FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Machnine Learning (PDEEC0049 : 15-782PP)
Homework 2

António Damião das Neves Rodrigues (200400437 : 700098386)

November 12, 2013

1 PROBLEM 1

Let us consider the losses one gets for wrongly choosing $Y_i = 0$ and $Y_i = 1$, for now disregarding the reject option and its loss value λ. One can define a loss matrix $\mathbf{L}$ for such problem as follows:

$$\mathbf{L} = \begin{bmatrix} \ell_{00} & \ell_{01} \\ \ell_{10} & \ell_{11} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{1.1}$$

In 1.1, ℓ_{01} represents the cost for deciding for $Y_i = 1$ when the right choice would be $Y_i = 0$. The costs for deciding for $Y_i = 0$ and $Y_i = 1$ would then be given by the following expressions:

$$\begin{aligned} \mathscr{L}(Y_i = 0) &= \ell_{00} P(Y_i = 0|x_i) + \ell_{10} P(Y_i = 1|x_i) \\ &= 0 \times (1 - p_i) + 1 \times p_i \\ &= p_i \\ \mathscr{L}(Y_i = 1) &= \ell_{01} P(Y_i = 0|x_i) + \ell_{11} P(Y_i = 1|x_i) \\ &= 1 \times (1 - p_i) + 0 \times p_i \\ &= (1 - p_i) \end{aligned} \tag{1.2}$$

Taking a reject option into consideration, one would decide for it when such a choice would be advantageous relative to choosing $Y_i = 0$ and $Y_i = 1$. Since we are given a quantifiable cost for choosing the reject option, λ, one should choose it when $\mathscr{L}(Y_i = 0) > \lambda \wedge \mathscr{L}(Y_i = 1) > \lambda$, or $p_i > \lambda \wedge 1 - p_i > \lambda$. This reasoning results in the optimal decision rule given in 1.3.

$$\text{choose} \begin{cases} Y_i = 0, & \text{if } p_i < \frac{1}{2} \wedge p_i < \lambda \\ Y_i = 1, & \text{if } \frac{1}{2} < p_i \wedge 1 - \lambda < p_i \\ \text{reject}, & \text{if } \lambda < p_i < 1 - \lambda \end{cases} \tag{1.3}$$

2 Problem 2

2.1

Our objective is to determine the posterior probabilities $P(C_k|F_1, F_2, F_3, F_4)$, using Bayes' Theorem in the form shown in expression 2.1:

$$P(C_k|F_1, F_2, ..., F_n) = \frac{P(F_1, F_2, ..., F_n|C_k)P(C_k)}{P(F_1, F_2, ..., F_n)} \tag{2.1}$$

In this case, we cannot use the naive Bayes assumption. This assumption treats $F_1, F_2, ..., F_n$ as conditionally independent (given the class C_k) between each other, which would allow us to easily compute the factor $P(F_1, F_2, F_3, F_4|C_k)$ as $\prod_{i=1}^{4} P(F_i|C_k)$. In this case we would only have to model five parameters, each of the class-conditional probabilities $P(F_i|C_k)$ and a prior $P(C_1)$ (since $k = 2$, $P(C_2)$ would follow as $1 - P(C_1)$). In general, using the naive Bayes assumption, we have to model $k \times n + (k - 1)$ parameters. The problem would be solved as proposed in problem 2.2 (see Section 2.2).

Without the naive Bayes assumption, we would have to compute the factor $P(F_1, F_2, F_3, F_4|C_k)$ in a different way, since features $F_1, F_2, ..., F_n$ are no longer treated as conditionally independent between each other.

Using the product rule of probability, the expression $P(F_1, .., F_n|C_k)$ is equivalent to $\frac{P(C_k, F_1, .., F_n)}{C_k}$. The numerator is the joint probability distribution of $C_k, F_1, .., F_n$. This can be represented by the chain rule of probability, as given in expression 2.2, for $C_k, F_1, F_2, ..., F_n$:

$$P(C_k, F_1, F_2, ..., F_n) = P(C_k)P(F_1|C_k)P(F_2|F_1, C_k)...P(F_n|F_{n-1}, ..., C_k) \tag{2.2}$$

One should note that the dependence of each feature F_i on other features F_j now ask for the determination of much more parameters than for the naive Bayes case. E.g. for $P(F_2|F_1, C_k)$, we must execute a sum for each of the 5 values the feature F_1 can take, i.e.:

$$P(F_2|F_1, C_k) = \sum_{i=1}^{5} P(F_2|F_1 = q_i, C_k)$$

This particular example requires the calculation of 5 parameters $P(F_2|F_1 = q_i, C_k)$, for a given C_k. For conditional probability factors that depend on multiple features, one must account for the combinations of values q between features F_i and F_j. Generalizing, the total number of class-conditional parameters to determine for our case with 4 features which can assume

5 values is $5^3 + 5^2 + 5^1 + 1 = 5^4 - 1$. Considering the 2 classes C_1 and C_2, the number becomes $2(5^4 - 1)$.

In order to train the classifier, one would arrange the training data in a suitable table fashion, determine the relative frequencies for each one of these class-conditional parameters (plus the priors) and then determine the expressions for the posteriors $P(C_k|F_1, ..., F_n)$.

2.2

First, let us consider the following:

- A binary class Y, which can take values $y \in \{0, 1\}$. A value $y = 0$ means a phone didn't last one year, while $y = 1$ means a phone lasted one year.
- G a feature assuming values $g \in \{0, 1\}$. It is used to indicate if the user gender is female (0) or male (1).
- C a feature assuming values $c \in \{0, 1\}$. It is used to indicate if the phone's colors is black (0) or white (1).
- T a feature assuming values $t \in \{0, 1\}$. It is used to indicate if the phone's type is keypad-based (0) or touchscreen-based (1).

The problem asks us to compare $P(Y = 1|G = 1, C = 1, T = 1)$ and $P(Y = 0|G = 1, C = 1, T = 1)$. To compute these two values, we can use Bayes Theorem in the form given in 2.3 to calculate these as posterior probabilities.

$$P(Y = y|G = g, C = c, T = t) = \frac{P(G = g, C = c, T = t|Y = y)P(Y = y)}{P(G = g, C = c, T = t)} \tag{2.3}$$

For simplicity, we will assume that features G, C and T are conditionally independent (i.e. given $Y = y$) between each other, making $P(G = g, C = c, T = t|Y = y) = P(G = g|Y = y)P(C = c|Y = y)P(T = t|Y = y)$, i.e. we will be making the naive Bayes assumption.

Before applying Bayes theorem, we must compute the relevant values of the class conditional probabilities and the priors. These can be obtained by working out the relative frequencies among the data given in the exercise.

E.g. in order to find the value for $P(G = 1|Y = 1)$, count the number of 'Male' cases which 'lasted for more than a year', k, and divide this number by the total amount of cases which lasted for more than a year, ℓ, resulting in resulting in $P(G = 1|Y = 1) = \frac{k}{\ell} = \frac{3}{5}$.

The values in Tables 2.1 and 2.2 show the values of the class-conditional and prior probabilities for the problem in question, by application of the previously described process.

	Y = 1	Y = 0
$P(G=1\|Y)$	$\frac{3}{5}$	$\frac{2}{5}$
$P(C=1\|Y)$	$\frac{1}{5}$	$\frac{3}{5}$
$P(T=1\|Y)$	$\frac{2}{5}$	$\frac{3}{5}$

Table 2.1: Class-conditional probabilities for Problem 2.2.

	Y = 1	Y = 0
$P(Y)$	$\frac{5}{10}$	$\frac{5}{10}$

Table 2.2: Prior probabilities for Problem 2.2.

With the class-conditional and prior probability values, one can now apply Bayes' theorem to calculate the posteriors $P(Y=1|G=1,C=1,T=1)$ and $P(Y=0|G=1,C=1,T=1)$ and compare their values:

$$\begin{aligned} P(Y=1|M,W,T) &= \frac{P(G=1|Y=1)P(C=1|Y=1)P(T=1|Y=1)P(Y=1)}{P(G=1,C=1,T=1)} \\ &= \frac{\frac{3\times1\times2}{125}\times\frac{1}{2}}{P(G=1,C=1,T=1)} = \frac{\frac{6}{250}}{P(G=1,C=1,T=1)} \\ P(Y=0|M,W,T) &= \frac{P(G=1|Y=0)P(C=1|Y=0)P(T=1|Y=0)P(Y=0)}{P(G=1,C=1,T=1)} \\ &= \frac{\frac{2\times3\times3}{125}\times\frac{1}{2}}{P(G=1,C=1,T=1)} = \frac{\frac{18}{250}}{P(G=1,C=1,T=1)} \end{aligned} \tag{2.4}$$

Since the factor on the denominator is common to both posterior values, one must only evaluate the numerator to verify which one is larger. In this case, $P(Y=1|G=1,C=1,T=1) < P(Y=0|G=1,C=1,T=1)$. Therefore, a white colored phone with touchscreen in the hands of a male individual is less likely to last for more than one year than not.

3 Problem 3

The conventions and notation specified on slide 2 in [2] will be used throughout sections 3.1 to 3.3.

3.1

Listing 1: MATLAB function for calculating the coefficients of the linear regression on a given training dataset, composed by the arrays.

```
function [weights] = regression(train_data_X, train_data_Y)
%% number rows in train_data_X = number of examples
%% number columns in train_data_X = dimension
%% train_data_Y

weights = (train_data_X'*train_data_X)\(train_data_X'*train_data_Y);

return;
```

The code on Listing 1 was obtained by deducing the normal equation of the error function 3.1, in matrix notation, leading to the expression shown in 3.2.

$$\mathscr{L}(\mathbf{w}, S) = \frac{1}{2}\sum_{n=0}^{N}(y_n - \mathbf{w}^t\mathbf{x}_n)^2 \tag{3.1}$$

$$\begin{aligned} \mathbf{X}^t\mathbf{X}\mathbf{w} &= \mathbf{X}^t\mathbf{y} \\ \mathbf{w} &= \left(\mathbf{X}^t\mathbf{X}\right)^{-1}\mathbf{X}^t\mathbf{y} \end{aligned} \tag{3.2}$$

Figure 3.1 was obtained by setting the 4^{th} argument of the function `poly_regression()` to **9** (i.e. in order to obtain the polynomial regression of degree 9 on the dataset composed by `train_data_X` and `train_data_Y`), on the MATLAB file `testRegression.m`.

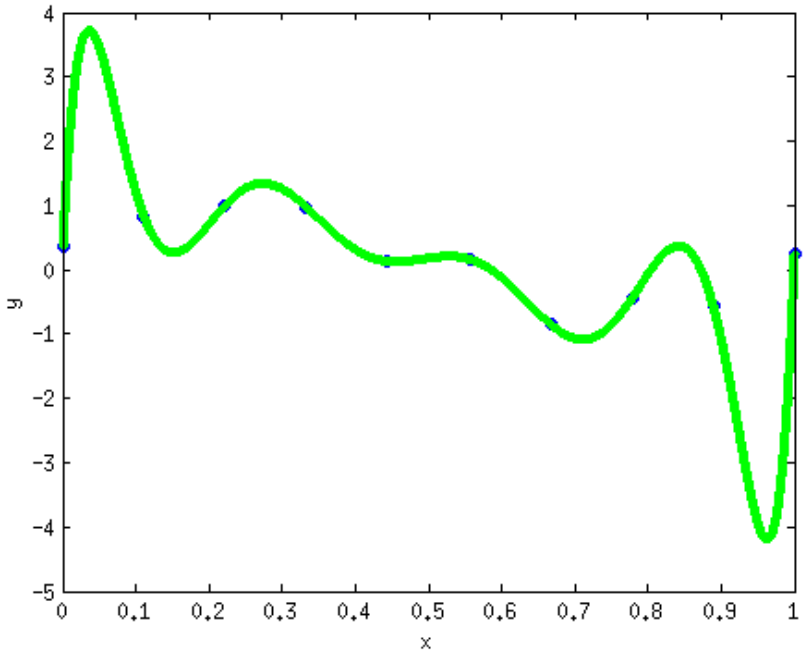

Figure 3.1: Plot of the polynomial function $\sum_{n=0}^{9} w_n x^n$, over values of x in the interval $[0,1]$ using the coeficients w_n produced by the code shown in Listing 1.

3.2

$$\mathscr{L}(\mathbf{w},S) = \frac{1}{2}\sum_{n=0}^{N} c_n(y_n - \mathbf{w}^t\mathbf{x}_n)^2 + \frac{1}{2}\lambda\mathbf{w}^t\mathbf{w} \tag{3.3}$$

In matrix notation, 3.3 becomes:

$$\begin{aligned}\mathscr{L}(\mathbf{w},S) &= \frac{1}{2}\left[(\mathbf{Xw}-\mathbf{y})^t\,\mathbf{C}\,(\mathbf{Xw}-\mathbf{y}) + \lambda\mathbf{w}^t\mathbf{w}\right] \\ &= \frac{1}{2}\left[(\mathbf{w}^t\mathbf{X}^t-\mathbf{y}^t)\,(\mathbf{CXw}-\mathbf{Cy}) + \lambda\mathbf{w}^t\mathbf{w}\right] \\ &= \frac{1}{2}\left[\mathbf{w}^t\mathbf{X}^t\mathbf{CXw} - \mathbf{w}^t\mathbf{X}^t\mathbf{Cy} - \mathbf{y}^t\mathbf{CXw} - \mathbf{y}^t\mathbf{Cy} + \lambda\mathbf{w}^t\mathbf{w}\right]\end{aligned} \tag{3.4}$$

$$\text{with}\quad \mathbf{C} = \begin{bmatrix} c_1 & 0 & 0 & \dots & 0 \\ 0 & c_2 & 0 & \dots & 0 \\ 0 & 0 & c_3 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & c_n \end{bmatrix}$$

To obtain the normal equations, take the derivative and set to zero, as in expression 3.5:

$$
\begin{aligned}
0 &= \frac{\partial \mathscr{L}}{\partial \mathbf{w}} \\
0 &= \frac{1}{2}\left[\frac{\partial}{\partial \mathbf{w}}\left(\mathbf{w}^t\mathbf{X}^t\mathbf{C}\mathbf{X}\mathbf{w}\right) - \frac{\partial}{\partial \mathbf{w}}\left(\mathbf{w}^t\mathbf{X}^t\mathbf{C}\mathbf{y}\right) - \frac{\partial}{\partial \mathbf{w}}\left(\mathbf{y}^t\mathbf{C}\mathbf{X}\mathbf{w}\right) - \frac{\partial}{\partial \mathbf{w}}\left(\mathbf{y}^t\mathbf{C}\mathbf{y}\right) + \lambda\frac{\partial}{\partial \mathbf{w}}\left(\mathbf{w}^t\mathbf{w}\right)\right] \\
0 &= \frac{1}{2}\left[\left(\left(\mathbf{X}^t\mathbf{C}\mathbf{X}\right)^t + \left(\mathbf{X}^t\mathbf{C}\mathbf{X}\right)\right)\mathbf{w} - \mathbf{X}^t\mathbf{C}\mathbf{y} - \left(\mathbf{y}^t\mathbf{C}\mathbf{X}\right)^t - 0 + 2\lambda\mathbf{w}\right] \\
0 &= \frac{1}{2}\left[2\left(\mathbf{X}^t\mathbf{C}\mathbf{X}\right)\mathbf{w} - 2\mathbf{X}^t\mathbf{C}\mathbf{y} + 2\lambda\mathbf{w}\right] \\
0 &= \left(\mathbf{X}^t\mathbf{C}\mathbf{X}\right)\mathbf{w} - \mathbf{X}^t\mathbf{C}\mathbf{y} + \lambda\mathbf{w} \\
\mathbf{X}^t\mathbf{C}\mathbf{y} &= \left(\mathbf{X}^t\mathbf{C}\mathbf{X} + \lambda\mathbf{I}\right)\mathbf{w}
\end{aligned}
\tag{3.5}
$$

3.3

Listing 2: MATLAB function for calculating the coefficients of the linear regression on a given training dataset, including a regularization factor.

```
1  function [weights] = regularizedregression(train_data_X, train_data_Y, c, ...
       lambda)
2  %% number rows in train_data_X = number of examples
3  %% number columns in train_data_X = dimension
4
5  % diagonal matrix C built as
6  % [c1 0 0 ... 0 ; 0 c2 0 ... 0 ; ... ; 0 0 0 ... cn]
7  C = diag(c);
8
9  % dimension for the identity matrix to be scaled by the lambda
10 % (regularization) factor
11 D = size(train_data_X,2);
12
13 weights = ((train_data_X'*C*train_data_X) + ...
       lambda*eye(D))\(train_data_X'*C*train_data_Y);
14
15 return;
```

The code on Listing 2 was obtained by implementing expression 3.5 in MATLAB, in order to $\mathbf{w}$. The vector `c`, passed to the function `regularizedregression()` as a $1 \times N$ vector, N being the number of lines in $\mathbf{X}$ (i.e. the number of samples in the dataset from PRMLBishop [1]), holds the coefficients c_n shown in expression 3.3. The expression in line 7 of Listing 2 diagonalizes the vector `c` into a $N \times N$ matrix $\mathbf{C}$, which holds the coefficients c_n as defined in expression 3.4.

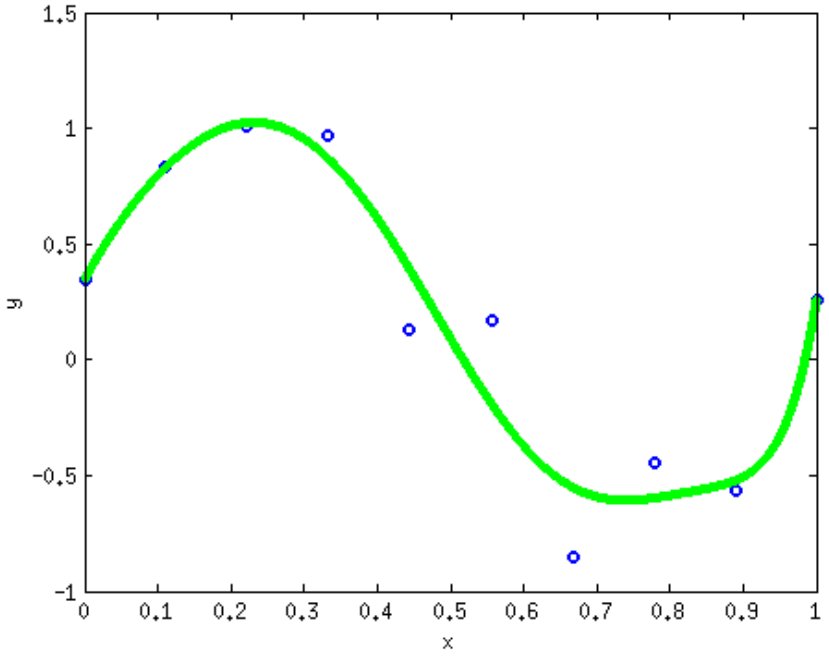

Figure 3.2: Plot of the polynomial function $\sum_{n=0}^{9} w_n x^n$, over values of x in the interval $[0,1]$ using the coefficients w_n produced by regularized regression expression 3.5 and the code shown in Listing 2. Regularization factor $\lambda = e^{-18}$.

Figure 3.2 was obtained by setting the 4$^{\text{th}}$ argument of the function `poly_regression()` to 9 (i.e. in order to obtain the polynomial regression of degree 9 on the dataset composed by `train_data_X` and `train_data_Y`), on the MATLAB file `testRegression.m`. Of course, the value of the `weights` variable in the MATLAB file `poly_regression.m` is set by the function `regularizedregression()`, with `lambda` $= e^{-18}$.

REFERENCES

[1] C.M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] Jaime S. Cardoso. PDEEC Machine Learning 2013/14 - Lecture 2: Regression Notes, 2013.