DATA11002 Introduction to Machine Learning, Fall 2017
Exercise set 1

Due November 8th–10th.

Pencil-and-paper problem

Problem 1 (2+2+2+2 points) Let $X_1, \ldots, X_n$ be independent and identically distributed (i.i.d.) binary outcomes distributed according to Bernoulli distribution, $\mathrm{Ber}(p)$, so that the probability that each of them takes value 1 is given by $E[X_i] = p$. *Hoeffding's inequality* is a useful result in probability that tells us that the probability that the total number of outcomes with value 1 divided by n is not too far from its expectation, which is $E[\frac{1}{n}\sum_{i=1}^{n} X_i] = \frac{1}{n}\sum_{i=1}^{n} E[X_i] = p$:

$$\Pr[|p - \frac{1}{n}\sum_{i=1}^{n} X_i| > \epsilon] \leq 2\exp(-2n\epsilon^2).$$

(You can think of the X_i indicating whether a classifier makes a correct prediction in a binary classification task. Hoeffding's inequality guarantees that observed performance is going to be a close to the true accuracy, p, of the classifier with high probability.)

(a) (2 points) Solve for the value of ϵ for which the above upper bound equals α. For example, with sample size $n = 10$ and $\alpha = 0.05$, this provides a bound that guarantees that with 95 % probability, the observed number of occurrences of $X_i = 1$ is within the interval $[n(p - \epsilon), n(p + \epsilon)]$. Evaluate the width of this interval for $n = 10, 100$, and 1000. (Note that for different n, you will get different values of ϵ as well.)

(b) (2 points) The Hoeffding bounds are typically quite conservative: the obtained confidence intervals are actually so wide that they contain the true value with *too* high confidence (higher than $1 - \alpha$ probability). To see that this is indeed the case, simulate the 10000 repetitions of the experiment by drawing binomial outcomes (which correspond to the number of 1's in n i.i.d. Bernoulli r.v.s) using `rbinom(10000, n, p)` with parameter values $p = 0.5, 0.9, 0.99$, and checking how many times the outcome is within the interval you calculated in the previous item.

The theory says that with $\alpha = 0.05$, for example, the outcome should be inside the interval at least 95% of the time. Repeat again with $n = 10, 100, 1000$.

(c) (2 points) The union bound (or Boole's inequality) is another simple and nice result in probability. It simply states that if there are a number of events, $A_1, \ldots, A_k$, with probabilities $P(A_1), \ldots, P(A_k)$, then the probability that at least one of them occurs is upper bounded by

$$P(\cup_{i=1}^{k} A_i) \leq \sum_{i=1}^{k} P(A_i).$$

Consider now a set of k classifiers, each of which is associated with a separate set of n Bernoulli trials for which we can apply Hoeffding's inequality. Use the union bound together with Hoeffding's inequality to bound the probability that *for any* of the classifiers, the difference between the observed number of outcomes with value 1 divided by n and its expectation, p_i, is greater than ϵ. (*Hint:* Here $P(A_i) = \Pr[|p_i - \frac{1}{n}\sum_{j=1}^{n} X_j| > \epsilon]$.)

Again, solve for ϵ for which the resulting probability upper bound equals some α. What does this tell you about the effect of k on the resulting guarantee about the observed vs real accuracy? Evaluate the width of the interval[1] $[n(p - \epsilon), n(p + \epsilon)]$ for $n = 10, 100$, and 1000 when $k = 1, 10$, and 100.

(d) (2 points) Again check whether the obtained bounds hold by simulating 10000 repetitions of the process: In each experiment, draw k binomial values from $\mathrm{Bin}(n, p)$ and check whether *all* of them are within the interval you calculated in the previous item.

[1] Note that while the accuracy, p, may differ from one classifier to another, the *width* of the interval will only depend on the sample size n and α.

Computer problems

Problem 2 (4 points)

Exercise 8 on p. 54 of the book.

Problem 3 (4+4+4 points)

In this problem, we will test linear regression on a simple synthetic dataset. We will use the following polynomial as the underlying target function

$$y = f(x) = 2 + x - 0.5\,x^2. \tag{1}$$

First, randomly sample 30 points x_i from the uniform distribution (function `runif` in R) on the interval $[-3, 3]$. Then, randomly generate the y_i using

$$y_i = f(x_i) + \epsilon_i, \tag{2}$$

where f is as defined above, and the ϵ_i are i.i.d. normal random variables (function `rnorm` in R) with zero mean and standard deviation 0.4. The resulting 30 pairs (x_i, y_i) is your data set for this exercise.

(a) (4 points) First, let's fit polynomials of order 0 to 10 to this dataset using linear regression, minimizing the sum of squares error. That is, fit functions of the form

$$\hat{y} = \sum_{p=0}^{K} w_p x^p \tag{3}$$

with $K = 0, \ldots, 10$ to the data. For instance, for $K = 4$ the polynomial to fit is

$$\hat{y} = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4. \tag{4}$$

For each of the 11 values of K, produce a separate plot showing the datapoints (x_i, y_i) and the fitted polynomial. (Plot the polynomial as a curve, in the full interval $[-3, 3]$, overlayed on the scatterplot of the points.) You should see that as the order of the polynomial K increases, the curve comes closer and closer to fitting all the datapoints.

Calculate the mean squared error (MSE) on the training data:

$$\text{MSE} = \frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n}, \tag{5}$$

and compare the MSE of the fitted different order models.

(b) (4 points) Next, generate 1000 more data points from the same polynomial and use them as a test set to evaluate the predictive performance of the fitted models. (*Hint:* The `predict` function that takes as arguments the fitted model and new data points will probably come in handy.)

Plot both the training MSE and the test MSE as a function of the polynomial order. What do you notice?

(c) (4 points) Finally, let's use a technique called 10-fold cross-validation to automatically select a model based on the 30 training examples we have. Divide the dataset into 10 equal-sized subsets (i.e. 3 datapoints in each subset), and, for each value of $K = 0, \ldots, 11$ and each data subset $j = 1, \ldots 10$, use all the data except the data in subset j to fit the polynomial of order K, and compute the resulting sum of squared errors on subset j. For each value of K, sum together the squared errors coming from the different folds j. Plot these results with K on the horizontal axis, and the sum of squared errors on the vertical axis. How does this function behave? Does the cross-validated error improve with increasing K? Which K gives the minimum error?