SIT384 Cyber security analytics

Distinction Task 10.1D: Model evaluation metrics

Task description:

The model evaluation and selection techniques are the most important tools in a data scientist's toolbox. So far, we have introduced many model evaluation methods/metrics, such as GridSearchCV, cross_val_score, confusion matrix, precision, recall and f-score, etc. In reality, classification problems rarely have balanced classes, and often false positives and false negatives have very different consequences. We need to understand what these consequences are, and pick an evaluation metric accordingly, therefore select a right model for the given dataset.

In this task, you are given a dataset "creditcard.csv" used in practical10. Based on the code example provided in practical10, try to find the "best" classification model by comparing the evaluation metrics, especially the recall rates produced by knn and random forest models.

You are given:

- Dataset: creditcard.csv
- thresholds = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9]
- Parameter grid (param_grid):
  For knn, n_neighbors = [1, 2, 3, 4, 5]
  For random forest, n_estimators = [5, 10, 20, 50]
- GridSearchCV (model_classifier(random_state=0), {param: param_grid}, cv=5, scoring='recall')
- Other parameters of your setting

You are asked to:

- use the train and test sets split in practical10 (X_train, X_test, y_train, y_test, and X_train_undersample, X_test_undersample, y_train_undersample, y_test_undersample).
- use Grid search with cross-validation to fit the undersample data with model knn and random forest. Set cv=5.
- find and print the best parameter for each model (knn, random forest) on X_train_undersample dataset.
- for each model, build classifier using the found best parameter, predict using test sets (X_test_undersample and X_test), and plot the confusion matrix for the two predictions.
- for each model, plot recall matrices for different threshold for the undersample dataset.
- for each model, plot precision-recall curve for the undersample dataset.
- maximize code reuse by defining functions for
  - searching best parameters, e.g., print_gridsearch_scores (clf, param, x_train, y_train).
  - plotting confusion matrix (pls refer to the plot_confusion_matrix function provided in practical10).
  - building the final model with the best parameters, predicting and plotting on the test dataset, e.g., predict_plot_test (clf, x_train, y_train, x_test, y_test).

- plotting recall matrics for different threshold values, e.g., plot_recall_for_threshold (clf, x_train, y_train, x_test, y_test).
- plotting precision-recall curve on a dataset, e.g., plot_precison_recall (clf, x_train, y_train, x_test, y_test).

The above function names and arguments are just for demonstration purposes. Yours might be different, but the purpose is the same, i.e., **maximizing code reuse**. It is expected that several lines of function calls will do most of the tasks for each model.

- write a brief report, analyse & compare the results, and present your findings. (approx. 400 words.)

Note: It is very likely you will find the best parameters found for undersample dataset do not work well for the whole skewed dataset, which is normal. The ideal solution is to use GridSearchCV to find the best parameters for the whole skewed dataset, then use the found best parameters to build a new classifier for the whole skewed dataset, however it may take quite a LONG time on an office/home laptop/computer due to the size of the whole skewed dataset and amount of resources required. If conditions allow, you are recommended to have a try. In this task, we will mainly play with the undersample dataset.

Submission:

Submit the following files to OnTrack:

1. Your program source code (e.g. task10_1.py)
2. A pdf file including all the screen shot of your program running and the report
   Or
1. Your jupyter notebook. You can edit your report in a text cell, or submit two separate files (notebook and report).

Check the following things before submitting:

1. Add proper comments to your code