Lab 11: Machine Learning

Part 0: Submission/Reference Materials

Remember to read (and write!) good documentation and use the internet to find code examples. Finding and using appropriate/accurate reference materials is hard to teach directly, but tends to be what separates bad scientific programmers from good ones!

If you get stuck:

- Review the lecture material
- Check the docstrings of functions you are trying to use (hint: use ?function in jupyter)
- The pandas official documentation includes all of the commands you need to complete this week's practical https://pandas.pydata.org/docs/user_guide/index.html#user-guide
- Use online resources like stackoverflow, w3schools, realpython etc. (but make sure you don't blindly copy code without working out HOW it works).
- Looking up materials is totally fine but remember if you copy code (or autocomplete it) directly from any source, you MUST cite where you got it from in a comment next to the code.
- Ask TAs! (send an email or book an online/in-person appointment)

Submit this assignment as a formatted notebook - include an explanation of your answers and make sure every function has a clear docstring that explains what it does, its arguments, and what it returns. This will be part of the grading of each of your answers.

You will need to import several libraries/modules from scikit-learn, pandas, numpy, and matplotlib for this assignment. These packages should all be installed with anaconda but double check they work but running this at the top of your notebook.

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.manifold import TSNE
from sklearn.cluster import KMeans
from sklearn.linear_model import LogisticRegressionCV
```

Part 1: Preparing the dataset

Today we are going to use scikit-learn to explore a dataset about Portuguese "Vinho Verde" red wine[1]. This dataset includes data about 11 different physicochemical properties of 1,599 red wine bottles along with a quality rating from 0 (very bad) to 10 (excellent). These quality ratings are based on the median score of 3 or more independent evaluations by a group of wine experts[2].

The dataset columns include the following measurements:

1 - fixed acidity

2 - volatile acidity

3 - citric acid

4 - residual sugar

5 - chlorides

6 - free sulfur dioxide

7 - total sulfur dioxide

8 - density

9 - pH

10 - sulphates

11 - alcohol

12 - quality (score between 0 and 10)

Our goal in today's lab will be to explore this data with some unsupervised machine learning algorithms. We will then train and compare a couple of supervised learning models to classify whether a bottle of wine is predicted to taste good or bad based on its physicochemical properties.

Q1 [6 points]: Complete the following step to prepare your dataset for exploration and ML analyses:

- Download the dataset in CSV format from the course website:

  https://maguire-lab.github.io/scientific_computing/static_files/practicals/winequality-red.csv

- Read the dataset into a pandas DataFrame called `wine` (*hint: look at the downloaded file to work out what character to use as the* `sep=` *argument*)
- Create a new column in `wine` with the name 'good' which contains 1 if the `wine['quality']` is >5 and 0 if `wine['quality']` is <=5. There are many different ways you can do this (e.g., for loops, indexing with boolean masks, applying a function to column).

---

[1] Cortez, Paulo, et al. "Modeling wine preferences by data mining from physicochemical properties." *Decision support systems* 47.4 (2009): 547-553.

[2] Although see this infamous paper for why we may have some doubts about the robustness of wine tasting: https://web.stanford.edu/class/linguist62n/morrot01colorofodors.pdf

- We then want to split our dataset into the data (physicochemical properties) and labels (the `wine['good']` column). Create the following pandas series by running `wine_good=wine['good']` and then drop the `quality` and `good` columns from `wine`. If we don't drop `quality` and `good` columns the classifier will end up just using them to predict the label instead of the physicochemical properties!
- Finally, we want to split our dataset into 80% training and 20% test set using `train_test_split` sklearn's `model_selection` module. *(hint: `test_size=0.2`)*

Part 2: Dataset Exploration

Now we are going to explore the dataset to see if there are any obvious problems and so we have an idea of how well our classifier is likely to work. As the training data has 11 dimensions to visually inspect it using a plot we will need to perform dimensionality reduction down to 2 dimensions.

Q2 [6 points]: Create a function called `apply_tsne` that fits and applies a t-SNE dimensionality reduction[3] model to the training data. The function should:

- Accept the parameters:
    - Pandas or numpy matrix of training data
    - Number of t-SNE components (i.e., how many dimensions should the transformed data have) (default=2)
    - t-SNE perplexity (default=30)[4]
- Fit and transform the data using t-SNE with the above parameters.
- Return the transformed data

Then apply this function to your wine training data with `n_components=2`

Now we are going to see how whether naive unsupervised clustering of the dataset captures the good and bad wine groups (i.e., `wine_good == 1` or `wine_good == 0`)

Q3 [6 points]: Create a function called `apply_kmeans` that performs K-means clustering on the training data. The function should:

- Take the following arguments:
    - pandas dataframe or numpy matrix of the training data
    - number of clusters (default should be set to 2)
- Fit the K-means model to the data using the number of cluster specified in the parameters (i.e., not hardcoded)
- Return the fitted model and the cluster assignments for each data point

Then apply this function to the wine training data using 2 centroids.

[3] See this article for why t-SNEs can be misleading: https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1011288
[4] You can play around on this website to see how perplexity impacts the way t-SNEs work https://distill.pub/2016/misread-tsne/

Q4 [8 points] Using matplotlib create 2 scatter plots of the same t-SNE results with the following color schemes (these should have the same scattering of points just coloured differently):

1. t-SNE 2-D transformed data points coloured by the `wine_good` labels
2. t-SNE 2-D transformed data points coloured by the K-means cluster assignments

Make sure these plots are labelled appropriately (xlabel, ylabel, title, and legend)

Part 3: Classification

In this section, we'll train and evaluate supervised logistic regression models to predict whether a wine is good (quality > 5) or not.

Q5 [8 points]: Create a function called `train_logistic_regression` that trains and compares a Logistic Regression model with L1 regularisation and a Logistic Regression model with L2 regularisation using cross-validation. The function should:

- Accept arguments:
  - Training data in pandas or numpy format
  - Training labels in pandas or numpy format
  - Number of cross-validation folds to use (default=5)
- Train a LogisticRegressionCV model with an L1 penalty and the specified number of CV folds
- Train a LogisticRegressionCV model with an L2 penalty and the specified number of CV folds
- Print the accuracy of each model on the training data.
- Return model with the highest average accuracy

Q6 [3 points]: Evaluate the test performance accuracy of the single best-scoring logistic regression model using the held-out test data from **Q1**.