EE353 / EE769 Introduction to Machine Learning (Jul 2024 edition)

Electrical Engineering, Indian Institute of Technology Bombay

Programming Assignment – 3 : Classification and Feature Engineering

Instructions:

a) Name files A3_<RollNo>.<extension>. Submit three files (or links to these): an .ipynb, a .py, and a video demo (approx 10 minutes)

b) Use good coding practices such as avoiding hard-coding, using self-explanatory variable names, using

functions (if applicable). This will also be graded.

c) You may use libraries such as scikit-learn, and need not code anything from scratch.

d) Cite your sources if you use code from the internet (line-by-line or block-by-block). Also clarify what you have modified.

Objective 1: Practice various steps and due diligence needed to train successful classification models.

Background: Banks and other businesses run various marketing campaigns to nudge existing or potential customers to take particular actions. If they take that desired action, the marketing campaign was successful. In order to optimize the marketing budget across various campaigns, it will be great to be able to predict the customers for which a particular marketing campaign will be successful. This prediction can be done based on the past data of success or failure of similar marketing campaigns. You are tasked with building and testing such a model based on a dataset available on Kaggle at: https://www.kaggle.com/datasets/janiobachmann/bank-marketing-dataset/data

1. Perform exploratory data analysis to find out: [1.5]
    a. Which variables are usable, and which are not? Why?
    b. Are there significant correlations or other relations among variables?
    c. Are the classes balanced? Classes are in poutcome column.
    d. Which classes will you use?
2. Select metrics that you will use, such as accuracy, F1 score, balanced accuracy, AUC etc. and state the reason for the choice. [0.5]
3. Develop a strategy to filter and code variables. [2]
    a. Should you be using continuous variables as they are, or should you normalize them, or take a transform? Why?
    b. Should you be using all values of discrete variables, or should you try to reduce them by combining some of the values?
    c. Are some variables very likely to be unreliable, noisy, or otherwise immaterial?
4. Carve out some test data. Should this be balanced in some way? [1]
5. Using five-fold cross-validation (you can use GridSearchCV from scikit-learn) to find the reasonable hyper-parameter settings for the following model types:
    a. RBF kernel SVM with kernel width and regularization as hyperparameters [1]

b. Neural network with single ReLU hidden layer and Softmax output (hyperparameters: number of neurons, weight decay) [1]

c. Random forest (max tree depth, max number of variables per node) [1]

6. Check feature importance for each model to see if the same variables are important for each model. Read up on how to find feature importance. [1]

7. See if removing some features systematically will improve your models (e.g. using recursive feature elimination https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFECV.html). [1]

8. Finally, test a few promising models on the test data. Is the model useful for the business? [1]

9. See if the model will work if you separate the training and test data in at least two pathological ways:

   a. All the training calls were in months other than June and July, while the testing was in June and July. If the test results are worse, then speculate on reasons why. [1]

   b. All the training calls were for professions other than technicians, while testing was on technicians. Is there a profession closest to technician what can be used as a substitute? [1]

Objective 2: Practice using pre-trained neural networks to extract domain-specific features for new tasks.

10. Read the pytorch tutorial to use a pre-trained "ConvNet as fixed feature extractor" from https://pytorch.org/tutorials/beginner/transfer_learning_tutorial.html and you can ignore "finetuning the ConvNet". Test this code out to see if it runs properly in your environment after eliminating code blocks that you do not need. [1]
11. Write a function that outputs ResNet18 features for a given input image. Extract features for training images (in image_datasets['train']). You should get an Nx512 dimensional array. [1]
12. Compare L2 regularized logistic regression and and random forest (do grid search on max depth and number of trees). Test the final model on test data and show the results -- accuracy and F1 score. [2]
13. Summarize your findings and write your references. [1]