Homework 1

Due: 11.59pm on Friday, January 24

Submission instructions: Submit one write-up per group on Gradescope. Group size should not be bigger than 3 people. You can work by yourself on the homework.

IMPORTANT: Write names of everyone that worked on the assignment on the submission.

If you have not used gradescope before:

- This PDF guide explains how to create high-quality scans of your solutions
  http://gradescope-static-assets.s3-us-west-2.amazonaws.com/help/submitting_hw_guide.pdf
- This video guide also exlains how to upload the PDF of your solution
  https://www.youtube.com/watch?v=-wemznvGPfg

Please do not bring printouts of your solutions to the classroom.

1 Question 1

For each one the ten statements below say whether they are true or not and explain why.

1. As one increases k, the number of nearest neighbor, in a kNN classifier,

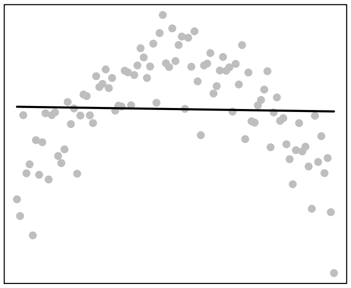
   (a) the bias of the classifier will increase;

   (b) the variance of the classifier will increase;

   (c) the misclassification rate on the training dataset will increase;

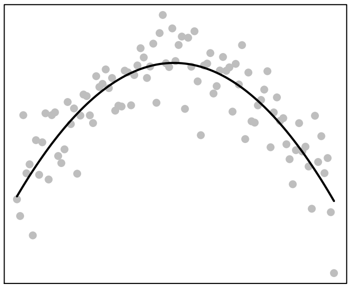
   (d) the misclassification rate on a test dataset will increase.

2. Consider the three line regression fits to the gray points plotted below.



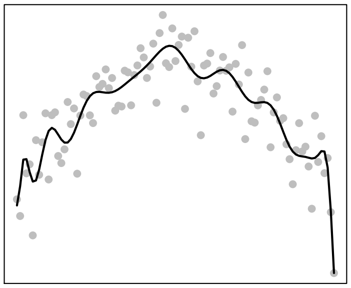
$\hat{y} = \beta_0 + \beta_1 x$

(1)



$\hat{y} = \beta_0 + \beta_1 x + \beta_2 x^2$

(2)



$\hat{y} = \beta_0 + \beta_1 x + \cdots + \beta_{20} x^{20}$

(3)

   (a) The estimate in (2) has a higher variance than the estimate in (1).

   (b) The estimate in (2) has a higher bias than the estimate in (3).

   (c) The estimate in (3) has the smallest training error.

(d) The estimate in (1) has the smallest test error.

3. Misclassification rate of a classifier evaluated on a validation set will never be smaller than the one evaluated on the training set that is used to build the classifier.

4. k-fold cross-validation provides an unbiased estimate of the predictive error of the models.

2 Question 2

In this question, you will run a simulation study to explore the bias-variance trade-off in more depth. The exercise will walk you through prediction using linear regression versus k-NN. You will discover under what circumstances one outperforms the other (in terms of the test error). In the end you will reproduce something similar to what you see in figures 3.17 through 3.20 in Section 3.6 of *Introduction to Statistical Learning*.

1. You will start by exploring a scenario where the true relationship between x and y is linear. You will generate data from the linear model

$$y = f(x) + \epsilon$$

where $f(x) = 1.8x + 2$. To create a synthetic training set, make 100 independent draws of x from a uniform distribution on $[-1, 1]$ to form $\{x_1, \cdots, x_{100}\}$. Similarly draw random noise ϵ from a standard normal distribution to have $\{\epsilon_1, \cdots, \epsilon_{100}\}$. For each $i \in \{1, \cdots, 100\}$, generate y_i from the linear model where $y_i = 1.8x_i + 2 + \epsilon_i$. Now, repeat the above process and generate additional 10,000 observations that will form the test set. Note that we are constructing a huge test set in order to have an accurate out-of-sample MSE. You will use this simulated dataset to try out and compare different models.

2. Create a scatter plot of y vs x. In the same figure, draw the true relationship in black solid line.

3. Using ordinary linear regression, find a relationship between y and x of the form

$$y = b_0 + b_1 \times x + e$$

using the training data you simulated. On the same plot from the last question, draw a blue dashed line that is the least squares fit to the data.

4. You may find that the linear regression line provides a very good estimate of $f(X)$. Now, use k-NN to find the relationship between y and x. You should experiment with $k = 2, 3, \cdots, 15$ to see how model complexity affects prediction accuracy. On one plot, redraw the scatter plot and the true relationship, but this time overlay it with predicted fit using k-NN with $k = 2$. On a juxtaposed graph, do the same for $k = 12$.

5. Plot the test set mean squared error using k-NN against $\log(1/k)$ for $k = 2, 3, \cdots, 15$. On the same graph, draw a horizontal dashed line that represents the test set mean squared error using linear regression. Which model performs the best? Comment on the relative performance of linear regression and k-NN with different values of k.

6. Redo 1-5, but consider a different data generating process where the true relationship between x and y is near, but not perfectly linear. Precisely, simulate data from the following model

$$y_i = \tanh(1.1 \times x_i) + 2 + \epsilon_i.$$

How does your conclusion in 5 change?

7. Consider yet another data generating process where the true relationship is strongly non-linear. Simulate data from the following model

$$y_i = \sin(2x_i) + 2 + \epsilon_i.$$

What can you say about the relative performance of the two methods now?

8. You might suspect, from your previous results, that in real world when none of the relationship could be linear, it would always pay off using k-NN over linear regression. Examine this hypothesis in the situation with more than 1 variable. Stay with the same true relationship from the model in 7, add additional noise variables that are not predictors of y. In particular, consider the model

$$y_i = \sin(2 \times x_{i1}) + 2 + 0 \times x_{i2} + \cdots + 0 \times x_{ip} + \epsilon_i$$

where p ranges from 2 to 20. Additional, noisy $x_{i2}, \ldots, x_{ip}$ variables are drawn from independent standard normal distributions. Create a plot for every p taking values in $1, 2, 3, \cdots, 20$. Each plot should have the test set MSE against model complexity $\log(1/K)$ and a dashed horizontal line showing test set MSE for best linear fit. Briefly explain the result of the simulation.

OPTIONAL BONUS QUESTION: Suppose that instead of 100 training samples, you had 1,000 training samples. Would that change conslucions you made above? Think about how the range of values of k for which k-NN does bettter that linear regression would change. What does having a large traing set allow you to do?

3 Question 3: Recommender System

In this homework, you will analyze a part of Amazon product review data collected by Julian McAuley http://jmcauley.ucsd.edu/data/amazon/

The file *videoGames.json.gz* contains data. The starter script will load data for you.

There are 100,000 reviews. For each review we have the following information:

- **itemID** The ID of the item. This is a hashed product identifier from Amazon.
- **reviewerID** The ID of the reviewer. This is a hashed user identifier from Amazon.
- **rating** The rating that the reviewer gave to the item.
- **helpful** Helpfulness votes for the review. This is a list with two subfields, 'nHelpful' and 'outOf'. The latter is the total number of votes this review received, the former is the number of those that considered the review to be helpful.
- **reviewText** The text of the review.
- **summary** Summary of the review.
- **unixReviewTime** Time of the review in seconds since 1970.
- **reviewTime** Plain-text representation of the review time.
- **category** Category labels of the product being reviewed.

The starter script will create a rating matrix from this information. As you will see, the starter script keeps only users that have rated more than 2 video games and video games rated by more than 3 users. This is primarily done for computational reasons, so that you do not have to wait too long before getting an answer.

Your tasks:

- Find the user that has rated the most amount of video games.
- Which video games has been rated by the most amount of users?
- Find the user that is most similar to "U141954350".
- Recommend a video game to the user "U141954350". For example, you may find a video game that the user "U141954350" has not bought yet and you expect that he would rate it high. Explain the recommendation strategy you used.