Applied Data Mining (UN3106)
Spring 2018
http://stat.columbia.edu/∼porbanz/UN3106S18.html

Peter Orbanz
porbanz@stat.columbia.edu
Phyllis Wan
pw2348@columbia.edu

Homework 1

Due: 22 February 2018

Homework submission: We will collect your homework **at the beginning of class** on the due date. If you cannot attend class that day, you can leave your solution in Phyllis Wan's postbox in the Department of Statistics, 10th floor SSW, at any time before then.

We do not accept homework submitted late. There will be no exceptions.

Problem 1 (Naive Bayes)

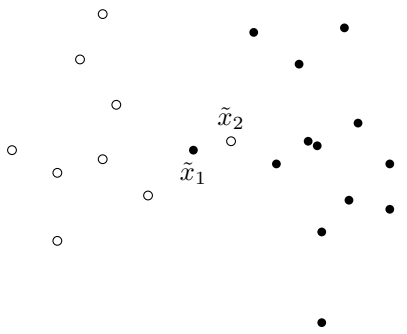
Consider a classification problem with training data $\{(\tilde{\mathbf{x}}_1, \tilde{y}_1), \ldots, (\tilde{\mathbf{x}}_n, \tilde{y}_n)\}$ and three classes $\mathcal{C}_1$, $\mathcal{C}_2$ and $\mathcal{C}_3$. The sample space is $\mathbb{R}^5$, so each data point is of the form $\mathbf{x} = (x^{(1)}, \ldots, x^{(5)})$. Suppose we have reason to believe that the distribution of each class is reasonably well-approximated by a spherical (unit-variance) Gaussian, i.e. the class-conditional distributions are $g(\mathbf{x}|\mu_k, \mathbb{I})$ for class $k \in \{1, 2, 3\}$.

Note: If a d-dimensional random vector $(X_1, \ldots, X_d)$ has a spherical Gaussian distribution, the scalar random variables $X_1, \ldots, X_d$ are stochastically independent from each other.

1. How is the Gaussian assumption translated into a naive Bayes classifier? Write out the full formula for the estimated class label $\hat{y}_{\text{new}} = f(\mathbf{x}_{\text{new}})$ for a newly observed data point $\mathbf{x}_{\text{new}}$.
   Hint: This equation should not contain the training data, only parameters estimated from the training data.
2. How do you estimate the parameters of the model? Give the estimation equations for (a) the parameters of the class-conditional distributions and (b) the class prior $P(y = k)$ for each class $\mathcal{C}_k$.
3. If our assumptions on the data source as described above are accurate, do you expect the naive Bayes classifier to perform well? Please explain your answer.

Problem 2 (Perceptron)

Consider the following training data, for a two class problem in which classes are labeled $+1$ (marked white in the figure) and -1 (marked black). There are $n = 22$ points in total. Clearly, this data is not linearly separable.



1. What is the minimal empirical risk a linear classifier can achieve on this data set, under 0-1 loss?

2. Consider the linear classifier in $\mathbb{R}^2$ given by $(\mathbf{v}_{\text{H}}, c)$, and two points $\mathbf{x}_1$ and $\mathbf{x}_2$. Suppose that

$$v_{\text{H}} := \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix} \qquad c := \frac{1}{2\sqrt{2}} \qquad \mathbf{x}_1 := \begin{pmatrix} -3 \\ 0 \end{pmatrix} \qquad \mathbf{x}_2 := \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} .$$

   Compute the classification result for $\mathbf{x}_1$ and $\mathbf{x}_2$.

3. Suppose you run the perceptron learning algorithm (say with a constant learning rate $\alpha(n) = 1$) on the data. How is the algorithm going to behave: Does it return a solution? If so, what can you say about the solution? If it does not find a solution, why not?

Problem 3 (Gradient descent)

In this problem, we use gradient descent to approximate the minima of the function,

$$f(x) = -\frac{1}{\sqrt{2\pi}} \exp\Big(-\frac{1}{2}x^2\Big) + \frac{1}{\sqrt{2\pi}} \exp\Big(-\frac{1}{2}\Big(\frac{x-1}{0.5}\Big)^2\Big) - \frac{1}{\sqrt{2\pi}} \exp\Big(-\frac{1}{2}\Big(\frac{x-2}{0.4}\Big)^2\Big).$$

1. In R, plot the function $f(x)$ for $x \in (-2, 4)$. Hint: f can be described as:

```
f <- function(x){
    - dnorm(x,0,1) + 0.5*dnorm(x,1,0.5) - 0.4*dnorm(x,2,0.4)
}
```

We will approximate the derivative $f'(x)$ as by replacing f by a linear function within a small window: We choose some small value $\delta > 0$, and compute

$$\hat{f}_\delta(x) \quad := \quad \frac{f(x+\delta) - f(x-\delta)}{|[x-\delta, x+\delta]|} \quad = \quad \frac{f(x+\delta) - f(x-\delta)}{2\delta} ,$$

and use $\hat{f}_\delta(x)$ as our estimate of $f'(x)$. The function `f.prime()` we ask you to write below implements $\hat{f}_\delta$.

2. Write a function in R, `f.prime(x)`, to calculate the numerical derivative given the location `x`. Take the approximation window to be $\delta = 0.001$. What is the output of `f.prime(-2)`?

3. Write a function in R, `grad.des(x1)`, to perform gradient descent from the starting point `x1`. Take the step sizes $\alpha_n = 1/n$, and precision $\epsilon = 0.05$, that is, compute

$$x_{n+1} := x_n - \frac{1}{n}\hat{f}_\delta(x_n) .$$

   Your function should output a list including the number of iterations N, the minima $x^* = x_N$, the minimum value $f_{\min} = f(x^*)$ and the vector of your search trajectory $(x_1, x_2, \ldots, x_N)$.

4. Start from $x_1 = -2$, what is the minimum your function finds? Plot $x_1, x_2, \ldots, x_{10}$ with the curve of f in Part 1.

As we have discussed in class, a gradient descent algorithm generally finds local minima, and these minima may not be global. One way to address this is to randomly start gradient descent at several different points, and then compare the function values at the resulting local minima.

5. Generate 100 random starting points between $(-2, 4)$, calculate the gradient descent result for each of them. Plot the histogram of the 100 minima you found. What is the global minimum?