Course: Statistical methods for machine learning

February 16, 2010

Case 2: Supervised learning

This assignment is based on parts of chapter 1-8 from C. M. Bishop *Pattern Recognition and Machine Learning*. The goal is for you to get familiar with the supervised learning task and to get practical experience using a number of methods such that after this assignment you will be able to use and interpret the results of the methods in other contexts.

You have to pass this and the following mandatory assignments in order to be eligible for the exam of this course. There are in total 3 mandatory pass/fail assignments on this course, which can be solved individually or in groups of no more than 3 participants. The course will end with a larger exam assignment which must be solved individually and is graded (7- point scale).

The deadline for this assignment is Tuesday 9/3 at 23:55. You must submit your solution electronically via the Absalon home page. Go to the assignments list and choose this assignment and upload your solution prior to the deadline. If you choose to work in groups on this assignment you should only upload one solution, but remember to include the names of all participants both in the solution as well as in Absalon when you submit the solution. If you do not pass the assignment (after having made a serious attempt), you will get a second chance of submitting a new solution.

A solution consist of:

- Your solution code (Matlab / R / Python scripts) with comments about the major steps involved in each Question (see below).
- Notes detailing your answer to the non-programming questions, which may include graphs and tables if needed (Max 2 page text).

1 Regression

In the regression part of the case we will study regression with linear models (as explained in Bishop Sec. 3.1) and we will experiment with different basis functions. The data set we will consider is a real-world data set which contains a set of estimates of the percentage of body fat determined by underwater weighing and various body circumference measurements from 252 men. We will try to make predictions of the percentage of body fat based on circumference measurements using linear regression models.

You can load the data set with the function `readbodyfat`. Since the data set seems to be ordered somewhat with respect to the age of the test subjects, we

should pick the members of the training and test sets at random. After loading the dataset the function performs a random split of the data set into a training and test set (in roughly 80% training and 20% test).

You find a detailed explanation of the variables in the dataset in the file `bodyfat.txt`, but the short description is that we would like to predict percentage body fat given in the 2nd column from a subset of the variables given in columns 4 to 15. We will not use the values found in column 1 (column 2 is actually a function of column 1) and column 3 (the age of the subject). We will think of column 2 as what is referred to in the book as the target variable t and a selection of columns 4 to 15 as the observations $\mathbf{x}$. We will in the following consider the following two selections of observation variables:

Selection 1: Let $\mathbf{x}$ consist of the data in columns 4, 7, 8, and 9, hence the dimensionality of this subset is $D = 4$.

Selection 2: Let $\mathbf{x}$ consist of the data in column 8, hence the dimensionality of this subset is $D = 1$.

Question 2.1

We will start by trying to model the data set with linear regression as defined in eq. (3.1) in the book and learn the parameters using the maximum likelihood approach. That is, we will use the linear model

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \ldots + w_D x_D \ .$$

Implement this model for variable selections 1 and 2 by constructing their corresponding design matrices as defined in eq. (3.16). Train each of these models on the training set by finding the maximum likelihood (ML) estimate by using eq. (3.15) (hint: In Matlab you can compute the pseudo inverse of the design matrix with the function `pinv` and in R by using `ginv` or `pseudoinverse`). Apply each model to the test set using eq. (3.3) with the ML parameter estimate and compute and report the root mean square (RMS) error

$$\mathrm{RMS} = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(t_n - y(\mathbf{x}_n, \mathbf{w}))^2} \ .$$

Which of the two models seem to provide the best prediction?

Question 2.2

Next let us try to learn the two models using Bayesian learning and maximum a posterior (MAP) estimation. Lets fix the prior distribution on the parameters to the zero mean isotropic Gaussian as given in eq. (3.52). Set the noise precision parameter to $\beta = 1$. We may then obtain the posterior distribution as given in eq. (3.49), by computing the estimates of the mean $\mathbf{m}_N$ and covariance $\mathbf{S}_N$ using eq. (3.53) and (3.54). The posterior mean is the MAP estimate. Using the MAP estimate and eq. (3.3) apply each model to the test set and compute and plot the root mean square (RMS) error for different values of the prior precision parameter α. Which of the two models seem to provide the best prediction? How does the results compare with the maximum likelihood results? For what value of the prior precision parameter α does the RMS error go below the RMS for the maximum likelihood solution (the solution from Question 2.1)?

2 Classification

The classification part of the case is constructed as follows: For visualization purposes we introduce an artificial example in two dimensions. The questions goes through all the classifiers we will investigate as applied to the artificial data set. In the last question we will apply a subset of the methods of you choice to a real data set of hand-written digits with the purpose of constructing digit classifiers. Code implementing the different methods are provided in the first question. In the last question you will have to code yourself building upon the code provided. Questions on decision theory for classification is also included.

We consider the following classifiers:

- Normal class conditional densities – Bayes classifier (generative)
- Logistic regression (discriminative)
- Naive Bayes classifier (generative)
- K-nearest neighbor (generative)
- Neural networks (discriminative)
- Support vector machines (discriminant functions)

The parenthesis indicates the classification of the methods in accordance with the taxonomy found in Bishop 1.5.4.

Optimal decision regions – artificial 2d data

We will start out by setting up an artificial classification example in two dimensions that we will use to illustrate the different classifiers used in this case. In the first question we introduce the Bayes classifier eq. (1.82) assuming that the conditional densities $P(\mathbf{x}|\mathcal{C}_k)$ are known for all classes $k = 1, \ldots, K$. This is of course an idealized situation. In general we will have to learn the classifier from a finite data set. The purpose of this question is to define the best possible classifier that we can use as a reference (ground truth) in the following.

The program `conddensity.m` returns class conditional densities for each class (use `help conddensity` in matlab) and the program `plotgrid.m` defines a grid of input points and can plot the different regions that you define when you build a classifier.

Note to R-users: All the provided programs also come in equivalent R-versions. The R and Matlab codes work in approximately the same way, however of course the randomly selected density distributions are not the same. But that is irrelevant to the solution of the exercise.

Question 2.3

Modify `plotgrid.m` to make a Bayes classifier for minimizing the misclassification rate. Try to vary the prior probabilities for the classes and see how that affect the optimal decision regions.

Class conditional densities - Bayes classifier

We will now try to learn from data. We will use maximum likelihood learning for Gaussian class conditional densities as in section 4.2.2. With the program `getdataconddensity.m` you can get training data from each of the conditional densities.

Question 2.4

Implement the maximum likelihood solution for Gaussian density for each class. Reusing code from question 2.1 is of course fine. For the first attempt use shared covariances eqs. (4.78)-(4.80) in the densities. You can modify the `plotgrid` program to plot the decision regions (assuming that we want to minimize misclassification rate). Compare the result to the optimal decision regions. What kind of decision boundaries do you get? Next, let each class conditional density have it's own covariance. What kind of decision boundaries do you get now? Are these closer to the optimal ones?

Discriminative – logistic regression

In this exercise, we will implement the Newton-Raphson algorithm for logistic regression as given in eq. (4.99). Remember that $\mathbf{\Phi}$ is the N by M design matrix. We will try different basis function sets (or feature spaces). The first is constant plus linear $\phi(\mathbf{x}) = (1, \mathbf{x}^T)^T$.

Question 2.5

Modify `plotgrid` again and plot the resulting decision boundary. What kind of decision boundaries do you get? Next, add a quadratic terms in the design matrix, that is let $\phi(\mathbf{x}) = (1, x_1, \ldots, x_d, x_1^2, \ldots, x_d^2)$, where d is the dimension of the input space and $M = 2d + 1$. Component-wise multiplication is `.*` in matlab while in R it is just the usual `*`. How does that affect your results?

Naive Bayes classifier

The naive Bayes model, eq. (1.84), assumes that the input features are independent given the class label:

$$p(\mathbf{x}|\mathcal{C}_k) = \prod_i p(\mathbf{x}_i|\mathcal{C}_k) \ .$$

With this assumption we can construct both class conditional and discriminative approaches from eq. (1.85).

Question 2.6

We consider again the artificial two dimensional data set and naturally assume the following conditional independence: $p(\mathbf{x}|\mathcal{C}_k) = p(x_1|\mathcal{C}_k)p(x_2|\mathcal{C}_k)$. Following eq. (1.85) the class conditional approach becomes

$$p(\mathcal{C}_k|\mathbf{x}) \propto p(x_1|\mathcal{C}_k)p(x_2|\mathcal{C}_k)p(\mathcal{C}_k)$$

and the discriminative

$$p(\mathcal{C}_k|\mathbf{x}) \propto \frac{p(\mathcal{C}_k|x_1)p(\mathcal{C}_k|x_2)}{p(\mathcal{C}_k)} \ .$$

Write down these equations with equality, that is find the normalizing constant. Describe in words how you would implement these approaches. Have we in fact used one of these models before (in what question)? Run code `case26` and comment on the result. Can you see the effect of the independence assumption visually?

K-nearest neighbor classifier

Question 2.7

Use `getdataconddensity.m` to generate a two dimensional artificial problem. Train the classifer using the k nearest neighbor rule, see `case27`. Plot the test error as a function of k, what is the best setting for k in terms of test error?. Try to change the number of observations in the training set. Is there a change in the trend of the test error? Why?

Decision theory

In question 2.1 we constructed the optimal decision boundary for minimizing the misclassification rate. In decision theory terms this corresponds to a symmetric loss. In the next two questions we explore aspects of decision theory for discrete and continuous data.

Question 2.8

The source of the following questions is

`http://courses.csail.mit.edu/6.825/fall04/exercises/dec_thy.pdf`

Dr. No has a patient who is very sick. Without further treatment, this patient will die in about 3 months. The only treatment alternative is a risky operation. The patient is expected to live about 1 year if he survives the operation; however, the probability that the patient will not survive the operation is 0.3.

1. Draw a decision tree for this simple decision problem. Show all the probabilities and outcome values. There is only one decision node (operation?), one chance node (to live or to die) and three outcome nodes (measured in months).

2. Let $U(x)$ denote the patient's utility function (the opposite of a loss function), where x is the number of months to live. Assuming that $U(12) = 1.0$ and $U(0) = 0$, how low can the patient's utility for living 3 months be and still have the operation be preferred? For the rest of the problem, assume that $U(3) = 0.8$. Show that the solution is that the operation would be preferred as long as $U(3) < 0.7$.

3. Dr. No finds out that there is a less risky test procedure that will provide uncertain information that predicts whether or not the patient will

survive the operation. When this test is positive, the probability that the patient will survive the operation is increased. The test has the following characteristics:

- True-positive rate: The probability that the results of this test will be positive if the patient will survive the operation is 0.90.
- False-positive rate: The probability that the results of this test will be positive if the patient will not survive the operation is 0.10.

What is the patient's probability of surviving the operation if the test is positive? Go through the calculation and explain each step ($S \equiv$ survive)

$$\begin{aligned} p(\text{survive}|\text{pos}) &= \frac{p(\text{pos}|\text{survive})\text{p(survive)}}{p(\text{pos})} \\ &= \frac{p(\text{pos}|\text{S})\text{p(S)}}{p(\text{pos}|\text{S})\text{p(S)} + \text{p(pos}|\text{not S)p(not S)}} \\ &= \frac{0.9 \times 0.7}{0.66} = 0.9545 \ . \end{aligned}$$

Why are the known quantities of the test defined in terms of the inverse probabilities and not directly in terms of p(survive|pos)?

4. Assuming the patient has the test done, at no cost, and the result is positive, should Dr. No perform the operation? $E[U(\text{op})] > 0.8$.

Question 2.9

We will now return to the two dimensional artificial data example and look at optimal decision regions when we have an asymmetric loss. Describe the meaning of the loss matrix for the cancer versus normal example (or equivalently the traffic example of the lectures)

	cancer	normal
cancer	0	1000
normal	1	0

and modify the `plotgrid.m` program again in order to find the optimal decision regions for this case. Take class 1 to be cancer.

Neural networks

We will use a two-layer feed-forward neural network on the two-dimensional classification problem used previously. The goal is to understand the influence of the number of hidden units and see how a weight decay regularizer will change the decission boundary.

Question 2.10

Use `case210a` to train a neural network on the two dimensional artificially generated data for minimizing the training/test misclassification rate. Try to vary the number of hidden units in the network and see both the training error and the resulting decision boundary. What is the trend? Should a simple

decission boundary or a complex one be preferred?. Now, consider the case of a fixed number of hidden units so the complexity of the network is controlled trough regularization. Run `case210b` with e.g. 8 hidden units and try to find the regularization coefficient values for which the training and test error are minimal.

Support vector machines

The code in `case29.m` implements a support vector machine for the non-separable case using the matlab function `quadprog.m` for solving the quadratic programming with slack variables described in section 7.1.1. mainly eqs. (7.23)-(7.28).

Question 2.11

Modify the code in `case211.m` to implement a support vector machine. Then try to alter the kernel function and the penalization parameter C and comment upon how that affects the decision boundary and the performance. Inspect visually whether we are close to Bayes classifier for some choices? You can quantify the performance by using a test set.

Note: If you are worried about how we in practice should choose kernel function and kernel parameters you can implement a cross validation scheme. But wait with this until you have completed the remaining questions.

Real data – hand-written digits

Finally we should test some of the classifiers on a relatively large real data set. The data set is found in the file `zipdata.mat` and is a subset of the so-called MNIST data set. The first column in the hand-written digit data contains the class label and the remaining 256 are the 16 by 16 image. Use `binaryfromzip` to load training and test data for two arbitrary digits (dig1 and dig2). If you want more data to work with google MNIST.

Question 2.12

Keep a log of your settings and results. Go through the following steps:

1. Split the data set in a training and test set. Choose the ratio of training to test cases. How should you balance the number of examples for each class?

2. Train a number (minimum two maximum four) of the classifiers we have discussed in this assignment and the lectures.

3. Report the performance on the training and test sets in terms of the misclassification rate and confusion matrices.

4. Compare the performance of the methods. Do you think that the differences are significant?

5. Some of the methods we have learned about have additional parameters that are not learned directly from the training data. Discuss briefly other means to learn these. One possibility is with cross-validation. Discuss briefly what value of the fold (for example 10 or N) is to be preferred.

Basically, the same protocol can be used for any other classification data set so now you should be ready to go on yourself. ;-)

Ricardo Henao, Kim Steenstrup Pedersen and Ole Winther, February 2010.