COMP2002 - Artificial Intelligence

Week 2 - Machine Learning Exercises

Introduction

The aim of this sheet of exercises is to get you started with implementing machine learning software. You should complete the exercises ahead of the week 3 seminar session. The model answers will be published shortly after, giving you enough time to re-attempt the exercises after the demonstration in the seminar. You should complete the exercises in a Jupyter Notebook. You can either install Jupyter on your own device or use the version available in the labs.

Activities

Your task is to go through the following tasks. Please note, you are expected to complete some work on this outside of the timetabled sessions.

Exercise 1 - Classify the Iris Data

In this week's lecture, you learned about the iris dataset, and this exercise requires you to build a classifier for that dataset. You should follow these steps. At the end of each step, run the cell to see the result.

First, **import the required modules** by adding the following to the first cell in the notebook:

```
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.datasets import load_iris
from sklearn.neighbors import KNeighborsClassifier
```

Then, load the Iris data. Insert a new cell and enter the following code:

```
# Load the Iris data.
data = load_iris()
inputs = data.data
targets = data.target
```

We need to reduce the dimensionality of the data using PCA to visualise it. In a new cell, enter the following:

```
# Project the data into two dimensions using PCA.
pca = PCA(n_components=2)
compressed = pca.fit_transform(inputs)
```

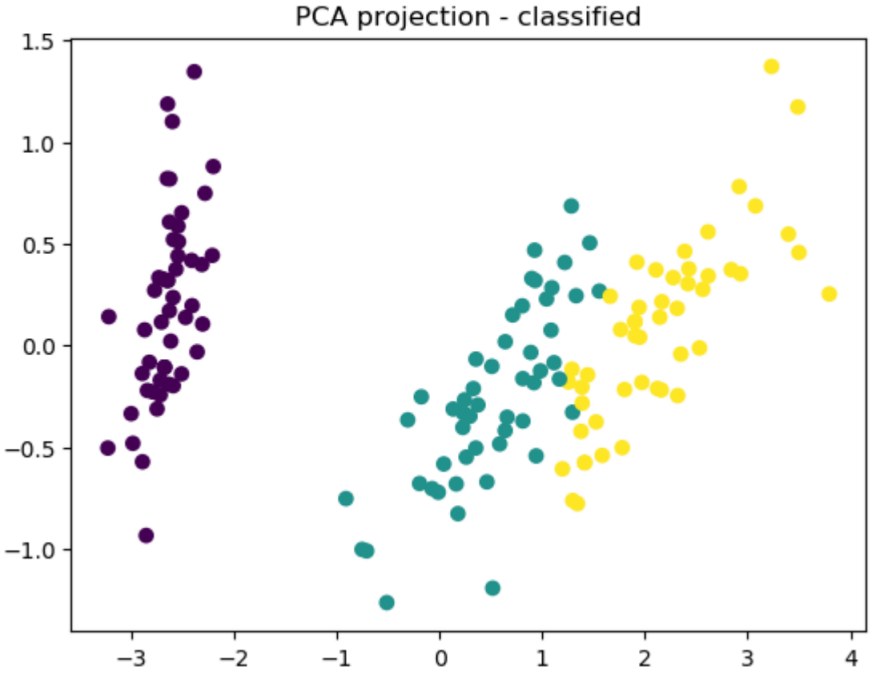
Next, train the classifier itself. Instantiate the KNearestNeighbors classifier (set it for 10 neighbours – you can try adjusting this to see what the effect is). Then fit the input data to the target values and use the predict method to see what the model prediction is for each of the 150 flowers.

```
# Train the classifier.
classifier - KNeighborsClassifier(n_neighbors=10)
classifier.fit(inputs,targets)
classifiedData = classifier.predict(inputs)
```

Finally, having obtained predictions for each flower, visualise the results. Produce a 2D scatter plot and colour the points according to its predicted value.

```
# Plot the results.
plt.figure()
plt.scatter(compressed[:,0], compressed[:,1],  c=classifiedData)
plt.title("PCA projection - classified")
plt.savefig("iris_pca_classified.png, bbox_inches="tight")
plt.show()
```

Your plot should look something like this:



Extension task: Write a loop to find the optimal value K. Display your results with an appropriate plot.

Exercise 2 - Loading Data with Pandas

You won't always have all of the data you need available in Scikit, so in this exercise, you'll see how to load data efficiently with the **Pandas** module. Pandas is a Python module that is designed to facilitate reading a wide variety of formats – in this exercise, you will load a CSV version of the Iris data and run it through your classifier.

To do this, replace the code loading the data with the following:

```
# Load the data using Panda. Print it to view it.
data = pandas.read_csv("iris.data")
print(data)
```

This loads the data. It's a good idea to print it so that you can check it's been read in properly. You may well find there are issues with the format or structure of the file, and it's good to spot it at this stage.

You then need to extract the data and targets; add the following:

```
# Extract the inputs.
inputs = data.values[:,:-1].astype(float)

# Extract the targets - convert to numerical values to help with
# colouring when we plot the results.
cls = ["Iris-setosa", "Iris-versicolor", "Iris-virginica"]
targets = [cls.index(c) for c in data.values[:,-1].astype(str)]
targets = np.array(targets)
```

The rest of the code should function as before.

Exercise 3 - Classifying Digit Images

Another example we looked at in the lecture was that of classifying written digits. In this exercise you'll use the skills you've learned in Exercise 1 and create a classifier for a dataset of written digits.
The data for this exercise comes in three parts: there is the data itself, some conveniently formatted images that you can use to see what each handwritten digit looks like, and a target value (0 to 9).
The majority of the code is the same as in Exercise 1 – you're loading some data, reducing its dimensionality with PCA and classifying it. There are two differences:

- You should plot some of the example images using Matplotlib's **imshow** method. See if you can add the target value as the figure title so that you can see which digit the image is showing (it's not always obvious!).
- You need to import the digit data rather than the iris data.

To import the data, you will need the following code. First, an import for the load_digits function:

```
from sklearn.datasets import load_digits
```

Second, the code that actually extracts the data and parses into three variables (data, images and targets):

```
# Load the data
digits = load_digits()
# Extract the three parts of the data.
data = digits.data
images = digits.images
target = digits.target
```