In this assignment we are going to apply the work on digits “one” and “seven”, which we have been doing in the recent lectures, to the digits “two” and “four”.

Q1. Create models of digits “two” and “four”

You can use three strokes to model each digit.

You can choose the parameters of each stroke by generating random values using `runif()`.

Please experiment with different ranges for `runif()`. How did you choose those ranges?

Please try to change the ranges and describe what effect it has on the generated digits.

In order to choose the ranges, you may find it useful to know the end point of a stroke. You can find this using these equations

```
xe = x – len*sin(theta)

ye = y + len*cos(theta)
```

where xe and ye are the coordinates of the end point, x and y are the coordinates of the start point, len is the length and theta is the angle of the stroke.

Q2. Fit your models to the real digits

Generate 10,000 random fours and fit them to the real fours using `findnns()`.

Display the images of some of the real fours alongside their best match among the random fours.

Please include some example images in your report. Show some examples of good matches and bad matches (if any).

How good were the matches? Were there any cases where the model did not fit well to a real four? Did you need to change the ranges of `runif()` in order to improve the matches? Describe any changes you had to make and why.

Is the three-stroke model adequate to represent all the fours? Does it need to be modified? Please explain your answer.

If you think you can improve the model (e.g. by adding an extra stroke) then please try it out and see if it gives better matches.

Repeat the above steps for the twos.

Q3. Fit your models to the other digits

Now please fit the random twos to the real fours and vice versa.

Are there cases where a random two fits better to a real four than a random four or vice versa? Please display any images in your report. Please explain why the mismatch has occurred for these particular examples.

Compute the confusion matrix. How many errors are there? Can you explain why these errors occur? Can you suggest ways to modify the models to reduce the errors?

Please make any changes needed to the models and try to fit them again. Compute the confusion matrix. How have the errors between affected? Can you explain why?

Q5. Visualising the distributions in parameter values

Find the “goodfits” for the twos, i.e. the random twos whose distance from the nearest real two is less than a certain threshold. You can choose the threshold by looking at the histogram of distances. Also find the goodfits for the fours.

Display scatter plots of pairs of parameter values for the goodfits e.g. plot x1 against y1 or theta1 against theta2. Describe the distributions. Is there any evidence of correlation between different parameters? Are the distributions denser in some areas than others? Is there any evidence of points “piling up” at the limits? Can you explain why this might be? What conclusions can you draw? If you think that the limits on the parameters need to be changed, then please make the appropriate changes and see what effect they have.

Q6. Applying PCA to the parameter values

Apply PCA to the parameter values of the goodfits. Or, if you think there is no evidence of correlation between parameter values, you can use Diarmuid’s technique of fitting a Normal distribution to each parameter independently.

Generate random twos and fours using either PCA or Diarmuid’s technique. Classify the real twos and fours using your random twos and fours. What is the confusion matrix? What is the best value of k?

How many errors are there? Display digits which have been mismatched. Can you explain why these errors occur? Can you suggest ways to modify the models to reduce the errors?

Q7. Using different numbers of image-eigenvectors

Find the accuracy of the kNN algorithm with different numbers of image-eigenvectors, where the number n could range from 2 up to 256 in steps of 5 or 10. By “image-eigenvectors” I mean eigenvectors calculated from the images of the random digits, as we did with the real digits in the previous Assignment.

What is the shape of the curve? Does it increase steadily? Does it level off or does it have a peak? Can you explain the shape?

How does the execution time change as the number of eigenvectors changes? Is there an optimum number?

Q8. Classifying the ones, twos, fours and sevens together.

Using your random digits generated in Q6 along with random digits for the ones and sevens generated using the code from the lectures, now compute a confusion matrix for all four digits together.

How many errors are there? Display digits which have been mismatched. Can you explain why these errors occur? Can you suggest ways to modify the models to reduce the errors?