HW7 (70 points)

Recall that you are not allowed to use AI-generated code on these problems, though you can ask an AI any questions you have about the material in general.

This problem set needs the spotify.csv file, unless you are doing a custom miniproject.

Problem 1: Nearest neighbors (24 points [1,2,3,4,3,6,5])

In this problem, we'll train on the wines dataset (https://archive.ics.uci.edu/dataset/109/wine).

```
from sklearn.datasets import load_wine

# Load the dataset
data = load_wine(as_frame=True)
X = data.data
y = data.target

X.head()
```

a, 1 pts) Print the column (feature) names of the dataframe X.

```
#TODO
```

b, 2 pts) Using the method `value_counts()` on the target variable `y` determine the number of targets within each distinct target class.

```
#TODO
```

c, 3 pts) Now use scikit-learn's train_test_split() function to separate X and y into training and test sets. Pass in the argument random_state=110 and use a test size of 0.2 so that your split is consistent with other students' and ours.

```
#TODO
```

d, 4 pts) Train a k-nearest neighbors classifier on the training data, where k is 3.

```
#TODO
```

e, 3 pts) Use the score() method of the nearest neighbors classifier on the test data, to determine how well this classifier does on the withheld data.

```
#TODO
```

f, 6 pts) This accuracy seems low. Look at your dataset and implement a solution which improves the accuracy. Do not adjust the number of neighbors k, the seed or the test_size in the train_test_split method. Retrain this new model and report the accuracy. (You may need to review the lecture notes on k-Nearest Neighbors to recall some conditions where they don't work as well.)

```
#TODO
```

g, 5 pts) Does using the score() method with your k=3 model on the training data give 100% accuracy? For which value of k would you expect a KNN model to give 100% accuracy on the training data and why?

```
#TODO (score() call)
```

TODO

Problem 2: Decision trees (21 points [8,5,4,4])

a, 8 pts) Reload the original wine data set and create the same train/test split as the previous problem, train a scikit-learn entropy-based decision tree classifier on the unscaled data, and use score() to evaluate the classifier. When creating the DecisionTreeClassifier (anytime throughout problem 2), pass in the argument random_state=110 so its behavior matches our solution.

```
#TODO
```

b, 5 pts) What is overfitting? (If you use language from another source to answer this, cite the source.) Call score() on the training data. Is this performance consistent with the idea that the model is overfitting?

```
#TODO
```

TODO

c, 4 points) Experiment with setting the following parameters of the DecisionTreeClassifier constructor: max_depth=2, min_samples_leaf=4, random_state=110. Does the evidence from train and test performance suggest the model is overfitting now?

```
#TODO
```

TODO

d, 4 points) Using tree.plot_tree(), draw the tree you made in part (c). You don't need to name the types or features.

```
#TODO
```

Problem 3: Miniproject (25 points + up to 48 points extra credit)

This problem has an open-ended component. You can get full credit by doing the problem set with the *suggested* data set (spotify.csv), but you can get **extra credit** by doing it with a different dataset that you have found on the Internet (for example, on Kaggle.com) and/or doing more analysis than required. You also have the opportunity to get **additional extra credit applied to your final exam grade** if you are selected to give a lightning talk on **4/27** about what you found with this project. The lightning talk is only possible if you chose your own dataset.

Note that for this problem, we are relaxing our requirements for generative AI and code outside the scope of the course - you may use either on this problem. However, such miniprojects may not be selected for lightning talks - we will show a preference for projects that use tools we have covered, well.

The default dataset, which is not eligible for extra credit, is the file spotify.csv, which originally came from https://www.kaggle.com/datasets/maharshipandya/-spotify-tracks-dataset but which you can now find on Blackboard where you found this assignment. (For a description of what its features mean, see https://developer.spotify.com/documentation/web-api/reference/get-audio-features.) We'll assume in the instructions that if you're using that dataset, you're trying to predict genre from the other numerical features.

If you choose your own dataset, pick one where you think it would be interesting but feasible to predict some variable in the dataset from the others. If it needs a lot of "cleaning" to be usable, you will get extra credit, but you could also consider looking at a few datasets and picking one that seems somewhat close to directly usable. (You do *not* need high accuracy in your classifier to get full or extra credit; problems like predicting stock prices from their histories are inherently harder than classifying country from latitude and longitude, for example.)

Some datasets are ineligible for extra credit because we already extensively analyze them in this class. These are the Titanic dataset, the Starbucks dataset, the Wisconsin Breast Cancer dataset, and the Palmer Penguins dataset.

a 5 points + 16 points EC) Load the dataset as a DataFrame and prepare it for machine learning. In the spotify.csv case, we suggest using a sklearn.preprocessing.LabelEncoder to turn the target column into numerical classes; see examples in the documentation (https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html). You should also fix any columns that look basically numerical, but for some reason were processed as strings instead; but the suggested dataset doesn't have any of these.

If you are cleaning a novel dataset, you may get extra credit here if it takes more work than the spotify dataset to clean.

Suggested dataset EC: For the suggested dataset, you will get better results if you limit the target genres to four easily distinguished categories, such as 'acoustic', 'dance', 'grunge', and 'show-tunes'. You can use .unique() to see all the categories available in this column. Choosing your own 4 categories from unique() is worth 4 points of extra credit. Limiting the classes like this is otherwise optional.

```
# TODO (may want to break this into several code boxes)
```

b, 10 points) Try predicting your target variable using a RandomForestClassifier from scikit-learn, with all the other numerical features in the dataset as your features. You can create a dataframe that includes just your numeric features with df.select_dtypes(include='number'), and drop your target (to-be-predicted) column from your features if you need to with df = df.drop(columns=['target']). (The suggested dataset should also drop the first 'Unnamed' column - that row number predicts the genre number pretty well in that dataset!) Use a train-test split with 10% of the data in the test set, and evaluate the accuracy on the test set.

```
# TODO
```

c, 6 points) Choose one argument to RandomForestClassifier besides n_estimators that you vary to try to improve your classifier's accuracy. (See documentation here: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html) Train three different classifiers with different values for this parameter, counting the one you already trained.

```
# TODO
```

```
# TODO
```

d, 2 points) Use the *feature_importances_* attribute of the RandomForestClassifier to find the relative importances of all your features in your best model.

```
# TODO
```

e, up to 32 points EC) In this step, perform some additional analysis of your choice on your dataset, such as looking at correlations, performing statistical tests, or training a different machine learning classifier or regression. You could also plot data for credit, using scatter plots, bar charts, or other visualizations. Choose your methods with an eye toward being interesting for step (f). This step is extra credit.

```
# Anything here:  correlations, statistical tests, other machine learning...
# TODO
```

f, 2 points plus shot at lightning talk) Look over your findings from parts (a-e) and summarize anything interesting you learned about the data from doing this study. The students with the best answers to this question (who also chose to analyze novel datasets) may be selected to give lightning talks for additional extra credit. (Interesting insights, making good use of tools we covered in the course, and extensiveness of analysis all play a role in the selection process.)

TODO

When you are done, submit both your .ipynb (File->Download->Download .ipynb in Google Colab) and a PDF (File->Print->Save to PDF) to Blackboard where you found this assignment. (The

backup PDF helps us give you points if there's a problem with your .ipynb.)