COMP60711 - Part 2 Coursework 2

Course Unit:	COMP60711: Data Engineering
Responsible Staff Member:	Professor John Keane
Marks:	This course is worth **25%** of the overall marks for this unit.
Submissions:	This is the **4th** of **4** assessed submissions.
Method of Submitting:	This notebook, after completion, should be saved as a HTML document and submitted using Blackboard
Deadline:	Thursday 4th November 9AM (UK time)
Late Submissions:	Extensions will only be granted as a result of formally processed Mitigating Circumstances. Marks for late submissions will be reduced in line with the University policy.

Please complete the questions in the spaces provided (under the "Answer" block for each question), then download the notebook in HTML format and submit to Blackboard.

Please also add your student ID and name below.

Student ID (7-8 digit number)	Full Name
12345678	Name Here

Reminders

- **Please make clear any assumptions and provide evidence to justify your answers**
- Jupyter notebooks use markdown. A brief summary of how to use markdown can be seen here. Otherwise, please refer to the brief guide on Blackboard.
- You **must** cite any sources used, from web pages to academic papers and textbooks.
- Please ensure your code has no errors, and that the output is shown in your submitted version.
- We have added some general notebooks on Blackboard to cover the basics of plotting in Python, Jupyter notebooks, and anaconda.
- Some questions require a mixture of code and text to answer the question. Marks are awarded based on the output of your code (i.e. graphs) and the explanation provided, not on the code itself.

Q1: Pre-processing & Feature Importance (9 marks)

This question will use the `"genes-leukemia.csv"` dataset available on Blackboard. For some background information about this dataset, see https://www.kdnuggets.com/data_mining_course/data/genes-leukemia-description.txt. The sub-questions will involve inspecting and pre-processing the data in order to use a decision tree. We will then look at which features are deemed important for prediction, and how removing important features affects tree structure.

It is expected that you will use `pandas` for this question, though this is not a requirement (but it may be more difficult if you do not).

Q1.1 (1 mark)

Count the number of records/examples where the "Treatment_Response" feature is non-missing. Describe these examples in terms of the other features (Year from XXXX to YYYY, Gender = X etc.)

Hint:

- You need to ensure that you are looking at all of the data. By default, some of the columns may be truncated, in which case you should adjust this (through e.g. `pd.set_option("display.max_columns", 100)`)

Q1.1 Answer

In [ ]:

Q1.2 (1 mark)

Explain why it is not correct to build predictive models for "Treatment_Response" using records where it is missing?

Q1.2 Answer

From Q1.3-Q1.6 (inclusive), use only the subset of data where "Treatment_Response" is non-missing.

Q1.3 (1 mark)

Remove the features that are either all the same or have all missing values. Which sample fields should you keep?

Hints:

- For simplicity in the following questions, also remove "FAB_if_AML".
- "SNUM" should be the index.

Q1.3 Answer

In [ ]:

Q1.4 (1 mark)

Fit a decision tree (`DecisionTreeClassifier`) using default settings to the data, now that it has been pre-processed.

As we have a small amount of data, if we want to more meaningfully assess the performance, we should use leave-one-out cross-validation. Report the accuracy across each fold, and the overall mean accuracy obtained.

Important: Please use `random_state=42` where necessary to ensure reproducible results.

Q1.4 Answer

In [ ]:

Q1.5 (3 marks)

Split the data into a training and test set (using a 75:25 ratio). Once again, fit a decision tree to this data, and report the accuracy. Visualize the tree (using `tree.plot_tree`), and state which feature/predictor is the most important. Then, removing this top predictor, fit the tree again with this feature removed. Again, report the accuracy and visualize the tree.

Compare the accuracy between the two trees. Explain why the tree is different with this feature removed.

Important: Please use `random_state=3` where necessary to ensure reproducible results.

Hint:

- You need to ensure that the *original* feature names are visible in the tree.

Q1.5 Answer

In [ ]:

Q1.6 (2 marks)

Which tree do you think is more generalizable? You may want to more thoroughly compare the trees (readability, sensitivity/specificity, structure simplicity, etc.).

Q1.6 Answer

Q2: Decision Boundaries (4 marks)

In this question, we will visualize the decision boundaries formed by three simple classifiers on an example dataset.

Q2.1 (4 marks)

We have provided code below to produce the data and to create the decision boundary. You will need to run this code using the following models:

1. "ZeroR" classifier - `sklearn.dummy.DummyClassifier` using the `"most_frequent"` strategy.
2. KNN classifier - `sklearn.neighbors.KNeighborsClassifier`
3. Decision tree classifier - `sklearn.tree.DecisionTreeClassifier`

You will need to modify the code to output the accuracy for each of the models. Using both this information and the visualized decision boundaries, explain the performance of these algorithms. A brief explanation of the classifiers will be required for this.

Hints:

- Although not necessary, the use of further visualizations, performance measures, or even datasets may help to support your discussion
- Use the decision boundaries as a reference point to explain the **differences** between the classifiers.

In [ ]:

```
'''
The code below provides you with the functions to get the data,
and plot the decision boundary.

The resulting graphs have not been properly formatted, however,
so you will need to add that. You will also need to modify the
code to output the accuracy.
'''

import numpy as np
from sklearn.datasets import make_classification

def get_data():
    # Create data
    data, labels = make_classification(
        n_features=2, n_redundant=0, n_informative=2,
        random_state=1, n_clusters_per_class=1
    )
    # Set the RNG
    rng = np.random.RandomState(42)
    # Add some noise
    data += 2 * rng.uniform(size=data.shape)
    return data, labels

def plot_boundary(X, ax, clf):
    # Plotting decision regions
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(
        np.arange(x_min, x_max, 0.1),
        np.arange(y_min, y_max, 0.1)
    )
    Z = clf.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
    ax.contourf(xx, yy, Z, alpha=0.4)
    return ax

def boundary_full(data, labels, model, name, **kwargs):
    # Create estimator/model/classifier
    clf = model(**kwargs)
    # Fit the classifier
    clf.fit(data, labels)

    # Create axis
    fig, ax = plt.subplots()
    # Call the provided function
    ax = plot_boundary(data, ax, clf)
    # Now add the data (using scatter)
    # Ensure to colour the points according to the prediction
    ax.scatter(data[:,0], data[:,1], c=labels, s=20, edgecolor="k")
    # Format the graph...
```

Q2.1 Answer

In [ ]:

```

```

Question 3: Training Time Comparison (4 marks)

Q3.1 (2 marks)

Plot the training time for both `DecisionTreeClassifier` and `GaussianNB` against the data size. A function to generate the data is provided to you, which takes the size as its only argument.

Explain what you observe and your understanding in terms of training time and data size (include a graph). Consider algorithm implementation and potential stochasticity in running times.

In [ ]:

```
# Use this function to measure the time
from time import time

# Use this function to generate the data
def create_data(size):
    # Create data
    data, labels = make_classification(
        n_samples=size,
        n_features=2, n_redundant=0, n_informative=2,
        random_state=4, n_clusters_per_class=1
    )
    return data, labels
```

Q3.1 Answer

In [ ]:

```

```

Q3.2 (2 marks)

What do you think would happen if we continue increasing the number of instances? Which of the algorithms would be more suitable for a very large number of instances and why? Consider the algorithms' complexity and how they scale.

Q3.2 Answer

Question 4: Memory Usage Comparison (3 marks)

Q4.1 (3 marks)

Plot the memory usage of the `DecisionTree` model against the data size. Explain the memory usage of the model (including a graph in your answer).

You should use the same `create_data()` function provided for Q3, and ensure that you have downloaded `memory.py` from Blackboard in order to load the `measure_memory()` function.

In [ ]:

```
from memory import measure_memory
```

Q4.1 Answer

In [ ]: