CCE3503—Practical Machine Learning

Assignment 1: Feature analysis

Trevor Spiteri
trevor.spiteri@um.edu.mt

Last updated: 2024-10-30

Aim

The aim of this assignment is to prepare and clean a dataset, to analyse its features, and to evaluate the feature selection by training and testing neural networks.

Dataset

The dataset to use is the Communities and Crime Unnormalized dataset publicly available at the UCI repository at archive.ics.uci.edu/ml/datasets/Communities+and+Crime+Unnormalized.

The dataset is also available for download on VLE as a csv file.

The dataset contains 147 features:

- 125 predictive: can be used to predict target features
- 4 non-predictive: cannot be used to predict target features
- 18 potential goal: can be used as target features

The column titles and the information about what kind of feature is in each column can be found under the *Additional Variable Information* section in the dataset page.

Missing values are marked as "?".

The target feature we are using in this assignment is the **number of assaults per 100K population**, with column label **assaultPerPop**.

Requirements

- The assignment involves some Python 3 coding and visualization using Jupyter notebooks.
- You can use the pandas toolkit for data analysis.
- You can use Matplotlib and/or Seaborn for visualization.
- You can also use NumPy and SciPy in your numeric and data manipulation.
- You can use scikit-learn for machine learning tools.
- Use markdown cells to separate task sections and to include text required by the tasks below.

Tasks

1 Data cleaning, missing data and normalization (20 marks)

- Remove the non-predictive features and the potential target features from the dataset.
- Use appropriate techniques to handle features with missing data. Comment on why you selected the techniques used for missing data, and any adverse effect they may have on the data.
- Normalize the data using an appropriate mechanism. Comment on the mechanism used and why you selected it.

2 Filter methods (20 marks)

- Obtain a colour coded correlation matrix for the remaining features (for example using the pandas `corr` function and the seaborn `heatmap` function).
- Select an appropriate threshold to apply to the correlation coefficient, and hence determine which features to keep. Comment on how you selected the threshold.
- Randomly split the dataset into a train subset and test subset using an appropriate ratio of train subset samples to test subset samples. Comment on the choice of ratio.

- Train a neural network on the selected features to predict the number of assaults per 100K population, and then use the trained network on the test subset to compute the mean squared error (MSE). (For the neural network, you can use for example MLPClassifier from scikit-learn.)

3 Wrapper methods (20 marks)

- Use Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS) to obtain a maximum of 40 features in each case.
- Discuss the strengths and limitations of SFS and of SBS in this context.
- For each of the SFS and the SBS feature selections, train a neural network using the same dataset split ratio as Task 2 to predict the number of assaults per 100K population, and compute the MSE on the test subset.

4 Feature projection (20 marks)

- Use Principal Component Analysis (PCA) to reproject the features to principal components, and produce a plot to show the variance ratios of the principal components.
- Select an appropriate cut-off point and remove features beyond this point.
- Comment on how you picked the cut-off point.
- Discuss the strengths and limitations of PCA in this context.
- Train a neural network using the same dataset split ratio as Task 2 to predict the number of assaults per 100K population, and compute the MSE on the test subset.

5 Comparison (10 marks)

- Compare the MSE for the four methods used (filter, SFS, SBS, PCA).
- Discuss which methods produce the best results and why.

6 Code quality (10 marks)

This is not an extra task: these marks are allocated to the quality of the code for the tasks above.

- Clarity of code: code should be clear and easy to understand.
- Efficiency: for example proper use of vectorization should be made where necessary, and unnecessary redundancy should be avoided.
- Documentation: any code that is doing something complex or which is not easy to interpret should be explained in comments; and markdown cells can be used to document the workflow, required steps, insights, and so on.

Plagiarism

All work must be your own individual work, except where acknowledged and referenced.

While you can use online tools and resources, all code, analysis and documentation must be your own work. Copying code or explanations directly from online sources, including tools like ChatGPT, is considered plagiarism except where acknowledged and referenced.

Submission

- The submission deadline is **2024-11-29 16:00**.
- A report must be submitted as both
    1. an executable Jupyter notebook and
    2. a pdf that should be generated from the Jupyter notebook.

  Make sure that the Jupyter notebook executes cleanly. A good idea is to restart the Python kernel, run all the cells in order immediately after restarting the kernel, and then ensure that no errors have occurred.
- A filled in plagiarism declaration form must be submitted as well.