MTAT.03.227 Machine Learning
Spring 2014 / Exercise session XIII
Nominal score: 10p
Maximum score: 15p
Deadline: 28th of May 8:15 EET

1. In the lecture, we discussed that the prediction error decomposes usually into three basic components: model variance, model bias and noise. The following exercise illustrates how these error components emerge when we consider a simple one-dimensional regression task. For clarity, we fix locations of x coordinates in the training and test sets, as the latter makes it easier to visualise the results. Analogous treatment can be carried out when the training set is sampled according to some other distribution. Let training set $\mathcal{S}$ consist of pairs (x_i, y_i) where the values of x_i form the grid $(-2.0, -1.8, \ldots, 1.8, 2.0)$. Let the test set $\mathcal{T}$ consist of pairs (x_i, y_i) where the values of x_i form the grid $(-1.9, -1.7, \ldots, 1.7, 1.9)$. For both datasets, the values of y_i are computed according to the model

$$y_i = x^3 - x^2 + \varepsilon_i$$

where ε_i are drawn independently form $\mathcal{N}(0, \sigma)$. We recommend the to set $\sigma = 1$ although you could also try different values to see what is the impact of noise in linear and polynomial regression.

   (a) Since the sampling procedure is noisy, each time we get a different training set and thus obtain different predictors. To visualise this concept, generate 100 training sets on the grid points specified above. Use `lm` to fit linear, cubic and 8th degree polynomial models. Plot the first 10 linear, cubic, and 8th degree polynomial prediction functions. Compute the predictions of all functions on the test set points and visualise the results. Draw mean values and 90% quantiles. Interpret results. How reliable are the predictions of different models? **(1p)**

   (b) Given the description of data distribution on the test set find a linear, cubic and 8th degree polynomial predictor that minimises the risk on the test set. Interpret the results. **(1p)**

   Hint: Note that, for this particular exercise, the set of future observations is finite—a sample from the test set—and thus the risk $R(f)$ can be expressed as an average

$$R(f) = \frac{1}{|\mathcal{T}|} \cdot \sum_{(x_i, y_i) \in \mathcal{T}} \mathbf{E}[(f(x_i) - y_i)^2]$$

   where the expectation is taken over all possible error values. In the standard conditions, the risk $R(f)$ would be a two-dimensional integral to factor in also the probability of different x values.

Hint: Recall that the mean value is the best predictor if we want to minimise mean square error between predictor and measured value.

Hint: If the first hint does not help recall that linear regression is asymptotically strongly consistent and thus given enough data points the regression outputs almost optimal coefficients.

(c) Study how the predictors vary depending on the draws of the training set. For that draw 100 training sets and train corresponding linear, cubic and 8th order models. Observe their predictions on the grid $(-2.0, \ldots, 2.0)$. Compute average predictor for each model class and compute lower and upper 5% quantiles for each grid point. Visualise the results by showing the mean of predictions and prediction quantiles. Inspect visually which of the models obtains the best bias-variance tradeoff. Do the prediction averages really converge to the function with minimum risk in the model class? (**1p**)

(d) Verify empirically that the bias-variance-noise decomposition holds empirically. First, estimate the expected mean square error of a cubic model. Second, compute bias term as

$$\mathsf{Bias}(f_*) = \frac{1}{|\mathcal{T}|} \cdot \sum_{(x_i,y_i)\in\mathcal{T}} (f_*(x_i) - f_\circ(x_i))^2$$

where $f_\circ(x) = x^3 - x^2$ is the true dependence between x and y and $f_*(x)$ is a cubic function that minimises the the bias. Third, estimate the variance

$$\mathsf{Variance} = \frac{1}{|\mathcal{T}|} \cdot \sum_{(x_i,y_i)\in\mathcal{T}} \mathbf{E}[f_*(x_i) - f_{\mathcal{S}}(x_i))^2]$$

by sampling enough training sets $\mathcal{S}_i$ to replace mathematical expectation with a simple average. Fourth, estimate the noise term

$$\mathsf{Noise} = \frac{1}{|\mathcal{T}|} \cdot \sum_{(x_i,y_i)\in\mathcal{T}} \mathbf{E}[(f_\circ(x_i) - y_i)^2]$$

by sampling enough outputs y_i to replace mathematical expectation with a simple average. Report all three estimates and verify that their sum is roughly equal to the expected mean square error of the cubic model. (**1p**) You get an extra point if you manage to visualise the result by emphasising the randomness of all estimations.

2. The precise knowledge of the model variance gives a good understanding whether a machine learning method is applicable or not. Unfortunately, you have only one data set for training and validation so you cannot use the simulation method described in the first exercise.

(a) However, you can use bootstrapping to reliably simulate the distribution of training sets. Repeat the sub-exercise (c) from the first exercise using bootstrapping. Compare the results with the original estimates. Is the bootstrapping a reasonable method for estimating model variance? (**2p**)

(b) As an alternative to bootstrapping is k-fold cross-validation. Namely, you can train the model on $k-1$ folds and to simulate sampling of different training sets. Repeat the sub-exercise (c) from the first exercise using cross-validation. Compare the results with the original estimates. Is the cross-validation a reasonable method for estimating model variance? (**2p**)

(c) Bootstrapping and cross-validation are commonly used for estimating the error of a machine learning method. What do these methods estimate? Do they estimate the average error of the model class such as linear or cubic regression or do they estimate the test error of a particular predictor? Explain why computing the average optimism is useful? (**2p**)

3. This exercise illustrates that rejection bias is not merely a theoretical concept and thus SLT bounds do not show what we hope unless we implicitly assume that the problem instance belongs to a solvable problem class. Here, we consider a very simple machine learning algorithm. The algorithm will count the labels of the training set and after that always output the label which occurred more than the other in the training set.

   (a) Use the quantiles of binomial distribution to obtain a simple bound on the optimism. For that you have to estimate

   $$\Pr\left[R(f_i) - R_m(f_i) \geq \varepsilon\right] \leq 2.5\%$$

   for constant function $f_0 \equiv 0$ or $f_1 \equiv 1$. The distribution of input data $\boldsymbol{x}$ is irrelevant. The bound can be expressed by considering only the binomial distribution of labels y. Use this result to obtain a union bound

   $$\Pr\left[\exists i : R(f_i) - R_m(f_i) \geq \varepsilon\right] \leq 5\%$$

   Tabulate the corresponding values of ε for $m \in \{3, 5, 7\}$. (**1p**)

   (b) Consider the hard problem instance where the label is generated randomly. Draw $10,000$ training set samples and find out on how many cases the following events occur: (1) the training error is below 20%; (2) the optimism is below ε form the previous sub-task. Draw the corresponding 2×2 contingency table and interpret the result (**1p**)

   (c) Consider the hard problem instance where the label is generated by choosing label one with probability 90%. Draw $10,000$ training set samples and find out on how many cases the following events occur:

(1) the training error is below 20%; (2) the optimism is below ε form the previous subtask. Draw the corresponding 2×2 contingency table and contrast the result with previous subtask. (**1p**)

4. This exercise illustrates the main concepts of statistical learning theory. First, assume that the input has uniform distribution over the square $[-1, 1] \times [-1, 1]$. Secondly, consider linear classifiers and linear classifiers with second order terms, i.e., you fit $y \sim x_1^2 + x_1x_2 + x_2^2 + x_1 + x_2 + 1$.

   (a) Determine the VC-dimension empirically. For that consider configurations where input points form regular convex polygons. Find the largest $n \in \{3, 4, 5, 6\}$ for which all possible labellings are achievable for both classifier classes. Formally, this will be the lower bound on VC-dimension but in the context of current exercise we take it as a precise value. Use the functions `VisualiseDescisionRegions` to visualise classifier configurations. (**1p**)

   (b) The main result in statistical learning theory claims that the difference between the test and training error depends on the VC-dimension. The higher is the VC-dimension of the the classifier, the higher the potential difference is. Verify this claim empirically by considering the data distribution which assigns random labels to each data point. Then the risk for any classifier is 50% and thus you can compute optimism in closed form. Experiment with different number of input samples $m \in \{3, 4, 5, 10, 25, 50, 100\}$. Do 1000 experiments for each value of m and draw corresponding box-plots. Interpret results. Does the SLT claim hold in our example? (**1p**)

   (c) Repeat the same experiment as in the previous sub-task when labels are assigned by choosing the label 1 with probability 95%. Do you observe the same behaviour on the optimism. To be more formal, compute 95% quantile for optimism values and draw corresponding graphs for the previous subtask and the current subtask. Interpret results. Does the graph of 95% quantiles strongly depend on the problem class or not? How loose must a SLT bound on 95% quantiles be if it must hold for any problem class? (**1p**)

5. Recall that in order to derive bounds on the risk of a classifier the statistical learning theory considers the distribution of optimism—difference between training and test errors over the sets of same size. The former allows to reduce the infinite class of potential classification rules into a finite set of evaluations.

   (a) To illustrate this concept, consider four data points $\boldsymbol{x}_1, \ldots \boldsymbol{x}_4$ that are either sampled uniformly from the square $[-1, 1] \times [-1, 1]$ or placed at corners. Next, draw randomly coefficients for the model $y \sim x_1x_2 + x_1 + x_2 + 1$ and classify the model into eight classes depending which points are put into same class. Continue with sampling until you have three models for each class. Visualise the result by drawing a plot with 3×3 subplot where each subplot contains decision borders for each class. (**2p**)

Hint: Use `contour(zmatrix, levels=0, add=TRUE)` to draw the decision borders. If it takes too much time to get representatives for all eight classes you can seek them directly with `glm`.

(b) Draw eight input points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_8$ randomly form the square $[-1, 1] \times [-1, 1]$ and set corresponding labels $y_1, \ldots, y_8$ to zero. Find out how many splits into training and test there exists a classifier such that the training error is zero while the test error is one, i.e., how many splits are such that the corresponding unique labelling is implementable. Estimate this empirically by looping through all 70 splits. (**1p**).

(c) Count the number of splits for which the there exists a classifier such that the training error is zero while the test error is $3/4$. Note that there now four potential labellings. Study whether each such possibility is actually realised, i.e., consider what happenes if you train the classifier on the training set. (**1p**)

6. The number of accessible labellings are is bounded by VC-dimension d. More precisely Sauer's Lemma states

$$G(n) \leq \sum_{i=0}^{d} \binom{n}{i} \leq n^d$$

where d is the VC-dimension of the classifier. Use that fact to estimate VC dimension of a neural network with 20 hidden sigmoid units. For that sample random configurations and labellings for each n until the classifier fails 20 times. Next use the Sauer's lemma to bound the probability of individual failure and find such d value that the procedure would return d that is smaller than the actual VC-dimension with probability 5%. (**3p**)