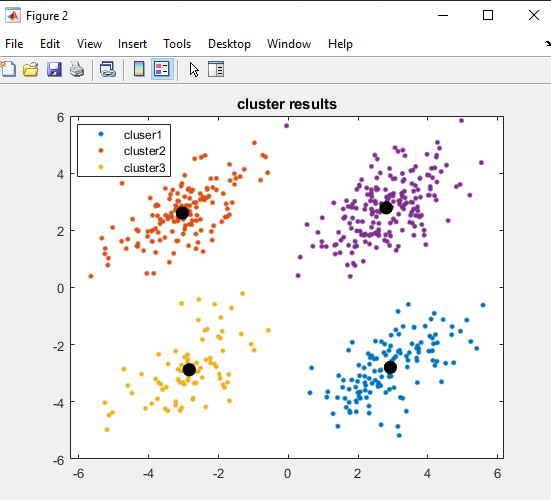
fprintf('Optimal number of clusters: K = %d\n', optimal\_k);

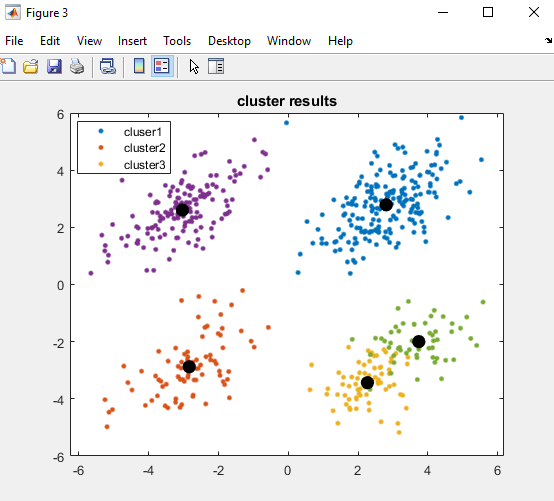
% part 2 end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%









3)

%part 3

%initializing variables for mean silhouette score

mean\_sil\_score = zeros(length(k\_values),1);

for i =1:length(k\_values)

%k means clustring for every k value

k2 = k\_values(i);

[indx,centroids] = kmeans(new\_data,k2);

%calculating sihoutte values

silht\_values = silhouette(new\_data,indx);

%calculating

mean\_sil\_score(i) = mean(silht\_values);

%ploting silht values for each cluster

figure;

silhouette(new\_data,indx);

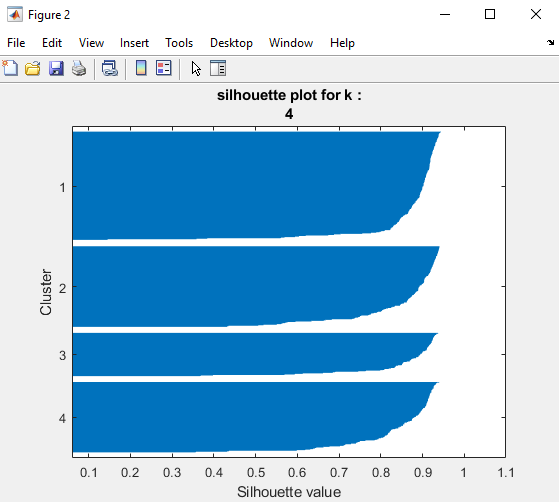
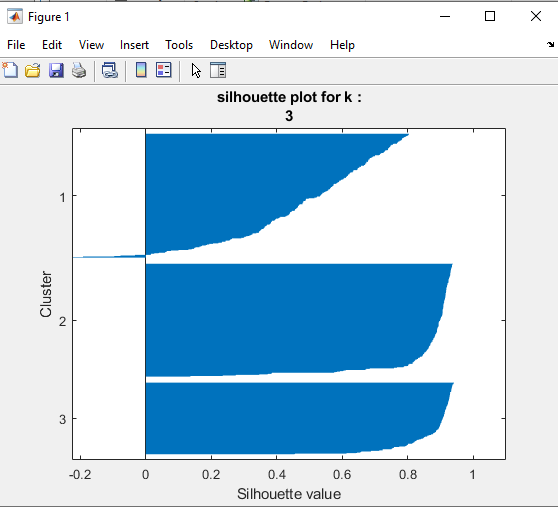
title (["silhouette plot for k :",num2str(k2)]);

xlabel('Silhouette value');

ylabel('Cluster');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%q3 end

A screen shot of a graph

Description automatically generated

4) when the data points are assigned to the closest cluster , kmeans clustering stops , as there no any improvements can be done . centroids stop from changing

%ploting clustrs

figure;

gscatter(new\_data(:, 1), new\_data(:, 2), indx);

hold on;

scatter(centroids(:, 1), centroids(:, 2), 50, 'k', 'filled');

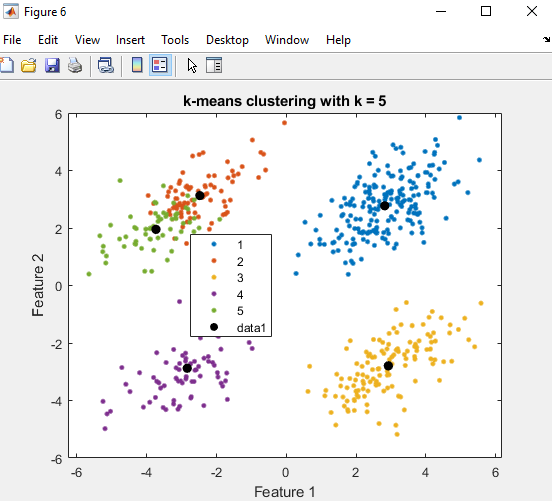
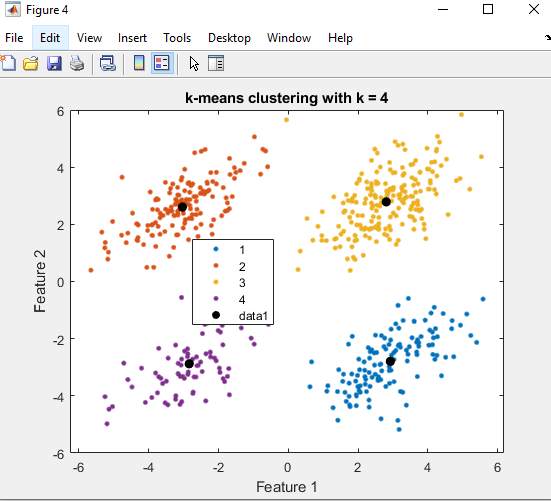
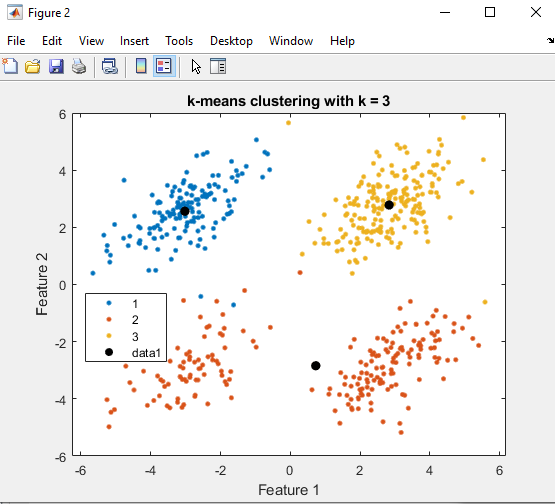
title(['k-means clustering with k = ', num2str(k2)]);

xlabel('Feature 1');

ylabel('Feature 2');

hold off;

end



5)

Best no of clusters is 4 as the highest mean silhouette score indicates the most appropriate no of clusters.

%q5

%to get the best no of clusters

[bst\_Scre,bst\_indx] =max(mean\_sil\_score);

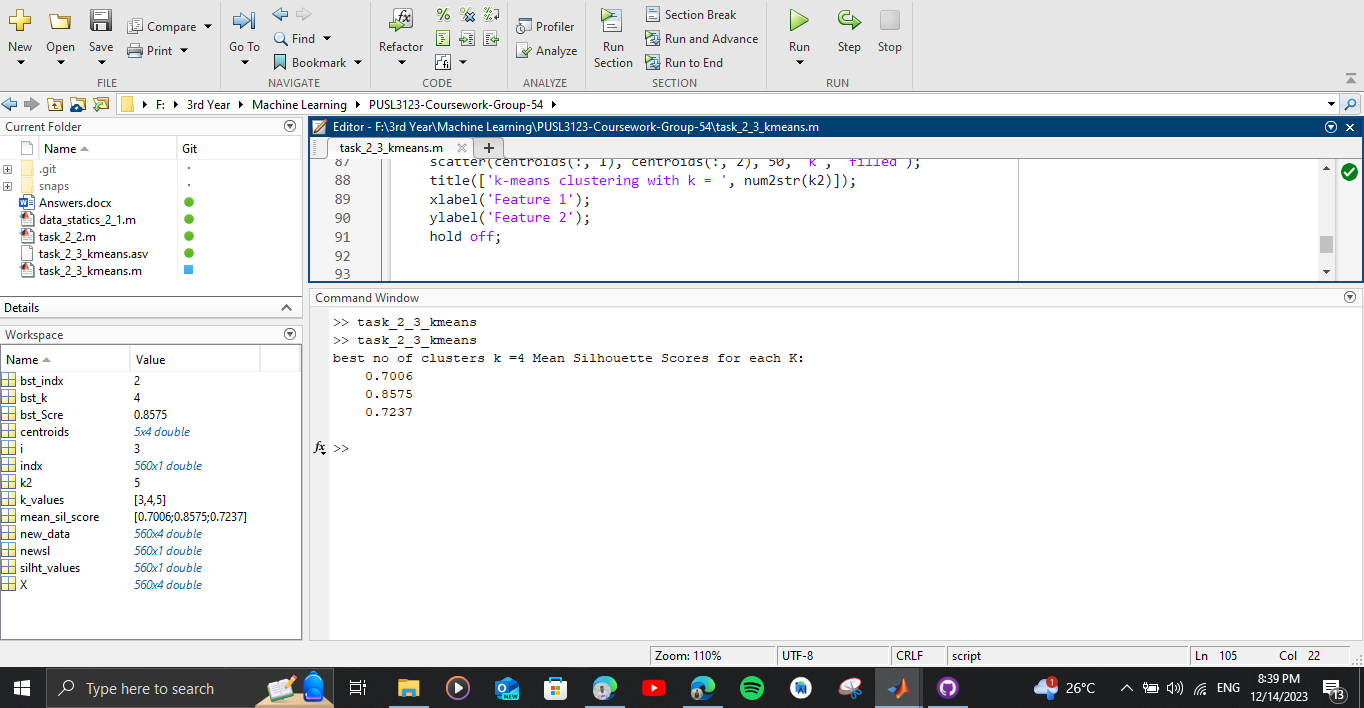
bst\_k = k\_values(bst\_indx);

%to get the output

fprintf("best no of clusters k =%d ",bst\_k);

disp('Mean Silhouette Scores for each K:');

disp(mean\_sil\_score);



6)

Depends on initial centroids. If the initial centroid assumption doesn’t reperesent data accurately final result wont be accurate

K need to be defined in the beginning as it wont work every time

Cant work with numeric data, need to be converted (eg : in fisheriris species )

Task 2.3 full code

%loading kmeans

load kmeansdata;

%assing data to a variable

new\_data = X;

%defining k values

k\_values = [3,4,5];

%part 2 begins

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% %for evaluating matrices

evl\_mtrcs = zeros(length(k\_values),1);

for i =1:length(k\_values)

%k means clustering for every k value

k2 = k\_values(i);

[indx,c] = kmeans(new\_data,k2);

%evaluating the clutering perfomance

evl\_mtrcs(i) = sum(sum(new\_data-c(indx,:)));

%visualizing

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% figure;

% gscatter(new\_data(:,1),new\_data(:,2),indx);

% hold on;

% %ploting centroids

% scatter(c(:,1),c(:,2),100,'k','filled');

% title("cluster results");

% legend('cluser1','cluster2','cluster3');

% hold off;

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if size(new\_data,2)<= 3

figure;

scatter3(new\_data(:,1),new\_data(:,2),new\_data(:,3),[],indx,'filled');

title(["k means clustering with k = ",num2str(k2)]);

xlabel('Feature 1');

ylabel('Feature 2');

zlabel('Feature 3');

drawnow;

end

end

[optimal\_value, indx\_optimal] = min(evl\_mtrcs);

optimal\_k = k\_values(indx\_optimal);

fprintf('Optimal number of clusters: K = %d\n', optimal\_k);

% part 2 end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%part 3

%initializing variables for mean silhouette score

mean\_sil\_score = zeros(length(k\_values),1);

for i =1:length(k\_values)

%k means clustring for every k value

k2 = k\_values(i);

[indx,centroids] = kmeans(new\_data,k2);

%calculating sihoutte values

silht\_values = silhouette(new\_data,indx);

%calculating

mean\_sil\_score(i) = mean(silht\_values);

%ploting silht values for each cluster

figure;

silhouette(new\_data,indx);

title (["silhouette plot for k :",num2str(k2)]);

xlabel('Silhouette value');

ylabel('Cluster');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%q3 end

%q4 begin

%ploting clustrs

figure;

gscatter(new\_data(:, 1), new\_data(:, 2), indx);

hold on;

scatter(centroids(:, 1), centroids(:, 2), 50, 'k', 'filled');

title(['k-means clustering with k = ', num2str(k2)]);

xlabel('Feature 1');

ylabel('Feature 2');

hold off;

end

%q5

%to get the best no of clusters

[bst\_Scre,bst\_indx] =max(mean\_sil\_score);

bst\_k = k\_values(bst\_indx);

%to get the output

fprintf("best no of clusters k =%d ",bst\_k);

disp('Mean Silhouette Scores for each K:');

disp(mean\_sil\_score);

**Task 2.4**

1)

%loading fisherirs

load fisheriris;

2)

%q1

%shuffleing the dataset

rng("default");

shfld\_indc = randperm(size(meas,1));

%split prcntg

trn\_p =0.6;

tst\_p =0.4;

total\_records = numel(shfld\_indc);

no\_trainining\_records = floor(trn\_p\*total\_records);

no\_testing\_records = total\_records - no\_trainining\_records;

%spltting dta

training\_data = meas(shfld\_indc(1:no\_trainining\_records),:);

testing\_data =meas(shfld\_indc(no\_trainining\_records+1:end),:);

training\_target = species(shfld\_indc(1:no\_trainining\_records));

testing\_target = species(shfld\_indc(no\_trainining\_records+1:end));

3)

%q3

%defining k values for evaluating

k\_values = [5,7];

for k = k\_values

%training the knn with k value

knn\_classifer = fitcknn(training\_data,training\_target,'NumNeighbors',k);

prdctd\_labels = predict(knn\_classifer,testing\_data);

%evaluating accuaracy

accrcy = sum(strcmp(prdctd\_labels,testing\_target))/numel(testing\_target);

%displaying the accuarcy for k value

fprintf("accuracy for k = %d : %.2f%%\n ",k,accrcy\*100);

%q4

%confussion matrix

cnfs\_mat = confusionmat(testing\_target,prdctd\_labels);

disp("Confusion matrix for k = " + k + ":");

disp(cnfs\_mat);

%prcntge of correct clssifications

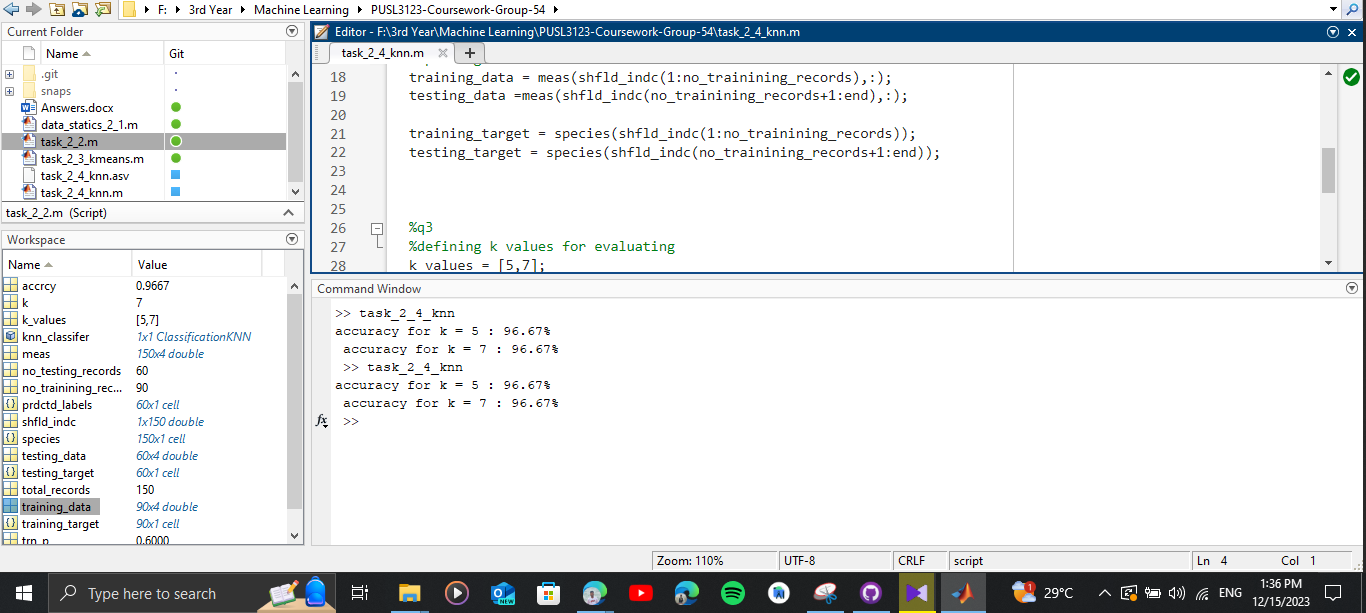
crct\_clssifctns = sum(diag(cnfs\_mat));

totl\_smpl = sum(cnfs\_mat(:));

prcntge = (crct\_clssifctns/totl\_smpl)\*100;

fprintf("Percentage of correct classifications for k = %d: %.2f%%\n\n", k, prcntge);

end



4)

%q3

%defining k values for evaluating

k\_values = [5,7];

for k = k\_values

%training the knn with k value

knn\_classifer = fitcknn(training\_data,training\_target,'NumNeighbors',k);

prdctd\_labels = predict(knn\_classifer,testing\_data);

%evaluating accuaracy

accrcy = sum(strcmp(prdctd\_labels,testing\_target))/numel(testing\_target);

%displaying the accuarcy for k value

fprintf("accuracy for k = %d : %.2f%%\n ",k,accrcy\*100);

%q4

%confussion matrix

cnfs\_mat = confusionmat(testing\_target,prdctd\_labels);

disp("Confusion matrix for k = " + k + ":");

disp(cnfs\_mat);

%prcntge of correct clssifications

crct\_clssifctns = sum(diag(cnfs\_mat));

totl\_smpl = sum(cnfs\_mat(:));

prcntge = (crct\_clssifctns/totl\_smpl)\*100;

fprintf("Percentage of correct classifications for k = %d: %.2f%%\n\n", k, prcntge);

end

A screenshot of a computer

Description automatically generated

5)

Requires a a considerable amount of time when executing

Needing k to be predefined . with this it may lead to issues with accuracy

Task 2.4 Full code

%loading fisherirs

load fisheriris;

%q1

%shuffleing the dataset

rng("default");

shfld\_indc = randperm(size(meas,1));

%split prcntg

trn\_p =0.6;

tst\_p =0.4;

total\_records = numel(shfld\_indc);

no\_trainining\_records = floor(trn\_p\*total\_records);

no\_testing\_records = total\_records - no\_trainining\_records;

%spltting dta

training\_data = meas(shfld\_indc(1:no\_trainining\_records),:);

testing\_data =meas(shfld\_indc(no\_trainining\_records+1:end),:);

training\_target = species(shfld\_indc(1:no\_trainining\_records));

testing\_target = species(shfld\_indc(no\_trainining\_records+1:end));

%q3

%defining k values for evaluating

k\_values = [5,7];

for k = k\_values

%training the knn with k value

knn\_classifer = fitcknn(training\_data,training\_target,'NumNeighbors',k);

prdctd\_labels = predict(knn\_classifer,testing\_data);

%evaluating accuaracy

accrcy = sum(strcmp(prdctd\_labels,testing\_target))/numel(testing\_target);

%displaying the accuarcy for k value

fprintf("accuracy for k = %d : %.2f%%\n ",k,accrcy\*100);

%q4

%confussion matrix

cnfs\_mat = confusionmat(testing\_target,prdctd\_labels);

disp("Confusion matrix for k = " + k + ":");

disp(cnfs\_mat);

%prcntge of correct clssifications

crct\_clssifctns = sum(diag(cnfs\_mat));

totl\_smpl = sum(cnfs\_mat(:));

prcntge = (crct\_clssifctns/totl\_smpl)\*100;

fprintf("Percentage of correct classifications for k = %d: %.2f%%\n\n", k, prcntge);

end

**Appendix**

task2\_1.m

%fisherirs dataset is loaded

load fisheriris

% part B

N = size(meas,1);

disp(["No of Rows in meas : ",num2str(N)]);

N2 = size(species,1);

disp(["No of Rows in species : ",num2str(N2)]);

% part c

%disp(meas)

% to get the featurs from 1-4

%creating the list for attritbutes

atr = {"sepal length","sepal width","petal length","petal width"}

for x =1:4

%assigning values

meas\_data = meas(:,x);

meas\_mean = mean(meas\_data);

meas\_st\_deviation = std(meas\_data);

meas\_max = max(meas\_data);

meas\_min = min(meas\_data);

meas\_root\_sq = sqrt(mean(meas\_data.^2));

%displaying

%column

disp(["Attribute ",atr(x)]);

%mean display

disp(["mean : " , num2str(meas\_mean) ]);

%standard deviation

disp(["standard deviation : " , num2str(meas\_st\_deviation) ]);

%maximum

disp(["Maximum : " , num2str(meas\_max) ]);

%minimum

disp(["minimum : " , num2str(meas\_min) ]);

%root mean square

disp(["root mean square : " , num2str(meas\_root\_sq) ]);

end

task2\_2.m

% Loading dataset

load fisheriris;

% Shuffling the dataset

rng("default");

indx = randperm(size(meas,1));

new\_meas = meas(indx,:);

new\_species = species(indx);

% Training percentage

trn\_p = 0.6;

% Spliting dataset into test and train

training\_count = floor(trn\_p \* size(new\_meas,1));

% Data

training\_data = new\_meas(1:training\_count,:);

testing\_data = new\_meas(training\_count+1:end,:);

% Target

training\_target = new\_species(1:training\_count);

testing\_target = new\_species(training\_count+1:end);

% Converting species labels to numerical values for the neural network

species\_labels = unique(new\_species);

num\_species = length(species\_labels);

new\_training\_targets = zeros(num\_species, length(training\_target));

for i = 1:length(training\_target)

species\_index = find(strcmp(species\_labels, training\_target{i}));

new\_training\_targets(species\_index, i) = 1;

end

%for testing target

new\_testing\_targets = zeros(num\_species,length(testing\_target));

for i = 1:length(testing\_target)

species\_index = find(strcmp(species\_labels, testing\_target{i}));

new\_testing\_targets(species\_index, i) = 1;

end

%this is the code for 2 part

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

%Creating the neural network

hidden\_layer\_size1 = 10;

nett = feedforwardnet(hidden\_layer\_size1);

% Training the neural network

nett = train(nett, training\_data', new\_training\_targets);

%testing the neural network

predicted = nett(testing\_data');

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%end of the 2 part

% prt 3

%defining different hidden layer sizes

hidden\_layers = [10,15,20];

%defining no of exprmnt times

no\_exps = 3;

%code for the part 3

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

for layer\_size = hidden\_layers

for exp = 1:no\_exps

%nn

nett =feedforwardnet(layer\_size);

%training

nett = train(nett,training\_data',new\_training\_targets);

%testing nn

predicted = nett(testing\_data');

%displaying

view(nett);

end

end

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%end of the part 3

%part 5

no\_runs = 5;

%to get the accuaracy

accrcy = zeros(length(hidden\_layers),no\_exps,no\_runs);

for layer\_indx = 1:length(hidden\_layers)

layer\_size = hidden\_layers(layer\_indx);

for exp = 1:no\_exps

for run = 1:no\_runs

%creating the nn

nett = feedforwardnet(layer\_size);

%traing the nn

nett = train(nett,training\_data',new\_training\_targets);

%testing the nn using testing data

predicted = nett(testing\_data');

%getting the percantage of correct classification

% correct\_prdctns = sum(strcmp(species\_labels(argmax(predicted)),testing\_target/length(testing\_target)));

% accrcy(layer\_indx,exp,run) = correct\_prdctns;

[~, predicted\_idx] = max(predicted);

predicted\_labels = species\_labels(predicted\_idx);

% Calculating accuracy by comparing predicted labels to actual testing labels

correct\_predictions = sum(strcmp(predicted\_labels, testing\_target)) / length(testing\_target);

accrcy(layer\_indx, exp, run) = correct\_predictions;

end

end

end

%calculating averge performance accros runs

average\_accuracy = mean(accrcy,3);

disp("average accuaracy =");

disp(average\_accuracy);

task2\_3.m

%loading kmeans

load kmeansdata;

%assing data to a variable

new\_data = X;

%defining k values

k\_values = [3,4,5];

%part 2 begins

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% %for evaluating matrices

evl\_mtrcs = zeros(length(k\_values),1);

for i =1:length(k\_values)

%k means clustering for every k value

k2 = k\_values(i);

[indx,c] = kmeans(new\_data,k2);

%evaluating the clutering perfomance

evl\_mtrcs(i) = sum(sum(new\_data-c(indx,:)));

%visualizing

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% figure;

% gscatter(new\_data(:,1),new\_data(:,2),indx);

% hold on;

% %ploting centroids

% scatter(c(:,1),c(:,2),100,'k','filled');

% title("cluster results");

% legend('cluser1','cluster2','cluster3');

% hold off;

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if size(new\_data,2)<= 3

figure;

scatter3(new\_data(:,1),new\_data(:,2),new\_data(:,3),[],indx,'filled');

title(["k means clustering with k = ",num2str(k2)]);

xlabel('Feature 1');

ylabel('Feature 2');

zlabel('Feature 3');

drawnow;

end

end

[optimal\_value, indx\_optimal] = min(evl\_mtrcs);

optimal\_k = k\_values(indx\_optimal);

fprintf('Optimal number of clusters: K = %d\n', optimal\_k);

% part 2 end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%part 3

%initializing variables for mean silhouette score

mean\_sil\_score = zeros(length(k\_values),1);

for i =1:length(k\_values)

%k means clustring for every k value

k2 = k\_values(i);

[indx,centroids] = kmeans(new\_data,k2);

%calculating sihoutte values

silht\_values = silhouette(new\_data,indx);

%calculating

mean\_sil\_score(i) = mean(silht\_values);

%ploting silht values for each cluster

figure;

silhouette(new\_data,indx);

title (["silhouette plot for k :",num2str(k2)]);

xlabel('Silhouette value');

ylabel('Cluster');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%q3 end

%q4 begin

%ploting clustrs

figure;

gscatter(new\_data(:, 1), new\_data(:, 2), indx);

hold on;

scatter(centroids(:, 1), centroids(:, 2), 50, 'k', 'filled');

title(['k-means clustering with k = ', num2str(k2)]);

xlabel('Feature 1');

ylabel('Feature 2');

hold off;

end

%q5

%to get the best no of clusters

[bst\_Scre,bst\_indx] =max(mean\_sil\_score);

bst\_k = k\_values(bst\_indx);

%to get the output

fprintf("best no of clusters k =%d ",bst\_k);

disp('Mean Silhouette Scores for each K:');

disp(mean\_sil\_score);

task2\_4.m

%loading fisherirs

load fisheriris;

%q1

%shuffleing the dataset

rng("default");

shfld\_indc = randperm(size(meas,1));

%split prcntg

trn\_p =0.6;

tst\_p =0.4;

total\_records = numel(shfld\_indc);

no\_trainining\_records = floor(trn\_p\*total\_records);

no\_testing\_records = total\_records - no\_trainining\_records;

%spltting dta

training\_data = meas(shfld\_indc(1:no\_trainining\_records),:);

testing\_data =meas(shfld\_indc(no\_trainining\_records+1:end),:);

training\_target = species(shfld\_indc(1:no\_trainining\_records));

testing\_target = species(shfld\_indc(no\_trainining\_records+1:end));

%q3

%defining k values for evaluating

k\_values = [5,7];

for k = k\_values

%training the knn with k value

knn\_classifer = fitcknn(training\_data,training\_target,'NumNeighbors',k);

prdctd\_labels = predict(knn\_classifer,testing\_data);

%evaluating accuaracy

accrcy = sum(strcmp(prdctd\_labels,testing\_target))/numel(testing\_target);

%displaying the accuarcy for k value

fprintf("accuracy for k = %d : %.2f%%\n ",k,accrcy\*100);

%q4

%confussion matrix

cnfs\_mat = confusionmat(testing\_target,prdctd\_labels);

disp("Confusion matrix for k = " + k + ":");

disp(cnfs\_mat);

%prcntge of correct clssifications

crct\_clssifctns = sum(diag(cnfs\_mat));

totl\_smpl = sum(cnfs\_mat(:));

prcntge = (crct\_clssifctns/totl\_smpl)\*100;

fprintf("Percentage of correct classifications for k = %d: %.2f%%\n\n", k, prcntge);

end