Homework 4 (Nov. 16th)

Deadline: Wednesday, November 30th, at 11:59pm.

Submission: Read the submission instruction carefully! There are 5 questions in this assignment. You need to submit two files through Quercus for this assignment.

- The first file should be a PDF file titled `hw4_writeup.pdf` containing your answers to Questions 1 – 5, as well as R code and R outputs requested for Questions 4 and 5. You can produce the file however you like (e.g. LaTeX, Microsoft Word, scanner), as long as it is readable.
- The second file should be your completed R code, named as `discriminant_analysis.R`. You need to ensure that this file has the exact name as indicated. DO NOT set or modify the working directory within this file.

Neatness Point: You will be deducted one point if we have a hard time reading your solutions or understanding the structure of your code.

Late Submission: 10% of the total possible marks will be deducted for each day late, up to a maximum of 3 days. After that, no submissions will be accepted.

- **Problem 1 (6 pts)**
  In this question, you will derive the maximum likelihood estimates for Gaussian Naïve Bayes in which a random discrete class label $Y \in [K] := \{1, 2, \ldots, K\}$ and a random feature $X \in \mathbb{R}^D$ satisfy

$$\mathbb{P}(Y = k) = \pi_k, \qquad X \mid Y = k \sim N_p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \qquad \forall k \in [K]. \tag{0.1}$$

  Here $\pi_1, \ldots, \pi_K$ are the priors of the class label Y, and conditioning on $Y = k$ for any $k \in [K]$, the feature vector $X \in \mathbb{R}^p$ has a p-dimensional Gaussian density with mean $\boldsymbol{\mu}_k \in \mathbb{R}^p$ and a diagonal covariance matrix

$$\boldsymbol{\mu}_k = \begin{bmatrix} \mu_{k1} \\ \vdots \\ \mu_{kp} \end{bmatrix}, \qquad \boldsymbol{\Sigma}_k = \begin{bmatrix} \sigma_{k1}^2 & 0 & \cdots & 0 \\ 0 & \sigma_{k2}^2 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \sigma_{kp}^2 \end{bmatrix}.$$

  Let $(y_1, \mathbf{x}_1), \ldots, (y_n, \mathbf{x}_n)$ be n i.i.d. realizations of (Y, X).

  1. **(3 pts)** Write down the log-likelihood function of $(y_1, \mathbf{x}_1), \ldots, (y_n, \mathbf{x}_n)$.
     Hint: Let Z be a categorical variable taking values from $\{1, \ldots, K\}$ with corresponding probabilities $\theta_1, \ldots, \theta_K$ with $\theta_k \geq 0$ and $\sum_k \theta_k = 1$. Its probability mass function at any $Z = z$ is

$$\mathbb{P}(Z = z) = \prod_{k=1}^{K} \theta_k^{1\{z=k\}}.$$

  2. **(3 pts)** Derive the maximum likelihood estimators of π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ for all $k \in [K]$. You may assume $\sum_{i=1}^n 1\{y_i = k\} > 0$ for all $k \in [K]$.

- **Problem 2 (4 pts)**
  It was mentioned in the lecture that a cubic regression spline with one knot at ξ can be obtained as

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

  where

$$(x - \xi)_+^3 = \begin{cases} (x - \xi)^3 & \text{if } x > \xi \\ 0 & \text{otherwise} \end{cases}.$$

  We now verify parts (1) – (4) below to conclude that $f(x)$ is indeed a cubic regression spline.

  1. **(1 pt)** Verify that $f(x)$ is a piecewise cubic polynomials. That is, show that $f(x)$ can be written as two cubic polynomials for $x > \xi$ and $x \leq \xi$.
  2. **(1 pt)** Denote the two polynomials as $f_1(x)$ and $f_2(x)$. Show that $f_1(\xi) = f_2(\xi)$, that is, $f(x)$ is continuous at ξ.
  3. **(1 pt)** Show that $f_1'(\xi) = f_2'(\xi)$, that is, the first order derivative $f'(x)$ is continuous at ξ.
  4. **(1 pt)** Show that $f_1''(\xi) = f_2''(\xi)$, that is, the second order derivative $f''(x)$ is continuous at ξ.

- **Problem 3 (4 pts)**
  A 1-dimensional binary classification training set $\{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}$ and $y_i \in \{0, 1\}$ is *linear separable* if there exists a threshold $t \in \mathbb{R}$ such that

$$\begin{aligned} x_i &< t, \quad \text{for all } y_i = 0 \\ x_i &\geq t, \quad \text{for all } y_i = 1. \end{aligned}$$

  1. **(2 pts)** Suppose we have the following 1-D dataset for binary classification:

x_i	y_i
-1	1
1	0
3	1

     Argue briefly (at most a few sentences) that this dataset is not linearly separable.

  2. **(2 pts)** Now suppose we map the 1-dimensional feature into a 2-dimensional space

$$h(x) = \begin{bmatrix} h_1(x) \\ h_2(x) \end{bmatrix} = \begin{bmatrix} x \\ x^2 \end{bmatrix}.$$

     Is the new data set $(h(x_1), y_1), (h(x_2), y_2), (h(x_3), y_3)$ linear separable? That is, does there exist pairs of (t_1, t_2) such that

$$\begin{aligned} t_1 h_1(x_i) + t_2 h_2(x_i) &< 1, \quad \text{for all } y_i = 0 \\ t_1 h_1(x_i) + t_2 h_2(x_i) &\geq 1, \quad \text{for all } y_i = 1. \end{aligned}$$

- **Problem 4 (12 pts)**
  For this question you will build classifiers to label images of handwritten digits. Each image is 8 by 8 pixels and is represented as a vector of dimension 64 by listing all the pixel values in raster scan order. The images are grayscale and the pixel values are between 0 and 1. The labels y are $\{0, 1, 2, \ldots, 9\}$ corresponding to which character was written in the image. There are 700 training points and 400 test points for each digit; they can be found in `digits_train.txt` and `digits_test.txt`. These data sets can be loaded by using the helper function in `utils.R`.

  You will implement both linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA) to classify these images. Recall that conditioning on each class $k \in \{0, 1, \ldots, 9\}$, the feature $X \mid Y = k$ follows a multivariate Gaussian distribution, that is,

$$\mathbb{P}(X = \mathbf{x} \mid Y = k) = (2\pi)^{-p/2} |\mathbf{\Sigma}_k|^{-1/2} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^\top \mathbf{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right\} \tag{0.2}$$

  where $\boldsymbol{\mu}_k \in \mathbb{R}^p$ is the conditional mean and $\mathbf{\Sigma}_k \in \mathbb{R}^{p \times p}$ is the conditional covariance matrix. For LDA, $\mathbf{\Sigma}_k$ is assumed to be the same across classes. The priors are

$$\pi_k = \mathbb{P}(Y = k), \qquad \text{for all } \; k \in \{0, 1, \ldots, 9\}.$$

  You will compute the maximum likelihood estimators of the priors π_k, the conditional means $\boldsymbol{\mu}_k$ and the conditional covariance matrices $\mathbf{\Sigma}_k$ for $k \in \{0, 1, \ldots, 9\}$, and use the estimators to construct classifiers.

  Read carefully the structure of `discriminant_analysis.R`. Include your code for all subquestions.

  1. **(4 pts)** Complete the functions `Comp_priors`, `Comp_cond_means` and `Comp_cond_covs` in the file `discriminant_analysis.R`.
  2. **(2 pts)** Complete the functions `Predict_posterior` and `Predict_labels` in the file `discriminant_analysis.R`.
  3. **(2 pts)** Use LDA to classify the test data by completing part a in `hw4_starter.R`. Report the misclassification error of LDA.
  4. **(2 pts)** Use QDA to classify the test data by completing part b in `hw4_starter.R`. Report the misclassification error of QDA.
  5. **(2 pts)** Complete part c in `hw4_starter.R`, i.e. perform LDA and QDA by using the built-in `lda` and `qda` functions and compare with your implementation in terms of both misclassification rates and computational speed.

- **Problem 5 (11 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. We will treat `dis` as the predictor and `nox` as the response. Include your code for each subquestion.

  1. **(1 pt)** Use the `poly()` function to fit a cubic polynomial regression to predict `nox` using `dis`. Report the fitted regression summary, and plot the resulting data and polynomial fits.
  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.
  3. **(2 pts)** Perform 10-fold cross-validation to select the optimal degree for the polynomial, and explain your results.
  4. **(2 pts)** Use the `bs()` function to fit a regression spline to predict `nox` using `dis`. Report the output for the fit using four degrees of freedom. Specify how you choose the knots and plot the resulting fit.
  5. **(2 pts)** Now fit a regression spline for a range of degrees of freedom, from {4, 6, 8, 10}, and plot the resulting fits and report the resulting RSS. Describe the results obtained.
  6. **(2 pts)** Perform 10-fold cross-validation to select the best degrees of freedom for a regression spline on this data. Describe your results.

- **Problem 1 (6 pts)**
  In this question, you will derive the maximum likelihood estimates for Gaussian Naïve Bayes in which a random discrete class label $Y \in [K] := \{1, 2, \ldots, K\}$ and a random feature $X \in \mathbb{R}^D$ satisfy

$$\mathbb{P}(Y = k) = \pi_k, \qquad X \mid Y = k \sim N_p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \qquad \forall k \in [K]. \tag{0.1}$$

  Here $\pi_1, \ldots, \pi_K$ are the priors of the class label Y, and conditioning on $Y = k$ for any $k \in [K]$, the feature vector $X \in \mathbb{R}^p$ has a p-dimensional Gaussian density with mean $\boldsymbol{\mu}_k \in \mathbb{R}^p$ and a diagonal covariance matrix

$$\boldsymbol{\mu}_k = \begin{bmatrix} \mu_{k1} \\ \vdots \\ \mu_{kp} \end{bmatrix}, \qquad \boldsymbol{\Sigma}_k = \begin{bmatrix} \sigma_{k1}^2 & 0 & \cdots & 0 \\ 0 & \sigma_{k2}^2 & \cdots & 0 \\ & & \ddots & \\ 0 & 0 & \cdots & \sigma_{kp}^2 \end{bmatrix}.$$

  Let $(y_1, \mathbf{x}_1), \ldots, (y_n, \mathbf{x}_n)$ be n i.i.d. realizations of (Y, X).

  1. **(3 pts)** Write down the log-likelihood function of $(y_1, \mathbf{x}_1), \ldots, (y_n, \mathbf{x}_n)$.
     Hint: Let Z be a categorical variable taking values from $\{1, \ldots, K\}$ with corresponding probabilities $\theta_1, \ldots, \theta_K$ with $\theta_k \geq 0$ and $\sum_k \theta_k = 1$. Its probability mass function at any $Z = z$ is

$$\mathbb{P}(Z = z) = \prod_{k=1}^{K} \theta_k^{1\{z=k\}}.$$

  2. **(3 pts)** Derive the maximum likelihood estimators of π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ for all $k \in [K]$. You may assume $\sum_{i=1}^{n} 1\{y_i = k\} > 0$ for all $k \in [K]$.

1. Since we know that $f(x) = \frac{1}{\sqrt{(2\pi)^D \det \Sigma}} \times \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$

and the we want to estimate for Gaussian Naive Bayes, and since we know that from Bayes theorem that $P(A|B) = \frac{P(B|A)P(A)}{P(B)}$ (post, condi, prior, measure)

and in this case, $P(Y=k \mid X=x) = \frac{P(X|Y=k)P(Y=k)}{P(X)}$

$= \frac{P(X|Y=k) \times \pi_k}{P(X)}$ → constant, therefore, we also know that Z is a categorical variable from

$\{1, \ldots, k\}$, and $p = \frac{\frac{1}{\sqrt{(2\pi)^n \det \Sigma}} \times \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right) \times \theta_k}{p(x)}$,

and since we know that $\sum_k \theta_k = 1$, then,

$\sum_k \alpha_k = 1$, and $p = \frac{\frac{1}{\sqrt{(2\pi)^n \det \Sigma}} \times \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)}{p(x)}$,

In order to maximize the $P(Y=k \mid X=x)$, we took $\log(P(Y=k \mid X=x))$, and since $P(X)$ is constant, and then maximize $P(X \mid Y=k) P(Y=k)$, and if we took $\log(P(X \mid Y=k) P(Y=k))$

$= \log(|P(X \mid Y=k)|) + \log P(Y=k)$

$$= \arg\max \left(\log \left(\sum_{i=1}^{n} f(x)^{1\{z=k\}} \right) + \log \sum_{i=1}^{n} \pi_k^{1\{z=k\}} \right)$$

$$= \arg\max \sum_{i=1}^{n} 1\{z=k\}$$

$$\sum_{j=1}^{n} \left[\log \left(\frac{1}{\sqrt{(2\pi)^D \times \det(\Sigma)}} \right)^{1\{z=k\}} + \sum_{j=1}^{n} \left(-\frac{1}{2} \right) \times (x-\mu)^T \Sigma^{-1} (x-\mu)^{1\{z=k\}} + \sum_{j=1}^{n} \pi_k^{1\{z=k\}} \right]$$

$=$

2. **(3 pts)** Derive the maximum likelihood estimators of π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ for all $k \in [K]$. You may assume $\sum_{i=1}^{n} 1\{y_i = k\} > 0$ for all $k \in [K]$.

2). If we take derive under μ_k, then we treat π_k and Σ_k as constant. and since $\log L(\theta)$ is

$$\sum_{i=1}^{n} \log \left(\frac{1}{\sqrt{(2\pi)^D \times \det(\Sigma)}} \times \exp\left(-\frac{1}{2} (x-\mu)^T \Sigma^{-1} (x-\mu) \right)^{1\{z=k\}} \times \pi_k^{1\{z=k\}} \right)$$

$$= \sum_{j=1}^{n} \left[\log \left(\frac{1}{\sqrt{(2\pi)^D \times \det(\Sigma)}} \right)^{1\{z=k\}} + \sum_{j=1}^{n} \left(-\frac{1}{2} \right) \times (x-\mu)^T \Sigma^{-1} (x-\mu)^{1\{z=k\}} \right.$$

$$\left. + \sum_{j=1}^{n} \pi_k^{1\{z=k\}} \right]$$

then take derivative μ_k,

we consider π_k and Σ equals to constant.

then we have

$$\frac{\partial \log L(\theta)}{\partial \mu_k} = \frac{\partial \sum_{i=1}^{n} \left(\left[-\frac{1}{2} \right] \times (x-\mu)^T \Sigma^{-1} (x-\mu) \right)}{\partial \mu_k}$$

$$= \frac{d}{d\mu} \left[I\{z=k\} \times \left[-\frac{1}{2} \times \Sigma^{-1} \right] \times (x-\mu)^T \times (x-\mu) \right]$$

$$= I\{ I\{z=k\} \times -\frac{1}{2} \times \Sigma^{-1} \times [-2 (x_i - \mu)]$$

$$= I\{y=k\} \Sigma^{-1} (x_i - \mu) = 0$$

$$= I\{y=k\} (x_i - \mu) = 0 \quad (\Sigma^{-1} \text{ is constant})$$

then $I\{y_i=k\} x_i - I\{y_i=k\} \mu = 0$

$$\Rightarrow I\{y_i=k\} x_i = I\{y_i=k\} \mu$$

$$\Rightarrow \mu = \frac{I\{y_i=k\} x_i}{I\{y_i=k\}}$$

② let $Z = \sqrt{(2\pi)^D \det(\Sigma_k)}$

$$\frac{\partial \log(\theta)}{\partial \Sigma_k^{-1}} = \frac{\partial \sum_{i=1}^{N} \left[(-\frac{1}{2}) \times (x-\mu)^T \Sigma^{-1} (x-\mu) \right]}{\partial \Sigma_k^{-1}}$$

$$= \frac{\partial \log(\theta)}{\partial \Sigma_k^{-1}}, \quad \text{since} \quad \frac{\partial \det(A)}{\partial A} = \det(A) A^{-1T},$$

$\det(A)^{-1} = \det(A^{-1})$, and $\frac{\partial x^T A x}{\partial A} = x x^T$,

$\Sigma^T = \Sigma$, then we know that

$$\frac{\partial \log L}{\partial \Sigma_k^{-1}} = -\sum_{i=0}^{N} 1(z^i = k) \left[-\frac{\partial \log Z}{\partial \Sigma_k^{-1}} - \frac{1}{2} (x^i - \mu_k)(x^i - \mu_k)^T \right]$$

$$= 0$$

and after calculation, then we have

$$\frac{\partial \log L(\theta)}{\partial \Sigma_k^{-1}} = -\sum_{i=0}^{N} I(z^i = k)\left[\frac{1}{2}\Sigma_k - \frac{1}{2}(x^i - \mu_k)(x^i - \mu_k)^T\right] = 0$$

Then, we have

$$\Sigma_k = \frac{\sum_{i=1}^{N} I(t^{(i)} = k)(x^{(i)} - \mu_k)(x^{(i)} - \mu_k)^T}{\sum_{i=1}^{N} I(t^{(i)} = k)}$$

- **Problem 2 (4 pts)**
  It was mentioned in the lecture that a cubic regression spline with one knot at ξ can be obtained as

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

  where

$$(x-\xi)_+^3 = \begin{cases} (x-\xi)^3 & \text{if } x > \xi \\ 0 & \text{otherwise} \end{cases}.$$

  We now verify parts (1) – (4) below to conclude that $f(x)$ is indeed a cubic regression spline.

  1. **(1 pt)** Verify that $f(x)$ is a piecewise cubic polynomials. That is, show that $f(x)$ can be written as two cubic polynomials for $x > \xi$ and $x \leq \xi$.
  2. **(1 pt)** Denote the two polynomials as $f_1(x)$ and $f_2(x)$. Show that $f_1(\xi) = f_2(\xi)$, that is, $f(x)$ is continuous at ξ.
  3. **(1 pt)** Show that $f_1'(\xi) = f_2'(\xi)$, that is, the first order derivative $f'(x)$ is continuous at ξ.
  4. **(1 pt)** Show that $f_1''(\xi) = f_2''(\xi)$, that is, the second order derivative $f''(x)$ is continuous at ξ.

1. Since we know that $f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x-\xi)_+^3$

and $(x-\xi)_+^3 = \begin{cases} (x-\xi)^3, & \text{if } x > \xi \\ 0, & \text{other,} \end{cases}$ then if we bring

$(x-\xi)_+^3$ into $f(x)$, we know that $(x-\xi)^3 = x^3 - 3x^2\xi + 3x\xi^2 - \xi^3$, $f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x^3 - 3x^2\xi + 3x\xi^2 - \xi^3)$

$= \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^3 - 3\beta_4 x^2\xi + 3\beta_4 x\xi^2 - \beta_4 \xi^3$

$= \beta_0 + (\beta_1 + 3\beta_4\xi^2)x + (\beta_2 - 3\beta_4\xi)x^2 + (\beta_3 + \beta_4)x^3 - \beta_4\xi^3$

$= (\beta_0 - \beta_4\xi^3) + (\beta_1 + 3\beta_4\xi^2)x + (\beta_2 - 3\beta_4\xi)x^2 + (\beta_3 + \beta_4)x^3$

Therefore, $f(x)=\begin{cases}(\beta_0-\beta_4\xi^3)+(\beta_1+3\beta_4\xi^2)x+(\beta_2-3\beta_4\xi)x^2+(\beta_3+\beta_4)x^3, & x>\xi \\ \beta_0+\beta_1x+\beta_2x^2+\beta_3x^3, & \text{other}\end{cases}$,

2. Since

$f(x)=\begin{cases}(\beta_0-\beta_4\xi^3)+(\beta_1+3\beta_4\xi^2)x+(\beta_2-3\beta_4\xi)x^2+(\beta_3+\beta_4)x^3, & x>\xi \\ \beta_0+\beta_1x+\beta_2x^2+\beta_3x^3, & \text{other}\end{cases}$, and

we let $(\beta_0-\beta_4\xi^3)+(\beta_1+3\beta_4\xi^2)x+(\beta_2-3\beta_4\xi)x^2+(\beta_3+\beta_4)x^3=f_1(x)$,

and $\beta_0+\beta_1x+\beta_2x^2+\beta_3x^3=f_2(x)$, then bring ξ we will

have $f_1(\xi)=\beta_0-\beta_4\xi^3+\beta_1\xi+3\beta_4\xi^3+\beta_2\xi^2-3\beta_4\xi^3+\beta_3\xi^3+\beta_4\xi^3=\beta_0+\beta_1\xi_1+\beta_2\xi^2+\beta_3\xi^3$,

$f_2(\xi)=\beta_0+\beta_1\xi_1+\beta_2\xi^2+\beta_3\xi^3=f_1(\xi)$ ☑

3. $f_1'(x)=\beta_1+3\beta_4\xi^2+2\beta_2x-6\beta_4\xi x+3\beta_3x^2+3\beta_4x^2$

$=\beta_1+2\beta_2x+3\beta_3x^2+3\beta_4(x-\xi)^2$, and

$f_2'(x)=\beta_1+2\beta_2x+3\beta_3x^2$, if we bring ξ then,

$$f_1'(\xi) = \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2 = f_2'(\xi) = \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2$$

Q. $f_1''(x) = 2\beta_2 + 6\beta_3 x + 6\beta_4(x-\xi)$, then $f_1''(\xi)$ is equals to $2\beta_2 + 6\beta_3\xi$, and $f_2''(x) = 2\beta_2 + 6\beta_3 x$, then $f_2''(\xi) = 2\beta_2 + 6\beta_3\xi = f_1''(\xi)$

- **Problem 3 (4 pts)**
  A 1-dimensional binary classification training set $\{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}$ and $y_i \in \{0, 1\}$ is *linear separable* if there exists a threshold $t \in \mathbb{R}$ such that

$$\begin{aligned} x_i &< t, \quad \text{for all } y_i = 0 \\ x_i &\geq t, \quad \text{for all } y_i = 1. \end{aligned}$$

1. **(2 pts)** Suppose we have the following 1-D dataset for binary classification:

x_i	y_i
-1	1
1	0
3	1

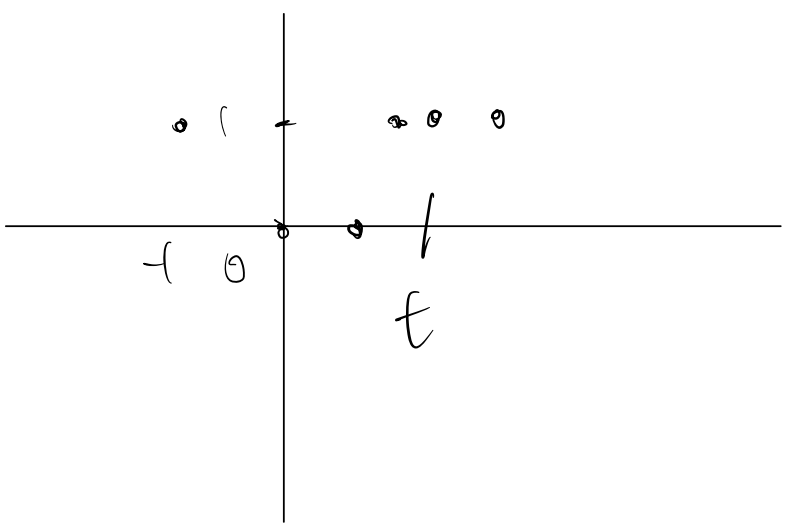
   Argue briefly (at most a few sentences) that this dataset is not linearly separable.

2. **(2 pts)** Now suppose we map the 1-dimensional feature into a 2-dimensional space

$$h(x) = \begin{bmatrix} h_1(x) \\ h_2(x) \end{bmatrix} = \begin{bmatrix} x \\ x^2 \end{bmatrix}.$$

   Is the new data set $(h(x_1), y_1), (h(x_2), y_2), (h(x_3), y_3)$ linear separable? That is, does there exist pairs of (t_1, t_2) such that

$$\begin{aligned} t_1 h_1(x_i) + t_2 h_2(x_i) &< 1, \quad \text{for all } y_i = 0 \\ t_1 h_1(x_i) + t_2 h_2(x_i) &\geq 1, \quad \text{for all } y_i = 1. \end{aligned}$$



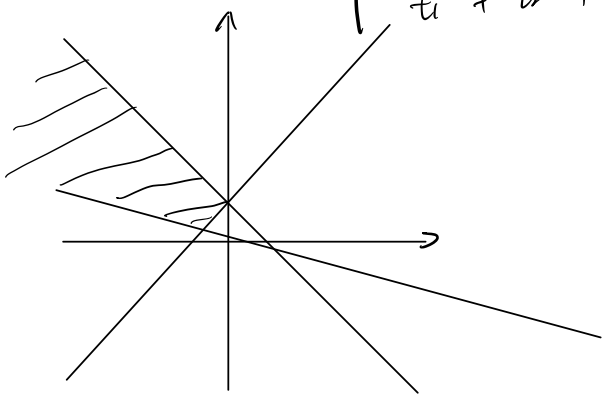
Since we know that $(x_i, y_i) = (-1, 1)$,

(3, 1), then if the data set is separable, then it means that there exist t s.t $t < -1$ and $t < 3$, so if we take a t from -1 to 3, the y_i should be 1, however, when $x_i = 1$, $y_i = 0$, it contradict. Therefore, it's not linearly separable.

2. If we assume $y_i = 1$, then we bring -1 into the equation, $t_1 \times (-1) + t_2 \cdot (-1)^2 \geq 1$ and at the same time, $t_1 \times 3 + t_2 \times (3)^2 \geq 1$, for $y_i = 0$,

we know $t_1 \times 1 + t_2 \times 1^2 < 1$ $\Rightarrow$ $\begin{cases} -t_1 + t_2 \geq 1, & y_i = 1 \\ 3t_1 + 9t_2 \geq 1, & y_i = 1 \\ t_1 + t_2 < 1, & y_i = 0 \end{cases}$ after

we plot the graph



- **Problem 4 (12 pts)**
  For this question you will build classifiers to label images of handwritten digits. Each image is 8 by 8 pixels and is represented as a vector of dimension 64 by listing all the pixel values in raster scan order. The images are grayscale and the pixel values are between 0 and 1. The labels y are $\{0, 1, 2, \ldots, 9\}$ corresponding to which character was written in the image. There are 700 training points and 400 test points for each digit; they can be found in `digits_train.txt` and `digits_test.txt`. These data sets can be loaded by using the helper function in `utils.R`.

  You will implement both linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA) to classify these images. Recall that conditioning on each class $k \in \{0, 1, \ldots, 9\}$, the feature $X \mid Y = k$ follows a multivariate Gaussian distribution, that is,

$$\mathbb{P}(X = \mathbf{x} \mid Y = k) = (2\pi)^{-p/2} |\mathbf{\Sigma}_k|^{-1/2} \exp\left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_k)^\top \mathbf{\Sigma}_k^{-1} (\mathbf{x} - \boldsymbol{\mu}_k) \right\} \tag{0.2}$$

  where $\boldsymbol{\mu}_k \in \mathbb{R}^p$ is the conditional mean and $\mathbf{\Sigma}_k \in \mathbb{R}^{p \times p}$ is the conditional covariance matrix. For LDA, $\mathbf{\Sigma}_k$ is assumed to be the same across classes. The priors are

$$\pi_k = \mathbb{P}(Y = k), \qquad \text{for all } \; k \in \{0, 1, \ldots, 9\}.$$

  You will compute the maximum likelihood estimators of the priors π_k, the conditional means $\boldsymbol{\mu}_k$ and the conditional covariance matrices $\mathbf{\Sigma}_k$ for $k \in \{0, 1, \ldots, 9\}$, and use the estimators to construct classifiers.

  Read carefully the structure of `discriminant_analysis.R`. Include your code for all subquestions.

  1. **(4 pts)** Complete the functions `Comp_priors`, `Comp_cond_means` and `Comp_cond_covs` in the file `discriminant_analysis.R`.
  2. **(2 pts)** Complete the functions `Predict_posterior` and `Predict_labels` in the file `discriminant_analysis.R`.
  3. **(2 pts)** Use LDA to classify the test data by completing part a in `hw4_starter.R`. Report the misclassification error of LDA.
  4. **(2 pts)** Use QDA to classify the test data by completing part b in `hw4_starter.R`. Report the misclassification error of QDA.
  5. **(2 pts)** Complete part c in `hw4_starter.R`, i.e. perform LDA and QDA by using the built-in `lda` and `qda` functions and compare with your implementation in terms of both misclassification rates and computational speed.

3. We know that according to the calculation, the misclassification error of LDA is 0.10225.

4. We know that according to the calculation, the misclassification error of RDA is 0.04075.

5. After compute the build-in lda, qda functions, we can test the error rate is same but the build-in function is faster.

Yanjie Hu STA314 hw4

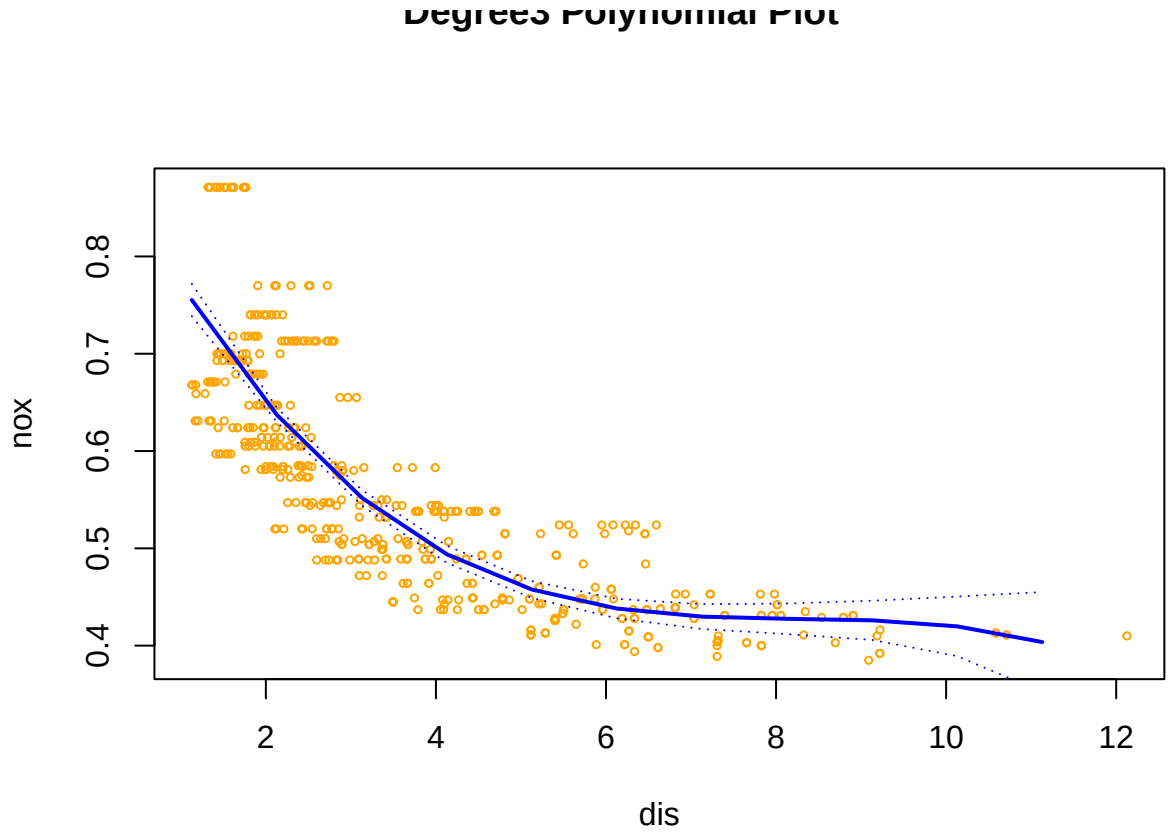
Problem 5

1.

```
library(MASS)
data(Boston)
attach(Boston)
fit <- lm(nox ~ poly(dis,3))
summary(fit)
```

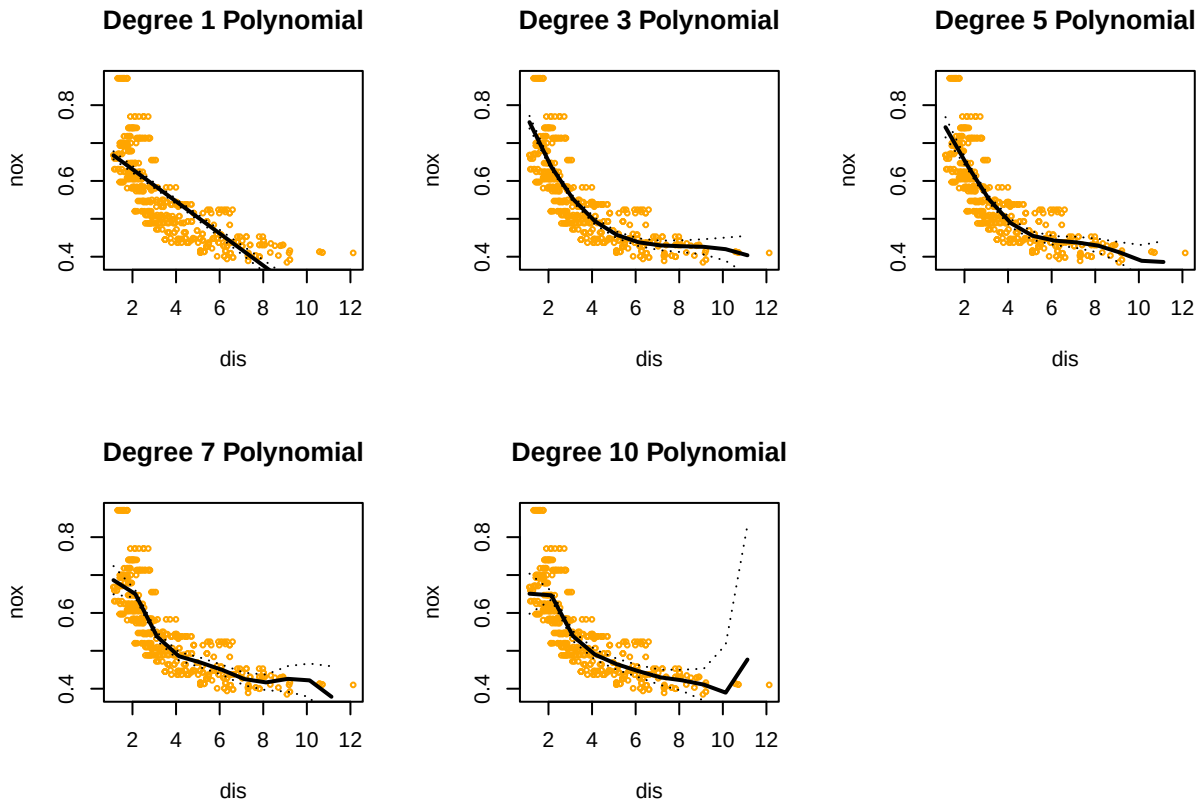
```
##
## Call:
## lm(formula = nox ~ poly(dis, 3))
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.121130 -0.040619 -0.009738  0.023385  0.194904
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.554695   0.002759 201.021  < 2e-16 ***
## poly(dis, 3)1 -2.003096   0.062071 -32.271  < 2e-16 ***
## poly(dis, 3)2  0.856330   0.062071  13.796  < 2e-16 ***
## poly(dis, 3)3 -0.318049   0.062071  -5.124 4.27e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.06207 on 502 degrees of freedom
## Multiple R-squared:  0.7148, Adjusted R-squared:  0.7131
## F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16
```

```
dislims <- range(dis)
dis.grid <- seq(from = dislims[1], to = dislims[2])
preds <- predict(fit, newdata = list(dis=dis.grid), se=TRUE)
se.bands <- cbind(preds$fit+2*preds$se.fit,preds$fit-2*preds$se.fit)
plot(dis, nox, xlim=dislims, cex=.5, col="orange")
title("Degree3 Polynomial Plot", outer=T)
lines(dis.grid, preds$fit, lwd=2, col="blue")
matlines(dis.grid, se.bands, lwd=1, col="blue", lty=3)
```



2.

```
par(mfrow=c(2,3))
d <- c(1,3,5,7,10)
RSS <- numeric(length(d))
for (i in 1:length(d)) {
  fit <- lm(nox ~ poly(dis,d[i]))
  RSS[i] <- sum(fit$residuals^2)
  names(RSS)[i] <- paste0("Degree-", d[i], " Polynomial")
  preds <- predict(fit, newdata = list(dis=dis.grid), se=TRUE)
  se.bands <- cbind(preds$fit+2*preds$se.fit,preds$fit-2*preds$se.fit)
  plot(dis, nox, xlim=dislims, cex=.5, col="orange",
       main=paste0("Degree ", d[i], " Polynomial"))
  lines(dis.grid, preds$fit, lwd=2, col="black")
  matlines(dis.grid, se.bands, lwd=1, col="black", lty=3)
}
```

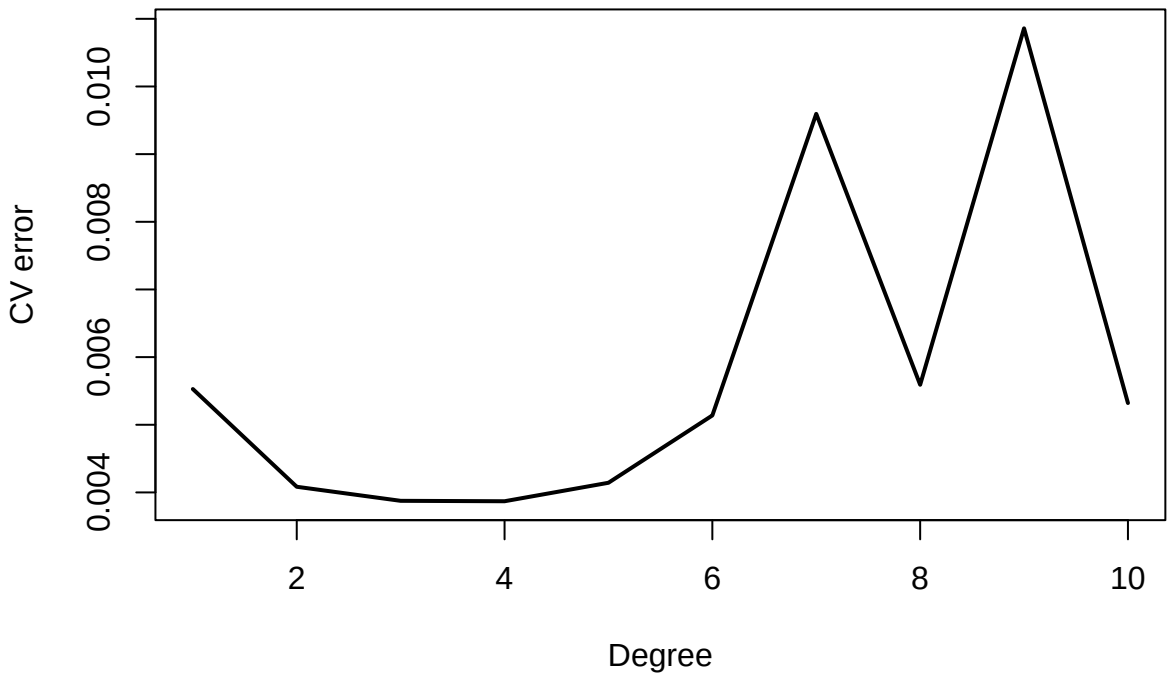



```
RSS
```

```
##  Degree-1 Polynomial  Degree-3 Polynomial  Degree-5 Polynomial
##             2.768563             1.934107             1.915290
##  Degree-7 Polynomial Degree-10 Polynomial
##             1.849484             1.832171
```

3.

```
library(boot)
set.seed(44)
deltas = rep(NA, 10)
for (i in 1:10) {
    glm.fit = glm(nox ~ poly(dis, i), data = Boston)
    deltas[i] = cv.glm(Boston, glm.fit, K = 10)$delta[2]
}
plot(1:10, deltas, xlab = "Degree", ylab = "CV error", type = "l", pch = 20,
    lwd = 2)
```

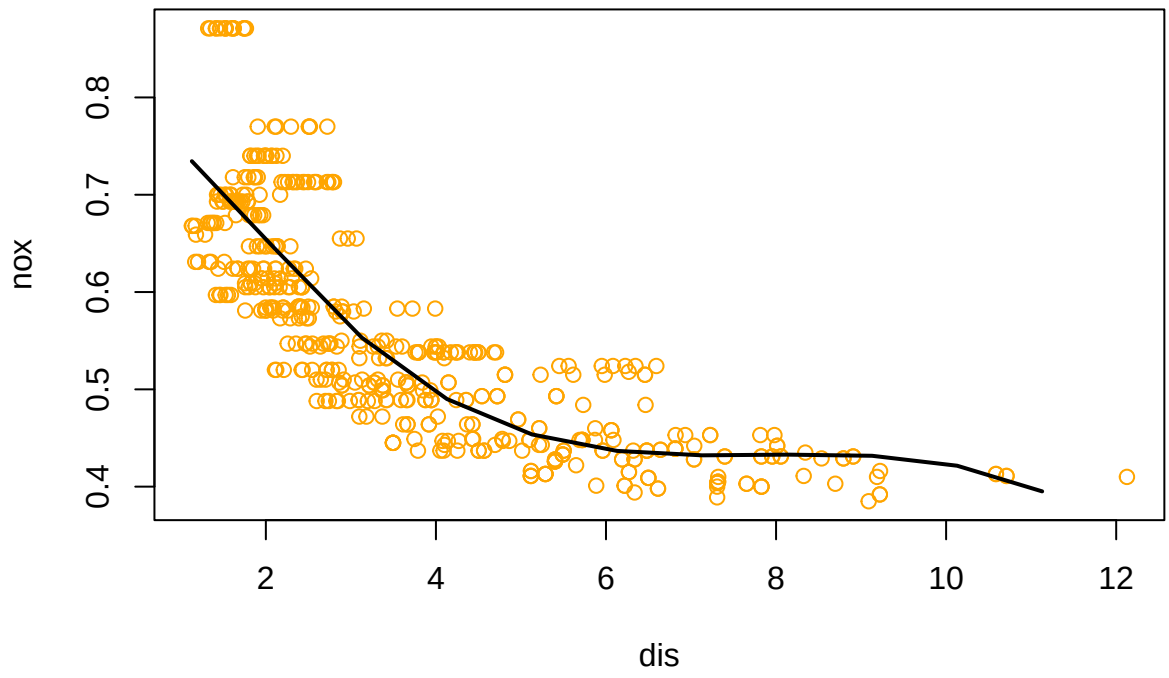


We can say that when the degree is 3 we will have the smallest error.

4.

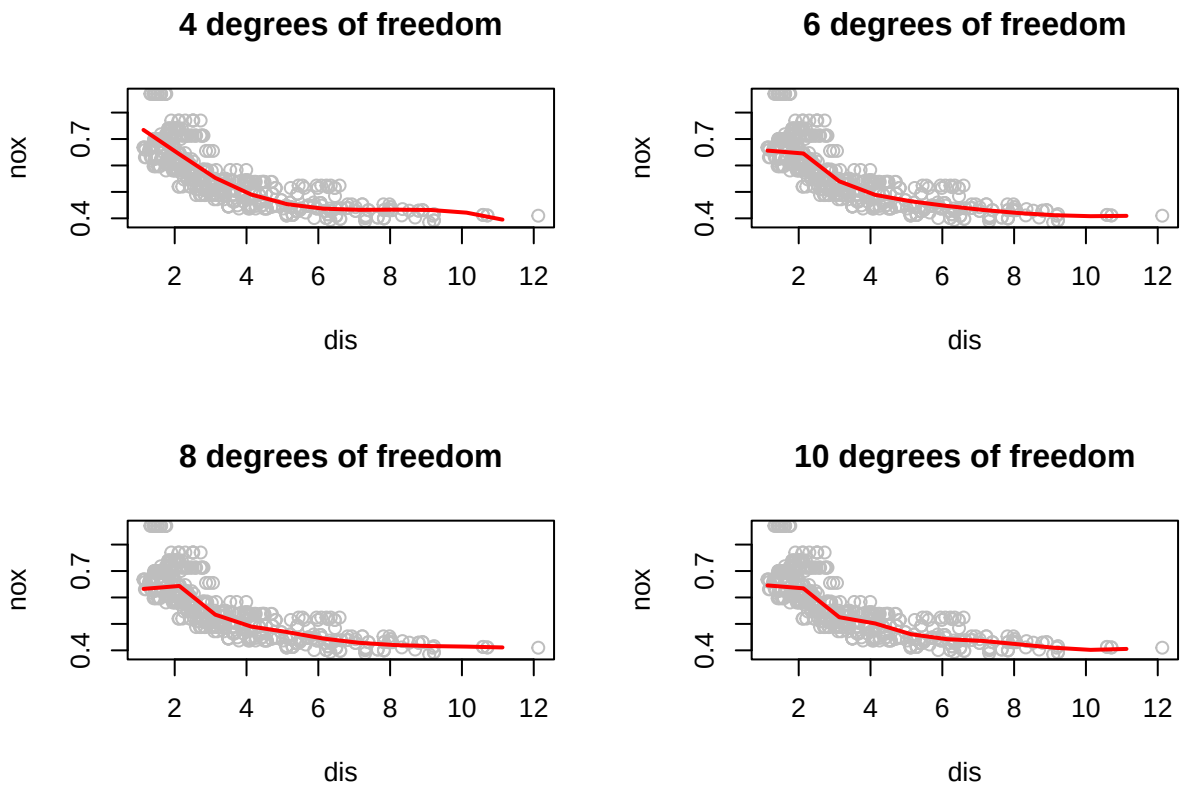
```
library(splines)
fit_nsplines <- lm(nox ~ bs(dis, df=4))
```

```
pred_nsplines <- predict(fit_nsplines, newdata=list(dis=dis.grid), se=T)
plot(dis, nox, col="orange")
lines(dis.grid, pred_nsplines$fit,col="black",lwd=2)
```



5.

```
par(mfrow=c(2,2))
df <- c(4,6,8,10)
RSS <- numeric(length(df))
for (i in 1:length(df)) {
  fit_nsplines <- lm(nox ~ bs(dis, df=df[i]))
  RSS[i] <- sum(fit_nsplines$residuals^2)
  names(RSS)[i] <- paste0(df[i], " degrees of freedom")
  pred_nsplines <- predict(fit_nsplines, newdata=list(dis=dis.grid), se=T)
  plot(dis, nox, col="gray",
       main=paste0(df[i], " degrees of freedom"))
  lines(dis.grid, pred_nsplines$fit,col="red",lwd=2)
}
```




```
RSS
```

```
##  4 degrees of freedom  6 degrees of freedom  8 degrees of freedom
##              1.922775              1.833966              1.816995
## 10 degrees of freedom
##              1.792535
```

When the degree of polynomial is getting larger, the rss is smaller. **6.**

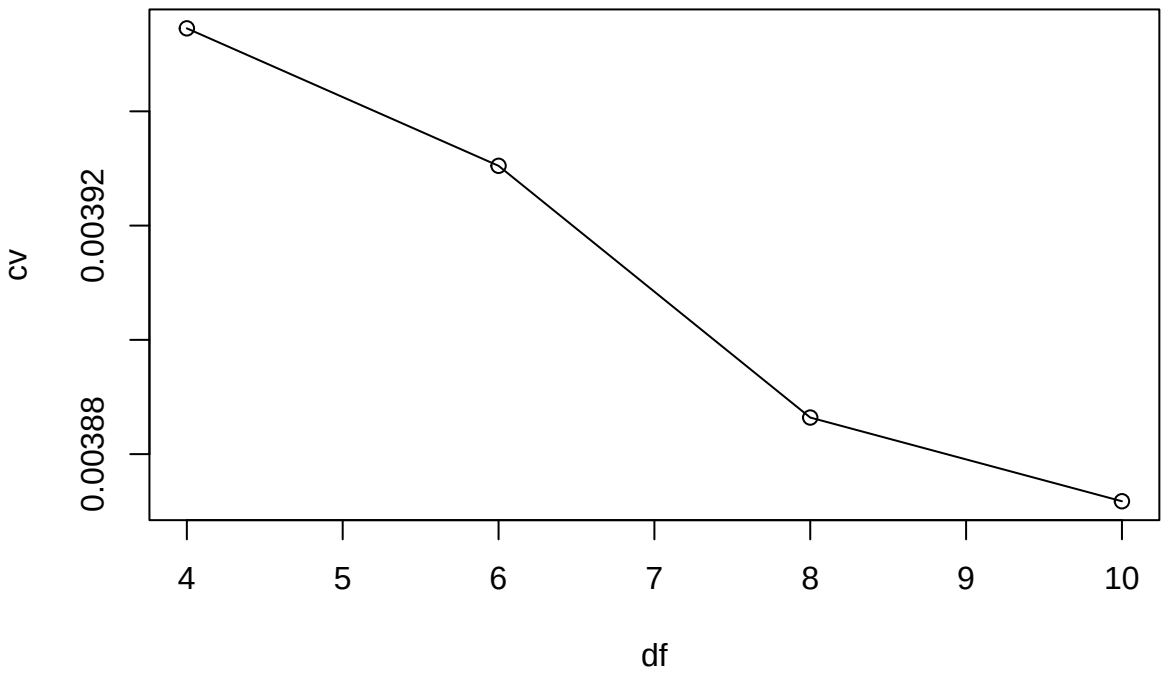
```
set.seed(44)
cv = rep(0,4)
df <- c(4,6,8,10)
for (i in 1:length(df)) {
    glm.fit = glm(nox ~ bs(dis, df = 4), data = Boston)
    cv[i] = cv.glm(Boston, glm.fit, K = 10)$delta[1]
}
```

```
## Warning in bs(dis, degree = 3L, knots = c('50%' = 3.0993), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c('50%' = 3.0993), Boundary.knots =
## c(1.1296, : some 'x' values beyond boundary knots may cause ill-conditioned
## bases
```

```
## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.2157), Boundary.knots =
## c(1.137, : some ’x’ values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.2157), Boundary.knots =
## c(1.137, : some ’x’ values beyond boundary knots may cause ill-conditioned bases


## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.19095), Boundary.knots =
## c(1.137, : some ’x’ values beyond boundary knots may cause ill-conditioned bases

## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.19095), Boundary.knots =
## c(1.137, : some ’x’ values beyond boundary knots may cause ill-conditioned bases


## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.1675), Boundary.knots =
## c(1.1296, : some ’x’ values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.1675), Boundary.knots =
## c(1.1296, : some ’x’ values beyond boundary knots may cause ill-conditioned
## bases


## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.2628), Boundary.knots =
## c(1.1691, : some ’x’ values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.2628), Boundary.knots =
## c(1.1691, : some ’x’ values beyond boundary knots may cause ill-conditioned
## bases


## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.26745), Boundary.knots =
## c(1.1691, : some ’x’ values beyond boundary knots may cause ill-conditioned
## bases

## Warning in bs(dis, degree = 3L, knots = c(‘50%‘ = 3.26745), Boundary.knots =
## c(1.1691, : some ’x’ values beyond boundary knots may cause ill-conditioned
## bases
```

```
plot(df, cv)
lines(df, cv)
```



When the degree of freedom reaches 10, the cross-validation preform the best.