# Problem set 3

- **Problem 1 (4 pts)**
  It was mentioned in the lecture that a cubic regression spline with two knots at $\xi_1 < \xi_2$ can be represented as

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x - \xi_1)_+^3 + \beta_5 (x - \xi_2)_+^3$$

  where

$$(x - \xi_k)_+^3 = \begin{cases} (x - \xi_k)^3 & \text{if } x > \xi_k \\ 0 & \text{otherwise} \end{cases}, \quad \forall\ k \in \{1, 2\}. \tag{0.1}$$

  We now verify $f(x)$ is indeed a cubic regression spline by proving:

  1. **(1 pt)** $f(x)$ can be written as three cubic polynomials for $x \leq \xi_1$, $\xi_1 < x \leq \xi_2$ and $x > \xi_2$.

  2. **(1 pt)** the three polynomials have the same value at each knot.

  3. **(1 pt)** the three polynomials have the same first order derivative at each knot.

  4. **(1 pt)** the three polynomials have the same second order derivative at each knot.

- **Problem 2 (7 pts)**
  Consider the classification problem with the label of $Y$ belong to $\mathcal{C} := \{1, 2, \ldots, K\}$ and any realization $x$ of $X \in \mathbb{R}^p$. The Bayes classifier $f^*$ at $X = x$ is defined as

$$f^*(x) := \underset{f(x)\in\mathcal{C}}{\operatorname{argmin}} \ \mathbb{E}\Big[1\{Y \neq f(X)\} \mid X = x\Big].$$

  1. (**2 pts**) Prove that
$$f^*(x) = \underset{k\in\mathcal{C}}{\operatorname{argmax}} \ \mathbb{P}(Y = k \mid X = x).$$

  2. (**1 pt**) Prove that the Bayes error at $X = x$ equals to
$$1 - \max_{k\in\mathcal{C}} \ \mathbb{P}(Y = k \mid X = x).$$

  3. (**2 pts**) Consider that $K = 3$. For a fixed $x_0$, assume that
$$\mathbb{P}(Y = 1 \mid X = x_0) = 0.5$$
$$\mathbb{P}(Y = 2 \mid X = x_0) = 0.3$$
$$\mathbb{P}(Y = 3 \mid X = x_0) = 0.2.$$
     State the Bayes classifier at $X = x_0$ and compute its error at $X = x_0$.

  4. (**2 pts**) Consider a naive classifier $\hat{f}$, called random guessing, which randomly picks one label from $\mathcal{C} = \{1, 2, 3\}$ with equal probability. Compute its expected error rate at $X = x_0$ and compare it with the Bayes error at $X = x_0$ in part 3.

- **Problem 3 (7 pts)**
  We will prove that a random guessing classifier for binary classification has the area under the curve (AUC) equal to 1/2. Suppose $Y \in \{0, 1\}$ with $Y = 1$ meaning *true*, and *false* otherwise. Consider the following random guessing classifier at any $X = x$

$$\hat{f}(x) = \begin{cases} 1, & \text{with prob. equal to } p \\ 0, & \text{with prob. equal to } 1 - p \end{cases}.$$

  1. (**3 pts**) Prove that the expected AUC of $\hat{f}$ is 1/2. [Hint: the expected False Positive Rate is $\mathbb{P}(\hat{f}(X) = 1 \mid Y = 0)$]

  2. (**2 pts**) Let $\eta(x) := \mathbb{P}(Y = 1 \mid X = x)$ for any $x$. Write the expected error rate of $\hat{f}$ at $X = x$ in terms of $\eta(x)$ and $p$.

  3. (**2 pts**) If you have the flexibility of choosing $p$ in your expression of part 2, which choice minimizes the expected error rate of $\hat{f}$ at $X = x$? Is the resulting classifier equivalent to the Bayes classifier? State your explanation.

- **Problem 4 (14 pts)**
  This question uses the variables `dis` (the weighted mean of distances to five Boston employment centers) and `nox` (nitrogen oxides concentration in parts per 10 million) from the `Boston` data in the library `MASS`. We will treat `dis` as the predictor and `nox` as the response. For drawing the fitted lines below, let's use the following code to generate a grid of `dis`

```
dislims<-range(Boston$dis)
dis.grid<-seq(from=dislims[1],to=dislims[2],length.out=100)
```

  1. **(2 pt)** Use the `poly()` function to fit a polynomial regression with degree equal to 10 to predict `nox` using `dis`. Plot the data, the polynomial fit and its 95% confidence band. Comment on the width of the confidence band.

  2. **(2 pts)** Plot the polynomial fits for a range of different polynomial degrees, from {1, 3, 5, 7, 10}, and report the associated residual sum of squares (RSS). What's your finding and how to explain it?

  3. **(2 pts)** Perform 10-fold cross-validation to select the best degree of the polynomial from {1, 3, 5, 7, 10}.

  4. **(2 pts)** Use the `bs()` function to fit a regression spline with 7 degrees of freedom to predict `nox` using `dis`. Specify how the knots are chosen and plot the resulting fit and its 95% confidence band. Comment on the width of confidence band.

  5. **(2 pts)** Use the `ns()` function to fit a natural regression spline with the same knots specified in the previous part. (You may find the `nox` function and `Boundary.knots` useful). Plot the resulting fit and its 95% confidence band. Comparing to the previous plot, comment on the difference.

  6. **(2 pts)** Now fit natural cubic splines for a range of degrees of freedom, from {5, 10, 15, 20} (specified via `df` in `ns()` function), and plot the resulting fits and report the resulting RSS. Comment on what you observe.

  7. **(2 pts)** Perform 10-fold cross-validation to select the best degrees of freedom for a natural regression spline from {5, 10, 15, 20}.

- **Problem 5 (18 pts)**
  You will implement different local regression approaches in this problem. Let's set the seed to `set.seed(20231101)`. For $n = 300$ training data, let's generate $(x_i, y_i)$ for $1 \le i \le n$ from

$$x_i \overset{i.i.d.}{\sim} N(3,1), \qquad \epsilon_i \overset{i.i.d.}{\sim} N(0,1),$$
$$y_i = 0.5 + 0.1x_i + 0.2x_i^2 + \epsilon_i.$$

  Let's generate another $m = 300$ test data points by the same generating mechanism. Set a grid of the training data as `x_grid=seq(0,6,by=0.02)`

  1. (**5 pts**) Implement the $k$-nearest-neighbor, that is, for any given $x_0$, use the averaged responses of its $k$ nearest neighbors to predict, namely,

$$\hat{y}_1(x_0) = \frac{1}{k} \sum_{i \in N_k(x_0)} y_i$$

     with $N_k(x_0)$ denoting the index set of $k$ nearest neighbors of $x_0$ among $\{x_1, \ldots, x_n\}$. Use your function to predict at each value of `x_grid`. Plot the training data points and add on the lines of your fitted values for $k \in \{5, 20, 50, 100\}$. Make sure you label each line correctly in your plot. Comment on your findings for different choices of $k$.

  2. (**5 pts**) Repeat step 1 for the weighted $k$-nn, that is, for any given $x_0$, predict by

$$\hat{y}_2(x_0) = \sum_{i \in N_k(x_0)} w(x_0, x_i) y_i.$$

     Here we choose the so-called *tricubic weights* as

$$w(x_0, x_i) = \frac{\widetilde{w}(x_0, x_i)}{\sum_{i \in N_k(x_0)} \widetilde{w}(x_0, x_i)}, \quad \forall\ i \in N_k(x_0) \tag{0.2}$$

     with

$$\widetilde{w}(x_0, x_i) = \left(1 - \left(\frac{|x_i - x_0|}{\max_{i \in N_k(x_0)} |x_i - x_0|}\right)^3\right)^3, \tag{0.3}$$

  3. (**5 pts**) Repeat step 1 for the weighted local linear regression, that is, for any given $x_0$,

$$\hat{y}_3(x_0) = \hat{\beta}_0 + \hat{\beta}_1 x_0$$

     with

$$(\hat{\beta}_0, \hat{\beta}_1) = \underset{\beta_0, \beta_1}{\operatorname{argmin}} \sum_{i \in N_k(x_0)} w(x_0, x_i)(y_i - \beta_0 - \beta_1 x_i)^2.$$

     Here $w(x_0, x_i)$ is the same as in (0.2) – (0.3). For implementing the above weighted least squares, you might use the `lm` function by specifying the `weights` argument.

  4. (**3 pts**) Use the test data to choose the best one among the above three procedures for $k \in \{5, 20, 50, 100\}$.