582631 Introduction to Machine Learning, Fall 2016
Set 5

Due December 8th–9th. NB: Deadline for returning solutions my email (in case you can't attend a session) is 12:15 on Friday.

Problem 1 (2 + 3 + 3) Decision trees

This is a continuation of last week's decision tree exercise (Prob. 4.3).

(a) (2 points) As it turned out, it is hard to get non-zero gain when impurity is defined as misclassification rate (p. 29 of the slides of Lectures 7–8).

Evaluate the gain of the same splits as you chose last week using entropy and Gini measures of impurity.

(b) (3 points) Consider the plot on p. 30 of the slides (Lecture 7–8), where misclassification rate, Gini, and entropy are plotted as a function of the probability of one class, when there are two class values (binary classification).

Consider a split that produces subsets D_1 and D_2, with relative sizes $\alpha = |D_1|/|D|$ and $1 - \alpha = |D_2|/|D|$. Let $p_0(D_i)$ denote the probability (or relative frequency) of class $Y = 0$ in subset D_i, and let $Q(D_i)$ denote the corresponding impurity values for $i \in \{1, 2\}$.

Suppose that the majority class is the same in both D_1 and D_2, i.e., both subsets correspond to points on the same side of the "hump" at $p_0 = 1/2$ in the plot. (So for example, if class 0 is the majority class, then $p_0(D_i) > 1/2$ for both $i \in \{1, 2\}$, and we are on the right half of the figure.)

Prove that under these circumstances, if we use misclassification rate as a measure of impurity, the gain is exactly zero.

Hint: Draw a figure and use it to help your reasoning: Include the (x, y) points $(p_0(D_1), Q(D_1))$, $(p_0(D_2), Q(D_2))$, and $(p_0(D), Q(D))$ on the graph. For example, suppose the whole set D has size $|D| = 168$ with class distribution $p_0(D) = k/|D| = 130/168 \approx 0.77$ where $k = 130$ is the number of goals in set D. Further, suppose that the subsets D_1 and D_2 are of respective sizes $|D_1| = 52$ and $|D_2| = 116$, and that the respective class distributions are $p_0(D_1) = k_1/|D_1| = 49/52 \approx 0.94$ and $p_0(D_2) = k_2/|D_2| = 81/116 \approx 0.70$. Observe that we have $0.70 \cdot 116/168 + 0.94 \cdot 52/168 \approx 0.77$, i.e., the class distribution in D can be expressed as a linear combination of the class distributions in D_1 and D_2.

Figure out how the position of the third point depends on the first two in general. You can use concerete example values to come up with a hypothesis and then prove (or disprove!) it formally.

(c) (3 points) Still on the same issue: Prove formally that when using the Gini index or the entropy, the gain is always positive (> 0) unless $p_0(D_1) = p_0(D_2)$.

Hint: Jensen's inequality. You can use the fact that both the Gini index and the entropy are convex functions without proof.

The point in these exercises is to appreciate the fact that even if the ultimate goal is to minimize misclassification error, other measures may be better at directing the tree building process towards useful splits — both the entropy and the Gini index favour splits that make the probabilities $p_0(D_1)$ and $p_0(D_2)$ as different as possible.

Recall that when building a decision tree, we usually use entropy or Gini, while the pruning stage usually uses the misclassification rate which matches the actual target (assuming that is indeed the misclassification rate).

(Exercises continued on the next page...)

Problem 2 (3 + 2 + 3 + 2 points) Naive Bayes, Bayes error, and interactions

Here too we continue last week's exercises: this time Problem 4.2 (with naive Bayes and logistic regression on discrete features X_1 and X_2).

What happened there is that both classifiers (NB and log.reg.) produced asymptotic error rate 0.4 as the sample size was increased, which was not particularly interesting.

(a) (3 points) Repeat the steps in last week's exercise 2 but evaluate the performance with the *logarithmic loss* (log-loss) $-\log_2 P(Y = c \mid X_1 = x_1, X_2 = x_2)$ instead of the zero–one loss. In other words, produce probabilistic predictions for all the test instances and let the negative logarithm of the probility of the actual (correct) class of the test instance be the loss. For example, if the model predicts $P(Y = 0 \mid \mathbf{x}) = 0.5$, $P(Y = 1 \mid \mathbf{x}) = 0.25$, and $P(Y = 2 \mid \mathbf{x}) = 0.25$, and the actual class value is 1, then the log-loss for that test instance is $-\log_2 0.25 = 2$.

For logistic regression models produced using the function `multinom` from package `nnet`, you can extract probabilistic predictions by using argument `type="probs"` in the `predict` function.

Interpret your findings in light of the Ng & Jordan paper from Set 2 of exercises. You can increase the sample size to make sure the values have converged.

Hint: The sum of logarithmic losses should be a bit less than 10000 (i.e., slightly less than 1.0 units (bits) per test instance).

(b) (2 points) Last week you were told to use the naive Bayes classifier even though we could tell that the features are *not* conditionally independent given the class.

This time, use the true class-conditional probabilities $P(\mathbf{X} \mid Y)$ as specified in the tables in last week's exercises to implement a classifier. Thus, for example, you should use $P(X_1 = 1, X_2 = 1 \mid Y = 2) = 0.0$ instead of using a product of the form $P(X_1 \mid Y) \times P(X_2 \mid Y)$.

The classifier that is obtained from the true distribution is called the *Bayes classifier* — see p. 37 onwards in the textbook. It has the lowest possible (expected) error rate among *all* classifiers. Usually such a thing is only of theoretical interest since in practice we almost never have the true distribution at hand.

Calculate the error rate of the Bayes classifier on the same test set (of size 10 000) as the naive Bayes classifier. Plot the errors in the same graph; use dashed line for the Bayes error following the convention in the textbook.

Hint: The Bayes error should be between 0.3 and 0.4.

(c) (3 points) Calculate the Bayes error by hand.

Hint: The probability you need to evaluate is

$$\Pr[h(X_1, X_2) \neq Y] = \sum_{\substack{c \in \{0,1,2\} \\ x_1 \in \{0,1\} \\ x_2 \in \{0,1,2\}}} P(Y = c) P(X_1 = x_1, X_2 = x_2 \mid Y = c)\, I_{[h(x_1,x_2) \neq c]}$$

where $h : \{0, 1\} \times \{0, 1, 2\} \to \{0, 1\}$ is the Bayes classifier that maps feature vectors to the most probable class value, and $I_{[\cdot]}$ is an indicator function.

(d) (2 points) Include an interaction between the features in the logistic regression model by writing
`multinom(Y ~ X1 * X2, data = train)`
where the `*` implies that an interaction should be included in the model, instead of `+` which assumes only additive effects.

Evaluate the misclassification rate and compare it to the performance of the naive Bayes classifier, the logistic regression model without interactions, and the Bayes error (dashed line).

Hint: The error rate of the logistic model with interactions should converge to the Bayes error as the sample size grows.

(Exercises continued on the next page...)

Problem 3 (2 + 4 points) SVMs

To prepare for this exercise, you should follow the steps in the Lab on pages 359–364 on the textbook. (You don't have to do the part on ROC curves.)

(a) (2 points) Generate $n = 200$ data points from the same source as in last week's Exercise 1 (two independent Gaussians with zero mean and variances 1 and 16 respectively, uniform class distribution). You'll get roughly 100 instances from each class.

Apply a support vector classifier (an SVM with a linear kernel) with a couple of different tuning parameters (argument `cost`), and visualize the decision boundary with `plot(fit, data=data)`.

Confirm that the linear classifier has no hope of performing well in this case.

Then try a polynomial kernel of degree 2 and an RBF kernel to see that performance improves. Experiment with different tuning parameters (see the documentation of function `svm` and the Lab in the textbook for information about the parameters).

Secondly, add the transformed features $X_3 = X_1^2$, $X_4 = X_2^2$ (squared features) to the data and try again. Note that you will not be able to visualize the classification boundary but you can use `predict` to evaluate the predictions on the training data to verify that even a linear kernel will now perform better.

Why is there an improvement?

(b) (4 points) Exercise 8 on p. 371 of the textbook.