Applied Machine Learning – Project 1 Assignment

Introduction

The dataset that we will use for this assignment is the Enron email dataset. You can find the full dataset on the web here: http://www.aueb.gr/users/ion/data/enron-spam/. The dataset is a collection of public domain emails from the Enron corporation. The emails have been manually classified as spam and non-spam. The primary goal of the assignment is to create a supervised classification pipeline to classify emails as spam or non-spam from the training data. You are free to use either the preprocessed emails or the raw emails for your analysis.

Preprocessing

The first stage in the pipeline is to preprocess and clean the dataset.

Training and test splits

The very first thing that you will need to do is split the data into training and test sets. Write a Python script to perform the split: 70% of the data for training and the remainder for test. Take appropriate measures to ensure that the test set is not biased in any way. There may be duplicate and empty emails that need to be handled appropriately. Store the resulting training and test sets in files using any convenient data format that you like. Collect and record statistics on the resulting training and test sets including total numbers of spam and non-spam emails in each set.

Feature extraction

The second part of preprocessing will be to extract the features you will need for the remainder of the analysis. You may revisit this stage many times as you become more familiar with the dataset and the kinds of features that may be useful for the classification task. You may want to start with using a bag-of-words model here to transform the documents into a fixed length representation suitable for classification. The sklearn.feature_extraction.text package may be useful here.

The features you choose will affect the performance of the final classifier, and there are many possibilities (e.g. stop word removal, TF-IDF encoding, infrequent word removal, etc.). Choose something you think is reasonable to start with and later you can experiment with alternatives on the validation set.

Exploratory data analysis

Use the training section of the dataset to perform some exploratory data analysis. The goal at this stage is to become accustomed with the data and gain insights into the kinds of features may be useful for classification.

Find the top-20 most frequently used words in spam and non-spam emails and use a bar plot to show their relative frequencies. Compare the distribution of email lengths in spam and non-spam emails using appropriate plots (e.g. a boxplot). Are spam emails usually shorter or longer? Look at the corresponding data points for the longest emails. Explain why these are so much longer than most others. Document all your findings and any other interesting observations that you can find.

Supervised classification

Train a supervised classification model on your features and calculate validation accuracy either on a hold out validation set or using cross validation. Record the final accuracy of the classifier. How many of the emails are correctly classified by the model? How many are misclassified? Use the sklearn.metrics package to investigate the kinds of errors that are being made. Document all findings. Use the Python pickle module to save the model to the disk.

Model selection

Select several candidate models that you want to compare. This could include different classifiers (e.g. naive Bayes MultinomialNB), different hyperparameters, or different sets of features (remember, hyperparameter selection is part of model selection!). Use a validation set or cross-validation to compare the accuracy of different models. Create plots to compare a subset of the models that you investigated during model selection. Retain the most effective model for evaluation.

Model evaluation

Estimate the out-of-sample error for the model that you found to be most accurate during model selection by evaluating it on the held-out test set. Use the sklearn.metrics package to benchmark the model in several ways. Create an graph plot for the model. Compute the model's accuracy on the test set. Comment on the implications of the resulting for a real production classifier.

Submission

You are required to submit:

- An electronic copy of the assignment report in **PDF format**;
- An electronic copy of all source code developed (.tar.gz or .zip archive).

Both files can be submitted with Canvas before **Sunday, March 10th**. Please ensure to include **your name and student number** on the front page of the **report** and the python **code**.

Report

The final submission should be a report documenting all assumptions, design decisions, and findings. Include visualizations, plots, and tables. You should strive to make your work completely reproducible using only the report document: include details on everything you tested and all results. When reporting metrics, remember to always say which subset of the data they are calculated on (train/validation/test). Include an appendix explaining how to run your code to reproduce the experiments.

The page limit for the report is 20 pages. Please do not include full texts of very long emails from the dataset as examples.

Code

I strongly recommend using Python for this project. You are free to use any external libraries you like (e.g. scikit-learn, pandas, seaborn, etc.), as long as you document which ones you used in the report. You can use both Python scripts and IPython notebooks for implementation as you see fit.

Late submissions

If you don't get the assignments done to your satisfaction and don't meet the minimum requirements by the deadline, you have the option (as with any assignment at CIT) of submitting up to 1 week late for a penalty of 10%.

This penalty is subtractive. Work that would have earned 55% if on time, would get 45% (not 49.5%) if late.

The penalty is applied weekly. So 1 day late costs the same 6. If you want to take that option please let me know. Otherwise, I will just correct whatever I have.

If you have a specific reason for submitting a late assignment (sickness, etc) please contact me directly or submit a medical certificate in the department secretary.

Plagiarism

Please read and strictly adhere to the CIT Honesty, Plagiarism and Infringements Policy Related to Examinations and Assessments. Note that reports are **checked** against each other and against external web sources for plagiarism. Any suspected plagiarism will be treated seriously and may result in penalties and a zero grade.

Grading

The assignment is worth 40% of the overall mark for the module. Marks will be awarded based on the quality of the resulting report. In particular, I will be checking to see if you are handling data correctly, carrying out exploratory analysis to gain insights, correctly performing model selection, and critically, documenting everything in a clear and concise way. The submitted code will also be checked to ensure that the work is your own.