CPSC 483 - Introduction to Machine Learning

Project 6, Fall 2020

due December 7 (Section 02) / December 10 (Section 01)

Last updated Friday November 20, 12:50 am PST

In this project we will run into a new set of challenges when working with a “real-world” dataset, and see how an imbalanced dataset can influence classifier performance.

The project may be completed individually, or in a group of no more than three (3) people. All students on the team must be enrolled in the same section of the course.

Platforms

The platform requirements for this project are the same as for previous projects.

Libraries

You will need pandas to load the datasets and encode features, scikit-learn to run the various classifiers, and imbalanced-learn to do random oversampling.

You may reuse code from the Jupyter notebooks accompanying the textbook and from the documentation for the libraries. All other code and the results of experiments should be your own.

Datasets

UC Irvine maintains a repository of datasets for use in machine learning experiments. In this project we will use the Bank Marketing Data Set, training a series of classifiers to predict whether clients will respond to a direct marketing campaign.

Experiments

Run the following experiments in a Jupyter notebook, performing each action in a code cell and answering each question in a Markdown cell.

1. Download `bank-additional.zip` and extract its contents. Since the dataset is large and some of the algorithms we will use can be time-consuming, we will train with `bank-additional.csv`, which is a subset of the original dataset.

   Once our models are trained, we will test against the full dataset, which is in `bank-additional-full.csv`.

   The archive also contains a text file, `bank-additional-names.txt`, which describes the dataset and what each column represents.

2. Use `read_csv()` to load and examine the training and test sets. Unlike most CSV files, the separator is actually `';'` rather than `','`.

3. The training and test DataFrames will need some significant preprocessing before they can be used:

   a. Several of the features are categorical variables and will need to be turned into numbers before they can be used by ML algorithms. The simplest way to accomplish this is to use dummy coding using `get_dummies()`.

      Some algorithms (e.g. logistic regression) have problems with collinear features. If you use one-hot encoding, one dummy variable will be a linear combination of the other dummy variables, so be sure to pass `drop_first=True`.

   b. Per `bank-additional-names.txt`, the feature duration “should be discarded if the intention is to have a realistic predictive model,” so removed.

   c. The feature y (or y_yes after dummy coding) is the target, so should be removed.

   d. Some algorithms (e.g. KNN and SVM) require non-categorical features to be standardized.

4. Fit Naive Bayes, KNN, and SVM classifiers to the training set, then score each classifier on the test set. Which classifier has the highest accuracy?

5. These numbers look pretty good, but let’s take another look at the data. How many values in the training set have y_yes = 0, and how many have y_yes = 1? What would be the accuracy if we simply assumed that no customer ever subscribed to the product?

6. Use `np.zeros_like()` to create a target vector representing the output of the “dumb” classifier of experiment *(5)*, then create a confusion matrix and find its AUC.

7. Create a confusion matrix and find the AUC for each of the classifiers of experiment *(4)*. Is the best classifier the one with the highest accuracy?

8. One of the easiest ways to deal with an unbalanced dataset is random oversampling. This can be done with an `imblearn.over_sampling.RandomOverSampler` object. Use `fit_resample()` to generate a balanced training set.

9. Repeat experiments *(4)* and *(7)* on the balanced training set of experiment *(8)*. Which classifier performs the best, and how much better is its performance?

Submission

Submit your Jupyter `.ipynb` notebook file through Canvas before class on the due date. Your notebook should include the usual identifying information found in a `README.TXT` file.

If the assignment is completed by a team, only one submission is required. Be certain to identify the names of all students on your team at the top of the notebook.