*Kenytt Avery, Computer Science Department, California State University, Fullerton*

# CPSC 483 - Introduction to Machine Learning

## Project 3, Fall 2021

## due Friday, November 19 at 9:45 pm PST

*Last updated Friday November 12, 4:25 pm PST*

In this project you will use scikit-learn, which is a higher-level machine learning library that works with NumPy data, and Pandas, a library that makes it easier to manipulate data. You will explore a variety of classification algorithms, and compare their performance on a “real-world” dataset, which will introduce its own set of challenges.

The project may be completed individually, or in a team of up to four students as long as all students are enrolled in the same section of the course.

## Platforms

The platform requirements for this project are the same as for Project 1.

## Libraries and Code

In addition to NumPy and pyplot, you will need Pandas to manipulate the dataset and scikit-learn to do machine learning. You may not use any other library except the Python Standard Library.

Code from the Python documentation, from *A Whirlwind Tour of Python*, from the Jupyter notebooks accompanying the textbook, and from the library documentation may be reused. All other code and the results of experiments must be your own original work or the original work of other members of your team.

## Dataset

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 prospective clients will respond to a direct marketing campaign.

## Experiments

Begin by adding a Markdown cell introducing your project. Be sure to include the names of the members of your team, the semester, section, and project number.

Run the following experiments in a Jupyter notebook, running Python code in code cells and describing your results in Markdown cells. Be sure to mark clearly in the notebook where each experiment begins.

1. Download `bank-additional.zip` and extract its contents. Use `pandas.read_csv()` to load and examine the dataset from `bank-additional-full.csv`, and `pandas.DataFrame.head()` to examine its contents.

   Note that unlike most CSV files, the separator is actually `';'` rather than `','`. The archive also contains a text file, `bank-additional-names.txt`, which describes the dataset and what each column represents.

2. Use `sklearn.model_selection.train_test_split()` to split the features and target values into separate training and test sets. Since the dataset is large, use 90% of the original data as a training set, and 10% for testing. To make sure that your results are reproducible, pass the keyword argument `random_state=(2021-10-25)`.

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

   a. Per the description in `bank-additional-names.txt`, the duration "should be discarded if the intention is to have a realistic predictive model."

   b. The feature y is the target response; set this aside for use in training and testing, then drop it from your features.

      Note that by default, `pandas.DataFrame.drop()` does not drop columns in-place.

4. First, let's see if we can determine whether a client will subscribe to a term deposit based on what we know about them. Let's take as features the subset of the input variables described as "bank client data" in `bank-additional-names.txt`.

   Most of these features are categorical variables that will need to be encoded before they can be treated as vectors. The simplest way to accomplish this is to use `pandas.get_dummies()`.

   Recall that some algorithms (e.g. logistic regression) will have problems with collinear features, so be sure to set the `drop_first` keyword argument.

5. Use scikit-learn to fit a Categorical Naive Bayes classifier to the features you identified in the previous experiment, then score it on both the training and test sets. How accurate is the classifier?

6. Take another look at the data you used for the previous experiment. Most of the data is categorical, but age is a quantitative predictor. Categorical Naive Bayes assumes that

each value of the age variable is a separate category. How many categories are there? Is this reasonable?

7. Try splitting ages into bins, one per decade. Verify the number of bins, then re-train your classifier the bins instead of the original age value. Does its performance change?

8. Repeat experiment *(5)* (i.e. with the original age values) with a KNN classifier. How do the results compare?

   Note that since KNN requires a pass over the data for each prediction, this classifier may take several minutes to score.

9. In the previous experiments, you should have found that your test results are suspiciously similar. Let's take a closer look.

   How many values in the test set have response 0, and how many have response 1? What would be the score if we simply assumed that no customer ever subscribed to the product?

10. Use `numpy.zeros_like()` to create a target vector representing the output of the "dumb" classifier described in the previous experiment, then create a confusion matrix and find its AUC.

11. Create confusion matrices and compute the AUC for each of the classifiers in experiments *(7)* and *(8)*. How well are these classifiers actually performing?

12. It should be clear from your results that we are dealing with imbalanced data. One of the easiest ways to deal with an unbalanced dataset is random oversampling.

    Use `pandas.DataFrame.where()` and `pandas.DataFrame.sample()` to generate balanced training sets by weighting the values with response 1 more heavily. To make sure that your results are reproducible, pass the keyword argument `random_state=(2021-10-25)`.

13. Retrain both classifiers on balanced training sets, and find the score, confusion matrix, and AUC for each. Which classifier performs better?

14. So far, we have been using data about our clients to predict subscriptions. But perhaps their decisions are influenced more by the overall health of the economy than by their individual circumstances.

    Let's try the input variables described as "social and economic context attributes" in `bank-additional-names.txt`. These features are quantitative, so you can use Gaussian Naive Bayes. Based on the score, confusion matrix, and AUC, how well does this data predict the response?

15. Do the results of the last experiment change if the training set is balanced?

## Submission

As described above, the first cell in your notebook should include the names of the members of your team, the semester, section, and project number. Only one submission is required.

Since you may be actively editing and making changes to the code cells in your notebook, be certain that each of your code cells still runs correctly before submission by selecting *Run All* from the drop-down menu bar.

Submit your Jupyter `.ipynb` notebook file through Canvas before 9:45 pm PST on the due date.

The Canvas submission deadline includes a grace period of an hour. Canvas will mark submissions after the first submission deadline as late, but your grade will not be penalized. If you miss the second deadline, you will not be able to submit and will not receive credit for the project.

**Note**: do not attempt to submit projects via email. Projects must be submitted via Canvas, and instructors cannot submit projects on students' behalf.

**Note**: In order to submit a project as a team, you must join a group for that project in Canvas.

- Teams are specific to the project, so you must join a new group even if you worked with the same team on a previous project.
- Teams must be created by the instructor; you cannot submit projects using a "Student Group."
- Several groups have been pre-created by the instructor, and your team may join one of those for your submission. The first team member to join will automatically be assigned as the group leader.
- If there are no groups available, email the instructor immediately to request a new group to be created. Do not wait until the due date.

See the following sections of the Canvas documentation for instructions on group submission:

- How do I join a group as a student?
- How do I submit an assignment on behalf of a group?

### Grading

The grade for the project will be assigned on the following five-point scale:

---

**Exemplary (5 points)**

Results are correct and clearly presented; explanatory text clearly and concisely tells the story with appropriate context and analysis; organization makes it easy to review.

**Basically Correct (4 points)**

The analysis comes to correct (or defensible) results and conclusions, but the presentation is not easy to follow and/or portions are not clear or lack context.

**Right Idea (3 points)**

The approach is appropriate, but the work has mistakes in code, analysis, or presentation that undermine the correctness of conclusions.

**Solid Start (2 points)**

The work makes a good start, but has fundamental conceptual problems in code, analysis, or presentation such that it will not produce legitimate results.

**Did Something (1 point)**

The solution began an attempt, but is either insufficient complete to assess correctness or is on entirely the wrong track.

**Did Nothing (0 points)**

Project was not submitted, submitted code belonging to someone other than the members of the team, or submission was of such low quality that there is nothing to assess.

---

Acknowledgements: this grading scale is drawn from the general rubric used by Professor Michael Ekstrand at Boise State University.