Introduction to Machine Learning

ECE 580
Spring 2022

HW #4

Submission Instructions

Submit your work to the corresponding assignment in Gradescope. Although Gradescope accepts multiple file formats, they strongly recommend submitting your assignment as a single PDF file.

You are responsible for ensuring the uploaded file is: 1) the correct file, 2) complete (includes all pages), and 3) legible.

You are responsible for tagging the pages that correspond to each question. Pages may be tagged after submission, even if the submission deadline has passed. If you are submitting close to the submission deadline, submit your assignment first then immediately return to tag the pages.

When code is requested, submit a PDF print-out of your code. Submitting a URL for a cloud-based repository is insufficient.

Scoring Information

Individual questions will be scored holistically on the 9-point scale described in the syllabus. The homework assignment score will be the weighted average of the individual scores for each question. (The weight for each question is shown in parentheses to the left of the question number.)

Late Submissions

Late submissions will be accepted up to 5 days after the submission deadline, with the following point penalty applied if its late submission is not excused: [1]

1 day (0^+ to 24 hours) late: 0.2 point deduction
2 days (24^+ to 48 hours) late: 0.5 point deduction
3 days late: 0.8 point deduction
4 days late: 1.6 point deduction
5 days late: 3.2 point deduction
6 or more days late: score = 3.0 (not accepted for credit)

The late policy is designed to be minimally punitive for submissions up to 3 days late, yet encourage staying current with the coursework for our course by not allowing one assignment's late submission to overlap with the next assignment's submission.

A homework score will not drop below 3.0 as a result of applying the late penalty point deduction.

[1] One day = one 24-hour period or fraction thereof.

Exploring K-Nearest Neighbors

It is helpful to understand not only *how* a classifier works, but when it might behave unexpectedly[2] and *why* it might behave unexpectedly. Here we are exploring how the relative proportions of the classes in the training set may influence the decision statistics produced by a KNN classifier.

1. Suppose you are using a K-Nearest Neighbors classifier with the L_2 distance metric and majority vote decision rule, and have a total of N training points for two classes H_0 and H_1. The number of H_0 training points is given by N_0 and the number of H_1 training points is given by N_1, so the total number of training points is $N = N_0 + N_1$.

(5) (a) Assume the classes in the training data are balanced, so $N_0 = N_1$. Sketch an example of 2-dimensional data for which performance is perfect for any value of $k \leq \frac{N}{2}$ when the classifier is tested on the training data (*i.e.*, no cross-validation). (It may be easier to represent the data as point clouds rather than discrete points.)

(5) (b) Suppose the training data for each class is distributed as you've drawn above, but the classes are not balanced, so $N_0 \neq N_1$ (but $N = N_0 + N_1$ is unchanged). Explain how to determine the maximum value of k for which performance will be perfect when the classifier is tested on the training data (*i.e.*, no cross-validation).

(5) (c) What are potential implications for KNN classification decisions when the classes in the training data are imbalanced, $N_0 \neq N_1$? In other words, how might KNN classification decisions be influenced by the relative proportions of H_0 and H_1 training data? It may be helpful to think about boundary conditions ($N_0 = 1, N_1 = N - 1$ and $N_0 = N - 1, N_1 = 1$) as part of developing your response to this question.

(5) (d) Suppose you believe the environment in which your classifier will be deployed has balanced classes $\left(P(H_0) = P(H_1) = \frac{1}{2}\right)$, but the classes in your training data are imbalanced. How could you modify KNN to mitigate potential issues associated with imbalanced classes in the training data?[3]

[2] Some might refer to this as the classifier failing, but often the classifier is doing exactly what it is supposed to do on data that is inconsistent with the classifier's underlying assumptions.

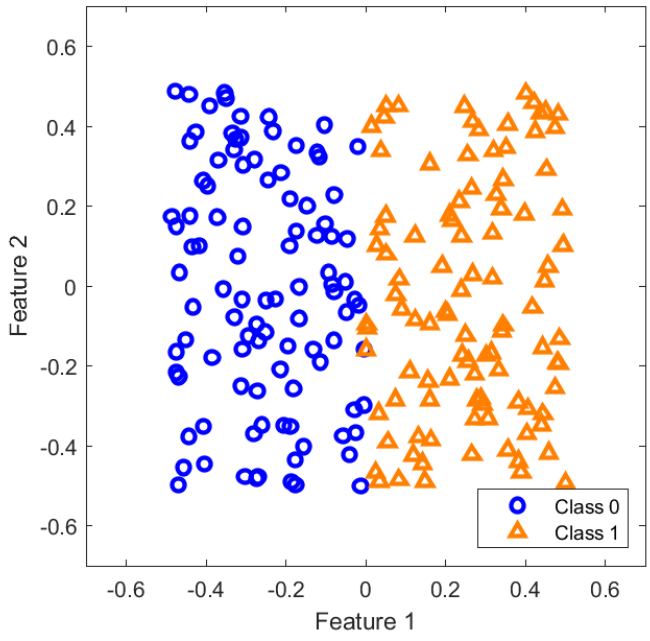
[3] This question is not asking for a formal derivation of an optimal strategy, but a qualitative description of what you would think about and how you might adapt KNN.

Exploring Cross-Validation within the Context of Classification

Make sure you are able to apply cross-validation to (more fairly) estimate classifier performance using the available training data.

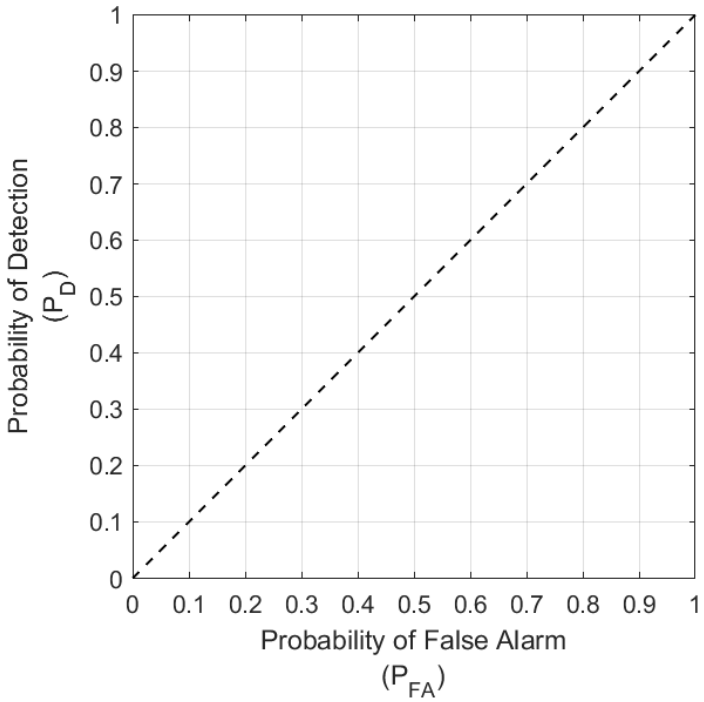
Regardless of whether you choose to write your own functions or leverage functions that may be available through Matlab or Python packages or libraries, you are responsible for understanding how the function(s) you are using work so you can effectively apply them to suit your needs and correctly interpret the results they provide.

The following questions concern a data set provided as a `csv` file that contains both data and suggested folds for cross-validation, `dataSetCrossValWithKeys.csv`. This csv file is organized such that each row contains the fold assignment (either 1 or 2), followed by the true class (either 0 or 1), followed by the associated (2-dimensional) feature vector. When you visualize the data set, you should see this:



2. Assuming a K-Nearest Neighbors classifier with the L_2 distance metric:

(5) (a) From visual inspection of this dataset (figure above), qualitatively sketch the ROC you would expect to represent performance on this dataset by a KNN classifier with $k = 5$.



(5) (b) i. Find the cross-validated ROC when a KNN classifier with $k = 5$ is applied to this data, using the folds specified as part of the dataset, and compare that ROC to the ROC found via incestuous training (training on the full training set followed by testing testing on the full training data set).

ii. Explain why the ROC found via cross-validation differs from the ROC you (qualitatively) expected. (Decision statistic surfaces for the two folds individually will likely be part of your explanation.)

(5) (c) Describe how you would modify the cross-validation on this dataset to achieve ROCs that provide more reasonable representative estimates of classifier performance for a KNN classifier with $k = 5$?

(5) (d) Implement your proposed modification to the cross-validation, and compare your cross-validated ROC when a KNN classifier with $k = 5$ is applied to this data to the ROC found via incestuous training and testing.

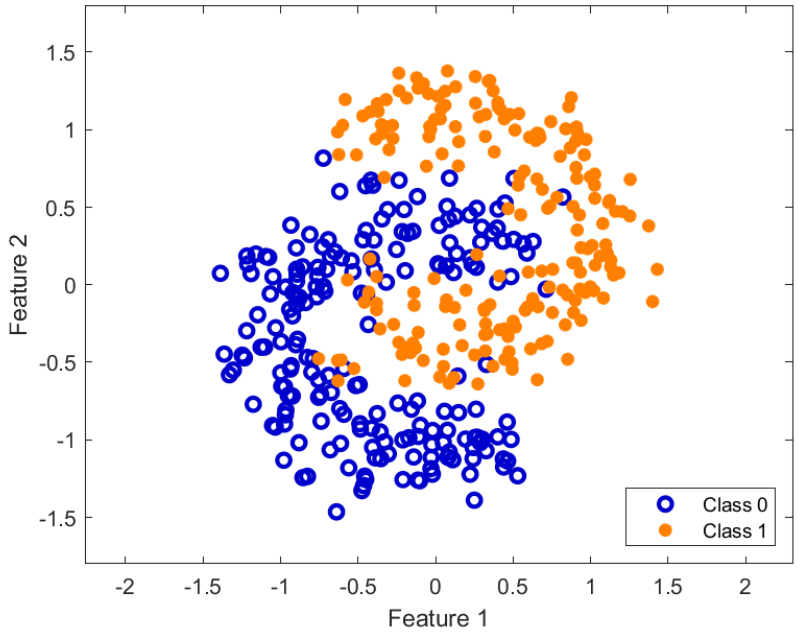
(5) (e) Submit a PDF print-out of your code for this section (Exploring Cross-Validation within the Context of Classification). (Submitting a URL for a cloud-based repository is insufficient.)

Applying KNN to Explore Bias-Variance Trade-off in Classification

Make sure you are able to apply K-Nearest Neighbors and that you have flexibility to specify how you want to measure the distance between points.

Regardless of whether you choose to write your own functions or leverage functions that may be available through standard Matlab or Python packages, it is critical that you understand how the function(s) you are using work so you can effectively apply them to suit your needs and correctly interpret the results they provide.

The following questions concern a data set that is provided as a `csv` file, `dataSetHorseshoes.csv`. This csv file is organized such that each row contains the true class (either 0 or 1) followed by the associated (2-dimensional) feature vector. When you visualize the data set, you should see this:



3. Suppose you are using a K-Nearest Neighbors classifier with the L_2 distance metric.

(5) (a) On the visualization of the `Horseshoe` data (shown above), sketch what you would consider to be an ideal boundary between the two classes (a decision boundary that balances correct classification on this data set and expected generalizability to a new data set).

(5) (b) How do you expect the decision boundary to deviate from the ideal boundary you sketched above as k becomes very, very small (tends toward 1)?[4] How do you expect the decision boundary to deviate from the ideal boundary you sketched above as k becomes very, very large (tends toward the number of observations, N)?

[4] You may also sketch both this and the following decision boundary on the same visualization of the `Horseshoe` data as you sketched your ideal boundary if you think that would be helpful to see them all on the same plot... if you do this make sure to clearly label the decision boundaries you sketch!

4. Suppose you are using a K-Nearest Neighbors classifier with the L_2 distance metric.

(5) (a) Visualize the decision statistic surface with both the training data and the majority vote decision boundary[5] superimposed on top of the decision surface for KNN with $k = 1$, $k = 5$, $k = 31$, $k = 91$, $k = \frac{N}{2} - 1$, and $k = N - 1$.[6] (N is the total number of training observations.)

(5) (b) Explain why the KNN majority vote decision boundary deviates from your ideal decision boundary as k gets small ($k \to 1$) and as k get large ($k \to N$), and how these deviations may be viewed as manifestations of "bias" or "variance".

5. Suppose you are using a K-Nearest Neighbors classifier with the L_2 distance metric.

(5) (a) Plot the ROCs for $k = 1$, $k = 5$, $k = 31$, $k = 91$, $k = \frac{N}{2} - 1$, and $k = N - 1$ when the KNN classifier is trained and tested on the training data, and find the max P_{cd} for each value of k.

(5) (b) Based only on these ROCs, and nothing else, what value of k do you recommend to maximize P_{cd}? Why?

6. Suppose you are using a K-Nearest Neighbors classifier with the L_2 distance metric.

(5) (a) Plot the ROCs for $k = 1$, $k = 5$, $k = 31$, $k = 91$, $k = \frac{N}{2} - 1$, and $k = N - 1$ when the KNN classifier is trained on the training data and tested on separate testing data provided in the `csv` file, `dataSetHorseshoesTest`[7], and find the max P_{cd} for each case.

(5) (b) Based only on these ROCs, and nothing else, what value of k do you recommend to maximize P_{cd}? Why?

7. Suppose you are using a K-Nearest Neighbors classifier with the L_2 distance metric.

(5) (a) i. Plot $\min P_e \, (= 1 - \max P_{cd})$ as a function of N/k for a sampling[8] of values of k between 1 and 399 for the 3 cases: 1) testing on the training data, 2) testing on separate testing data, and 3) 10-folds cross-validation on the training data.[9]

ii. Explain how the graph of cross-validated performance should be interpreted, and how this graph illustrates the principle of bias-variance trade-off.

(5) (b) Based on this graph, what value of k do you recommend to maximize P_{cd}? Assume you do not have separate test data available to you, how would you select a value of k to recommend to maximize P_{cd}?

(5) 8. Submit a PDF print-out of your code for this section (Applying KNN to Explore Bias-Variance Trade-off in Classification). (Submitting a URL for a cloud-based repository is insufficient.)

[5] In Matlab, the `contour` function may be helpful for finding a desired decision boundary.

[6] It would be highly unusual to choose k so high in practice, but looking at the full range of values for k, and both boundary conditions on k ($k = 1$ and $k = N$) in particular, can help develop insight. We are looking at $k = N - 1$ instead of $k = N$ so there will be a majority of neighbors in one of the classes.

[7] This csv file is organized such that each row contains the true class (either 0 or 1) followed by the associated (2-dimensional) feature vector.

[8] You choose the values of k where you want to sample this curve.

[9] The flexibility of a KNN classifier is proportional to N/k; $k = 1$ provides the greatest flexibility and $k = N$ provides the least flexibility.