Corrige IFT6266 - TP 2

20 septembre 2006

1. *Considérez un probléme de classification à N classes où l'on veut minimiser non pas la probabilité d'erreur mais plutôt un coût spécificié par une matrice C de dimension $N \times N$ avec l'élément C_{ij} indiquant le coût de la décision $f(x) = i$ quand $y = j$, pour un exemple (x, y) et une fonction de décision f. Si on vous fournit un estimateur $P(Y = i|X = x)$ de la probabilité conditionnelle pour les N classes étant donné x, quelle serait la fonction de décision f qui minimise le coût espéré selon ce P?*

   Pour un point x fixé, le coût de la décision $f(x)$ a une espérance égale à

$$E_Y[C_{f(x)Y}|X = x] = \sum_{j=1}^{N} C_{f(x)j} P(Y = j|X = x)$$

   et la meilleure fonction f est donc celle qui, à tout x, associe

$$f(x) = \text{argmin}_{i \in \{1,\ldots,N\}} \sum_{j=1}^{N} C_{ij} P(Y = j|X = x).$$

2. *La divergence de Kullback-Liebler $KL(p||q)$ entre deux densités p et q est*

$$\text{KL}(p||q) = \int p(x) \log \frac{p(x)}{q(x)} dx.$$

   Calculer cette divergence quand p et q sont deux normales univariées de paramètres (μ, σ) et (m, s) respectivement. Vérifiez que le résultat est toujours positif sauf quand $m = \mu$ et $s = \sigma$ (truc utile: $\log(x) \leq x - 1$).

$$\begin{aligned}
\text{KL}(p\|q) &= \int_{\mathbb{R}} p(x) \ln \frac{p(x)}{q(x)} dx \\
&= \int_{\mathbb{R}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \ln \left(\frac{s}{\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2} + \frac{(x-m)^2}{2s^2}} \right) dx \\
&= \ln\left(\frac{s}{\sigma}\right) \int_{\mathbb{R}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx + \frac{1}{\sigma\sqrt{2\pi}} \int_{\mathbb{R}} \left(\frac{(x-m)^2}{2s^2} - \frac{(x-\mu)^2}{2\sigma^2} \right) e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx.
\end{aligned}$$

   L'intégrale dans le premier terme vaut simplement 1, puisque c'est l'intégrale d'une densité Gaussienne sur $\mathbb{R}$. Pour calculer le second terme, faisons le changement de variable

$$\begin{aligned}
z &= \frac{x - \mu}{\sigma} \\
x &= \sigma z + \mu
\end{aligned}$$

   ce qui donne :

$$\begin{aligned}
\text{KL}(p\|q) &= \ln\left(\frac{s}{\sigma}\right) + \frac{1}{2\sigma\sqrt{2\pi}} \int_{\mathbb{R}} \left(\frac{(\sigma z + \mu - m)^2}{s^2} - z^2 \right) e^{-\frac{z^2}{2}} \sigma dz \\
&= \ln\left(\frac{s}{\sigma}\right) + \frac{1}{2\sqrt{2\pi}} \int_{\mathbb{R}} \left(\left(\frac{\sigma^2}{s^2} - 1 \right) z^2 + 2\frac{\sigma(\mu - m)}{s^2} z + \frac{(\mu - m)^2}{s^2} \right) e^{-\frac{z^2}{2}} dz. \quad (1)
\end{aligned}$$

Sachant que l'intégrale d'une densité Gaussienne de moyenne 0 et de variance 1 est égale à 1, on a :

$$\int_{\mathbb{R}} e^{-\frac{z^2}{2}} dz = \sqrt{2\pi}. \tag{2}$$

D'autre part,

$$\int_{\mathbb{R}} z e^{-\frac{z^2}{2}} dz = \left[-e^{-\frac{z^2}{2}}\right]_{-\infty}^{+\infty} = 0 \tag{3}$$

et on peut calculer $\int_{\mathbb{R}} z^2 e^{-\frac{z^2}{2}} dz$ par intégration par parties, en notant $f(z) = z$ et $g'(z) = ze^{-\frac{z^2}{2}}$:

$$\begin{aligned}
\int_{\mathbb{R}} z^2 e^{-\frac{z^2}{2}} dz = \int_{\mathbb{R}} f(z)g'(z)dz &= [f(z)g(z)]_{-\infty}^{+\infty} - \int_{\mathbb{R}} f'(z)g(z)dz \\
&= \left[-ze^{-\frac{z^2}{2}}\right]_{-\infty}^{+\infty} + \int_{\mathbb{R}} e^{-\frac{z^2}{2}} dz \\
&= \sqrt{2\pi}.
\end{aligned} \tag{4}$$

En utilisant les équations (2), (3) et (4) dans (1), on obtient :

$$\begin{aligned}
\mathrm{KL}(p\|q) &= \ln\left(\frac{s}{\sigma}\right) + \frac{1}{2\sqrt{2\pi}}\left(\frac{\sigma^2}{s^2} - 1 + \frac{(\mu - m)^2}{s^2}\right)\sqrt{2\pi} \\
&= -\ln\left(\frac{\sigma}{s}\right) + \frac{1}{2}\left(\frac{\sigma^2}{s^2} - 1 + \frac{(\mu - m)^2}{s^2}\right).
\end{aligned}$$

On va utiliser l'inégalité $-\ln(x) \geq 1 - x$ (avec égalité ssi $x = 1$) avec $x = \frac{\sigma}{s}$, ce qui donne l'inégalité

$$\mathrm{KL}(p\|q) \geq 1 - \frac{\sigma}{s} + \frac{\sigma^2}{2s^2} - \frac{1}{2} + \frac{(\mu - m)^2}{2s^2} = \frac{1}{2}(\frac{\sigma}{s} - 1)^2 + \frac{(m - \mu)^2}{2s^2}$$

On a finalement que

$$\mathrm{KL}(p\|q) \geq 0$$

avec égalité ssi $\sigma = s$, $\frac{\sigma}{s} = 1$ et $\mu - m = 0$, i.e. $\sigma = s$ et $\mu = m$.

3. *Vous allez démontrer le résultat de la régression linéaire Bayesienne, c'est à dire avec au départ une distribution à priori sur les paramètres w et à l'arrivée une distribution à postériori sur ces mêmes paramètres, après avoir vu les données $D = \{(x_t, y_t)\}_{t=1}^n$, $x_t \in \mathbb{R}^d$, $y \in \mathbb{R}$. On suppose $Y|X = x$ normal de variance σ^2 (connue pour simplifier) et d'espérance $w'x$. Soit $p(w) = \mathcal{N}(w|m_0, S_0)$ la loi à priori sur w, et notons $\mathbf{X}$ la matrice dont les rangées sont les x_t et $\mathbf{Y}$ le vecteur colonne dont les éléments sont les y_t.*

   En utilisant la formule de Bayes :

$$\begin{aligned}
p(w|\mathbf{X}, \mathbf{Y}) &= \frac{p(\mathbf{X}, \mathbf{Y}|w)p(w)}{p(\mathbf{X}, \mathbf{Y})} \\
&= \frac{p(\mathbf{Y}|\mathbf{X}, w)p(\mathbf{X}|w)p(w)}{p(\mathbf{Y}|\mathbf{X})p(\mathbf{X})} \\
&= \frac{p(\mathbf{Y}|\mathbf{X}, w)p(w)}{p(\mathbf{Y}|\mathbf{X})}
\end{aligned}$$

   car $P(\mathbf{X}|w) = P(\mathbf{X})$ puisque $\mathbf{X}$ et w sont indépendants. On a donc (en utilisant que les exemples sont tirés indépendamment et que le dénominateur ne dépend pas de w) :

$$\begin{aligned}
p(w|\mathbf{X}, \mathbf{Y}) &\propto \Pi_{t=1}^n p(y_t|\mathbf{X}, w)p(w) \\
&\propto \Pi_{t=1}^n p(y_t|x_t, w) \exp\left(-\frac{1}{2}(w - m_0)' S_0^{-1}(w - m_0)\right)
\end{aligned}$$

$$
\begin{aligned}
&\propto \exp\left(-\frac{1}{2}\left(\sum_{t=1}^{n}\frac{(y_t-w'x_t)^2}{\sigma^2}+(w-m_0)'S_0^{-1}(w-m_0)\right)\right)\\
&\propto \exp\left(-\frac{1}{2}\left(\sum_{t=1}^{n}\left(\frac{y_t^2}{\sigma^2}-2w'\left(\frac{y_tx_t}{\sigma^2}\right)+w'\frac{x_tx_t'}{\sigma^2}w\right)+(w-m_0)'S_0^{-1}(w-m_0)\right)\right)\\
&\propto \exp\left(-\frac{1}{2}\left(w'\left(S_0^{-1}+\sum_{t=1}^{n}\frac{x_tx_t'}{\sigma^2}\right)w-2w'\left(S_0^{-1}m_0+\sum_{t=1}^{n}\frac{y_tx_t}{\sigma^2}\right)\right)\right)\\
&\propto \exp\left(-\frac{1}{2}(w-m_n)'S_n^{-1}(w-m_n)\right) \qquad (5)
\end{aligned}
$$

avec

$$
\begin{aligned}
S_n &= \left(S_0^{-1}+\sum_{t=1}^{n}\frac{x_tx_t'}{\sigma^2}\right)^{-1}=\left(S_0^{-1}+\frac{\mathbf{X'X}}{\sigma^2}\right)^{-1} \qquad (6)\\
m_n &= S_n\left(S_0^{-1}m_0+\sum_{t=1}^{n}\frac{y_tx_t}{\sigma^2}\right)=S_n\left(S_0^{-1}m_0+\frac{\mathbf{X'Y}}{\sigma^2}\right)
\end{aligned}
$$

Comme la constante de proportionalité dans (5) ne dépend pas de w, la forme de cette équation montre que $w|\mathbf{X},\mathbf{Y}$ suit une loi normale de moyenne m_n et covariance S_n.

Contrairement à la régression linéaire ordinaire, on obtient non seulement une prédiction mais aussi une incertitude sur les prédictions qui sera d'autant plus petite qu'il y a beaucoup d'exemples. Utilisez la formule ci-haut pour obtenir la distribution à postériori sur Y pour un nouvel exemple $X=x$, i.e., donnez une formule pour $p(Y|X=x,\mathbf{X},\mathbf{Y})$.

Dans les équations qui suivent, on utilise le signe de proportionalité ($\propto$) pour la multiplication par une constante qui ne dépend ni de y, ni de w.

$$
\begin{aligned}
&p(Y=y|X=x,\mathbf{X},\mathbf{Y})\\
=\;&\int_w p(Y=y,w|X=x,\mathbf{X},\mathbf{Y})dw\\
=\;&\int_w p(Y=y|X=x,\mathbf{X},\mathbf{Y},w)p(w|X=x,\mathbf{X},\mathbf{Y})dw\\
=\;&\int_w p(Y=y|X=x,w)p(w|\mathbf{X},\mathbf{Y})dw\\
\propto\;&\int_w \exp\left(-\frac{(y-w'x)^2}{2\sigma^2}\right)\exp\left(-\frac{1}{2}(w-m_n)'S_n^{-1}(w-m_n)\right)dw\\
\propto\;&\exp\left(-\frac{y^2}{2\sigma^2}\right)\int_w \exp\left(-\frac{1}{2}\left(w'\left(S_n^{-1}+\frac{xx'}{\sigma^2}\right)w-2w'\left(S_n^{-1}m_n+\frac{yx}{\sigma^2}\right)\right)\right)dw\\
\propto\;&\exp\left(-\frac{y^2}{2\sigma^2}\right)\int_w \exp\left(-\frac{1}{2}(w-\bar{m}_n)'\bar{S}_n^{-1}(w-\bar{m}_n)+\frac{1}{2}\bar{m}_n'\bar{S}_n^{-1}\bar{m}_n\right)dw
\end{aligned}
$$

avec

$$
\begin{aligned}
\bar{S}_n &= \left(S_n^{-1}+\frac{xx'}{\sigma^2}\right)^{-1}\\
\bar{m}_n &= \bar{S}_n\left(S_n^{-1}m_n+\frac{yx}{\sigma^2}\right).
\end{aligned}
$$

En utilisant que l'intégrale d'une loi normale est une constante qui ne dépend que de sa matrice de covariance (ici $\bar{S}_n$, indépendante de y), on obtient alors :

$$
\begin{aligned}
p(Y=y|X=x,\mathbf{X},\mathbf{Y}) &\propto \exp\left(-\frac{1}{2}\left(\frac{y^2}{\sigma^2}-\bar{m}_n'\bar{S}_n^{-1}\bar{m}_n\right)\right)\\
&\propto \exp\left(-\frac{1}{2}\left(\frac{y^2}{\sigma^2}-\left(S_n^{-1}m_n+\frac{yx}{\sigma^2}\right)'\bar{S}_n\left(S_n^{-1}m_n+\frac{yx}{\sigma^2}\right)\right)\right)
\end{aligned}
$$

$$\begin{aligned} &\propto \exp\left(-\frac{1}{2}\left(y^2\left(\frac{1}{\sigma^2}-\frac{x'}{\sigma^2}\bar{S}_n\frac{x}{\sigma^2}\right)-2y\left(\frac{x'}{\sigma^2}\bar{S}_nS_n^{-1}m_n\right)\right)\right) \\ &\propto \exp\left(-\frac{1}{2}\frac{(y-\mu)^2}{s^2}\right) \end{aligned}$$

avec

$$s^2 = \left(\frac{1}{\sigma^2}-\frac{x'}{\sigma^2}\bar{S}_n\frac{x}{\sigma^2}\right)^{-1} = \frac{\sigma^2}{1-x'\left(\sigma^2S_n^{-1}+xx'\right)^{-1}x} \quad (7)$$

$$\mu = s^2\left(\frac{x'}{\sigma^2}\bar{S}_nS_n^{-1}m_n\right) = s^2x'\left(\sigma^2S_n^{-1}+xx'\right)^{-1}S_n^{-1}m_n \quad (8)$$

et $Y|X=x,\mathbf{X},\mathbf{Y}$ suit donc une loi normale de moyenne μ et variance s^2.

Ces expressions peuvent être simplifiées si l'on utilise l'identité de Woodbury (cf. le livre de recettes de matrices) pour calculer

$$\left(\sigma^2S_n^{-1}+xx'\right)^{-1} = \frac{S_n}{\sigma^2}-\frac{S_n}{\sigma^2}x\left(1+x'\frac{S_n}{\sigma^2}x\right)^{-1}x'\frac{S_n}{\sigma^2}.$$

En utilisant cette expression dans (7), on obtient

$$\begin{aligned} s^2 &= \frac{\sigma^2}{1-x'\frac{S_n}{\sigma^2}x+x'\frac{S_n}{\sigma^2}x\left(1+x'\frac{S_n}{\sigma^2}x\right)^{-1}x'\frac{S_n}{\sigma^2}x} \\ &= \sigma^2\left(\frac{1+x'\frac{S_n}{\sigma^2}x-x'\frac{S_n}{\sigma^2}x\left(1+x'\frac{S_n}{\sigma^2}x\right)+\left(x'\frac{S_n}{\sigma^2}x\right)^2}{1+x'\frac{S_n}{\sigma^2}x}\right)^{-1} \\ &= \sigma^2\left(1+x'\frac{S_n}{\sigma^2}x\right) \\ &= \sigma^2+x'S_nx. \end{aligned} \quad (9)$$

On peut alors réécrire (8) comme suit :

$$\begin{aligned} \mu &= \left(\sigma^2+x'S_nx\right)x'\left(\frac{S_n}{\sigma^2}-\frac{S_n}{\sigma^2}x\left(1+x'\frac{S_n}{\sigma^2}x\right)^{-1}x'\frac{S_n}{\sigma^2}\right)S_n^{-1}m_n \\ &= \left(\sigma^2+x'S_nx\right)\left(\frac{x'm_n}{\sigma^2}-x'S_nx\left(\sigma^2+x'S_nx\right)^{-1}\frac{x'm_n}{\sigma^2}\right) \\ &= \left(\sigma^2+x'S_nx\right)\frac{x'm_n}{\sigma^2}-x'S_nx\frac{x'm_n}{\sigma^2} \\ &= x'm_n \end{aligned}$$

ce qui est conforme à l'intuition que l'espérance de Y devrait être le produit scalaire de l'entrée $X=x$ et du vecteur de poids moyen $m_n = E[w|\mathbf{X},\mathbf{Y}]$. D'autre part, de manière intuitive, lorsque n augmente, d'après (6) les valeurs propres de S_n sont plus petites, donc $x'S_nx$ est plus petit, et (9) montre donc que la variance de Y est plus petite (et tend vers le bruit σ^2 lorsque $n\to\infty$) : l'incertitude sur la valeur de Y diminue.

4. *(BONUS) Vous allez montrer que si l'enveloppe convexe de deux ensembles de points ont une intersection, alors on ne peut les séparer avec un classifieur linéaire. Soit $\{u_1,\ldots,u_n\}$ des exemples de la classe 1 et $\{v_1,\ldots,v_m\}$ des exemples de la classe 2, avec $u_i\in\mathbb{R}^d$, $v_j\in\mathbb{R}^d$. Pour chaque ensemble d'exemples on définit l'enveloppe convexe associée. Par exemple pour les u_i c'est $\{x\in\mathbb{R}^d | x=\sum_i\alpha_iu_i, \alpha_i\geq 0, \sum_i\alpha_i=1\}$. Vérifier d'abord que si on pouvait trouver un classifieur linéaire qui sépare les u_i des v_j cela voudrait dire qu'il existe $w\in\mathbb{R}^d$ tel que $w'u_i > w'v_j$, $\forall i,j$. Je vous suggère ensuite de procéder en montrant que si on* **peut** *trouver un tel w alors on* **ne peut** *avoir une intersection des enveloppes convexes.*

Soit $U = \{u_1, \ldots, u_n\}$ et $V = \{v_1, \ldots, v_m\}$ les deux ensembles dont on considère les enveloppes convexes $C(U)$ et $C(V)$. Par l'absurde, supposons qu'il existe un classifieur linéaire (défini par un vecteur de poids w et un biais b) qui sépare U et V, et que $C(U) \cap C(V) \neq \emptyset$. Alors on a d'une part que, pour tout $u_i \in U$ et pour tout $v_j \in V$,

$$(w'u_i + b)(w'v_j + b) < 0 \tag{10}$$

et, d'autre part, qu'il existe un point $x \in C(U) \cap C(V)$, qui vérifie donc

$$x = \sum_i \alpha_i u_i = \sum_j \beta_j v_j$$

avec $\alpha_i \geq 0$, $\beta_j \geq 0$, et $\sum_i \alpha_i = \sum_j \beta_j = 1$.

On peut alors écrire :

$$\begin{aligned}
(w'x + b)^2 &= (w'x + b)(w'x + b) \\
&= \left(w' \sum_i \alpha_i u_i + b \sum_i \alpha_i \right) \left(w' \sum_j \beta_j v_j + b \sum_j \beta_j \right) \\
&= \left(\sum_i \alpha_i (w'u_i + b) \right) \left(\sum_j \beta_j (w'v_j + b) \right) \\
&= \sum_{i,j} \alpha_i \beta_j (w'u_i + b)(w'v_j + b) \\
&< 0
\end{aligned}$$

en utilisant (10) et le fait que les α_i et β_j sont positifs ou nuls, avec au moins un α_i et un β_j non nuls. Comme un nombre élevé au carré ne peut pas être strictement négatif, on en déduit que notre hypothèse de départ était fausse, donc qu'on ne peut pas avoir un classifieur linéaire séparant U et V lorsque $C(U) \cap C(V) \neq \emptyset$.

5. *Vous allez faire une implantation de l'algorithme de régression à noyau aussi appelé fenêtres de Parzen ou encore modèle de Nadaraya-Watson. (...)*

   La figure 1 montre l'erreur de validation en fonction de σ. La valeur optimale de σ est 0.025, pour une erreur moyenne de 0.713 sur l'ensemble de validation.

   La figure 2 montre la sortie de notre estimateur pour $\sigma = 0.025$, ainsi que les points d'entraînement et de validation. On y reconnaît vaguement une fonction oscillante, sans doute assez éloignée de la "vraie" fonction génératrice, à cause du bruit important dans les données.

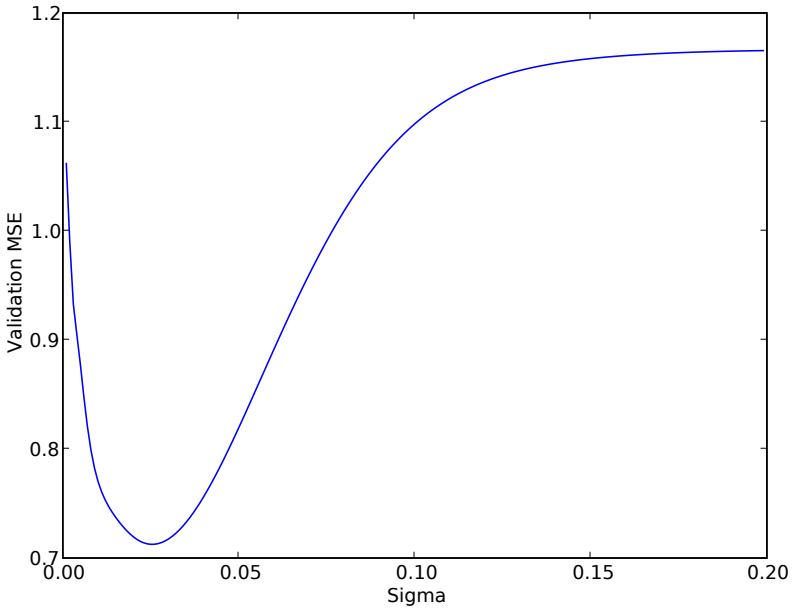

Figure 1: Erreur (MSE) en validation en fonction de σ.

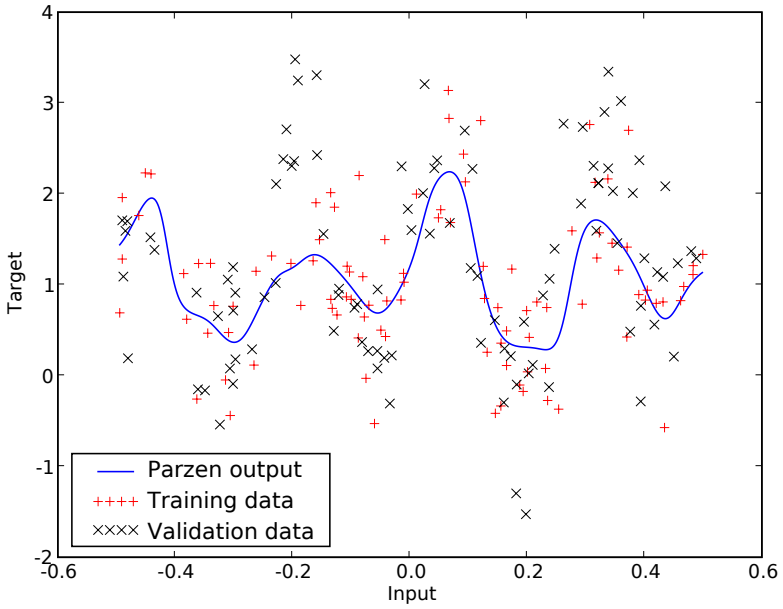

Figure 2: Fonction estimée avec la valeur optimale de σ. Sont aussi représentés les ensembles d'entraînement et de validation.

Le code ci-dessous a été écrit par Olivier Delalleau en Python, et nécessite les librairies Matplotlib et NumPy.

```
from matplotlib.pylab import figure, legend, load, plot, scatter, show, \
                             xlabel, ylabel
from numpy import arange, argmin, array, concatenate, dot, exp, Infinity, \
                  matrix, mean, multiply, repmat, reshape, sum, zeros

############
## parzen ##
############
def parzen(x, train, sigma_square):
    """Compute the output of Parzen windows regression on test points x, with
    training dataset train and Gaussian kernel with variance sigma_square"""
    x_width = x.shape[1]
    k_x_xi = reshape(
            exp(-0.5 * sum((repmat(x, 1, len(train)).reshape(-1, x_width) -
                            repmat(train[:, 0:x_width], len(x), 1))**2,
                            axis = 1) \
                     / sigma_square),
            (len(x), len(train)))
    k_x_xi /= sum(k_x_xi, axis = 1).reshape(-1, 1)
    return dot(k_x_xi, train[:, x_width:])

##########
## main ##
##########
def main():
    """Main body of the program"""

    # Load datasets.
    train = load('train_data.txt')
    valid = load('valid_data.txt')

    # Loop over various sigma values to find the optimal one on validation set.
    sigma_range = arange(0.001, 0.2, 0.001)
    parzen_mse = lambda sigma : \
            mean((valid[:,1:] - parzen(valid[:,0:1], train, sigma**2))**2)
    parzen_exp = map(parzen_mse, sigma_range)
    best_exp = argmin(parzen_exp)
    [ best_sigma, best_error] = [ sigma_range[best_exp], parzen_exp[best_exp] ]
    print 'Best value for sigma: %s (validation MSE = %s)' \
            % (best_sigma, best_error)

    # Plot the validation error against the kernel bandwidth.
    plot(sigma_range, parzen_exp)
    xlabel('Sigma')
    ylabel('Validation MSE')

    # Compute output over the range spanned by training and validation samples,
    # for the best value of sigma that was found.
    sigma = best_sigma
    all_x_values = concatenate( (train[:,0], valid[:,0]) )
    x_range = arange(min(all_x_values), max(all_x_values), 0.001).reshape(-1, 1)
    print 'Computing output values on range [%s, %s]' % \
            (min(x_range), max(x_range))
```

```
    parzen_output = parzen(x_range, train, sigma**2)

    # Plot Parzen output, as well as training and validation samples.
    figure()
    plot(x_range, parzen_output, label = 'Parzen output')
    plot(train[:,0], train[:,1], 'r+', label = 'Training data')
    plot(valid[:,0], valid[:,1], 'kx', label = 'Validation data')
    xlabel('Input')
    ylabel('Target')
    legend(loc = 'lower left')

    # Actually show plots on screen.
    show()

# Run.
main()
```