STA208: Homework 2

Prof. James Sharpnack

Due 4/13 in class

In the following, show all your work. Feel free to do all the analytical questions first and then include the code and output second, but the different parts and which question that you are answering should be clearly marked. Code should be as modular as possible, points will be deducted for code that is not reusable (i.e. not broken into general purpose functions), and in the case of gratuitous hard coding.

1. The following losses are used as surrogate losses for large margin classification. Demonstrate if they are convex or not, and follow the instructions.
   (a) exponential loss: $\phi(x) = e^{-x}$
   (b) truncated quadratic loss: $\phi(x) = (\max\{1-x, 0\})^2$
   (c) hinge loss: $\phi(x) = \max\{1-x, 0\}$
   (d) sigmoid loss: $\phi(x) = 1 - \tanh(\kappa x)$, for fixed $\kappa > 0$
   (e) Plot these as a function of x.

   (This problem is due to notes of Larry Wasserman.)

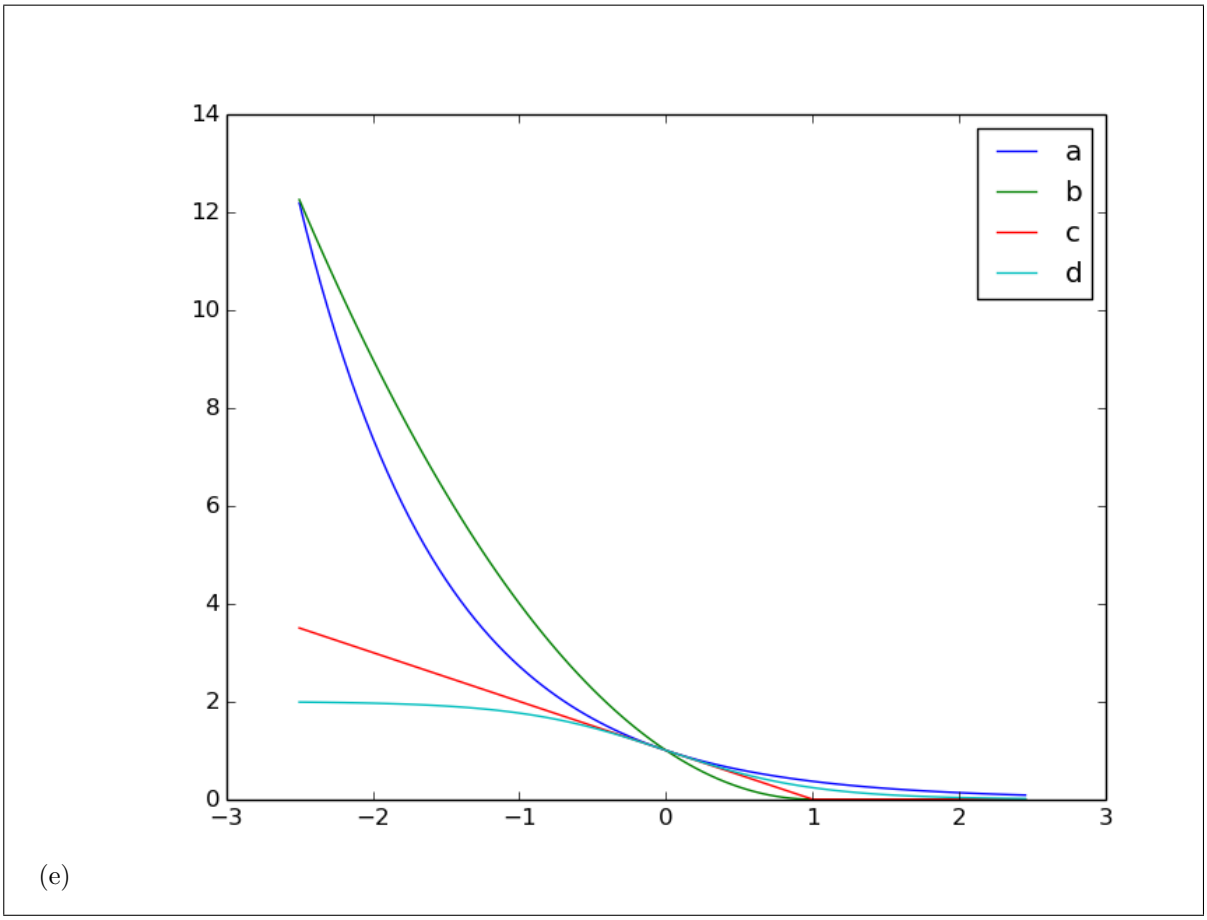
Solution:

(a) $\phi''(x) = e^{-x} > 0$ so it is convex

(b) Define $g(x) = x^2$. $1 - x$ and 0 are convex functions, and the pointwise maximum preserves convexity so $\max\{1-x, 0\}$ is convex (answering c). g is convex and $\phi(x)$ is the composition of two convex functions which also preserves convexity.

(c) See above.

(d) This is not convex, to see this, $\phi''(x) = 2\tanh(x)^2(x)$ which is negative for $x < 0$.



(e)

2. Consider the least-squares problem with n p-dimensional covariates, $\{\mathbf{x}_i, y_i\}_{i=1}^n \subset \mathbb{R}^p \times \mathbb{R}$. We would like to fit the following linear model, $\hat{y}(\mathbf{x}) = \mathbf{x}^\top \boldsymbol{\beta}$. Also, suppose that there are coefficients $C_+ \subset \{1, \ldots, p\}$ such that for all $j \in C_+$ we require that $\beta_j \geq 0, j \in C_+$, and another set $C_- \subset \{1, \ldots, p\}$, such that $\beta_j \leq 0, j \in C_-$ (assume that C_+ and C_- are non-overlapping). Suppose that $\mathbf{X}^\top \mathbf{X}$ is invertible.

   Such examples occur in insurance applications: the cost of a given insurance policy is based on a model for the amount of money a customer will cost the company, and each covariate is a variable specific to the customer (such as gender, age, credit history, etc.). It looks bad for the company if the insurance policy is more expensive for a customer that has an older account with the company than a newer account, when everything else is held fixed. Let $x_{i,j} = 1\{\text{customer } i \text{ is has had a policy for more than 2 years}\}$, then $\beta_j \geq 0$ is necessary for this property to hold.

   *For the sake of this problem being more concrete and easier to understand you may answer the following: Assume that $p = 20$, $C_+ = \{1, 4\}$, $C_- = \{7, 11\}$, which means that you need to solve the program:

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^{20}} \sum_{i=1}^n (y_i - \mathbf{x}_i^\top \boldsymbol{\beta})^2 \quad \text{such that } \beta_1 \geq 0, \beta_4 \geq 0, \beta_7 \leq 0, \beta_{11} \leq 0.$$

   Notice that this is a special case of the above program. Recall the definition of the Lagrangian is

$$\mathcal{L}(\boldsymbol{\beta}, \lambda) = \sum_{i=1}^n (y_i - \mathbf{x}_i^\top \boldsymbol{\beta})^2 + \lambda^\top \mathbf{g}(\boldsymbol{\beta})$$

   where $\mathbf{g}(\boldsymbol{\beta})$ is the vector of constraints that need to be less than or equal to 0, and $\lambda \geq 0, \lambda \in \mathbb{R}^4$. Also, recall that the dual is the following,

$$\ell(\lambda) = \min_{\boldsymbol{\beta} \in \mathbb{R}^{20}} \mathcal{L}(\boldsymbol{\beta}, \lambda).$$

(a) Write the constrained optimization for the empirical risk minimization with the constraints.

(b) Derive the dual for the optimization as a function of dual parameters.

(c) Write the KKT conditions and remark on the implication of the complementary slackness condition (specifically, if $\lambda_i > 0$ what does this tell us about the $\boldsymbol{\beta}$?).

Solution:

(a) The primal program is

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}^\top\boldsymbol{\beta})^2 \quad \text{s.t.} \quad -\beta_j \le 0, j \in C_+, \beta_j \le 0, j \in C_-$$

(b) Let $k_+ = |C_+|$ and $k_- = |C_-|$. Introduce the dual parameters, $\boldsymbol{\gamma}^+ \in \mathbb{R}^{k_+}$ and $\boldsymbol{\gamma}^- \in \mathbb{R}^{k_-}$. Also, define $s_+(j)$ and $s_-(j)$ to be the jth elements of C_+, C_- respectively. Then the Lagrangian of the optimization is

$$\mathcal{L}(\boldsymbol{\beta}, \boldsymbol{\gamma}^+, \boldsymbol{\gamma}^-) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top\boldsymbol{\beta})^2 - \sum_{j=1}^{k_+}\gamma_j^+\beta_{s_+(j)} + \sum_{j=1}^{k_-}\gamma_j^-\beta_{s_-(j)}.$$

Take the gradient with respect to $\boldsymbol{\beta}$,

$$\frac{2}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top\boldsymbol{\beta})\mathbf{x}_j = 0, j \notin C_+ \cup C_-, \tag{1}$$

$$\frac{2}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top\boldsymbol{\beta})\mathbf{x}_j - \gamma_j^+ = 0, s_+(j) \in C_+, \tag{2}$$

$$\frac{2}{n}\sum_{i=1}^{n}(y_i - \mathbf{x}_i^\top\boldsymbol{\beta})\mathbf{x}_j + \gamma_j^- = 0, s_-(j) \in C_-, \tag{3}$$

$$\tag{4}$$

Let $\boldsymbol{\gamma} \in \mathbb{R}^p$ such that $\gamma_j = -\gamma_{j'}^+$ for $j = s_+(j')$, $\gamma_j = \gamma_{j'}^-$ for $j = s_-(j')$, and $\gamma_j = 0$ elsewhere. Then

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}(\mathbf{X}^\top\mathbf{y} + \boldsymbol{\gamma})$$

by the stationarity condition. Plugging this back in to the Lagrangian, we have the following,

$$\begin{aligned}\mathcal{L}(\hat{\boldsymbol{\beta}}, \boldsymbol{\gamma}) &= \frac{1}{n}\|(\mathbf{I} - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top)\mathbf{y} - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\boldsymbol{\gamma}\|_2^2 + \boldsymbol{\gamma}^\top(\mathbf{X}^\top\mathbf{X})^{-1}(\mathbf{X}^\top\mathbf{y} + \boldsymbol{\gamma}) \\ &= \frac{n+1}{n}\boldsymbol{\gamma}^\top(\mathbf{X}^\top\mathbf{X})^{-1}\boldsymbol{\gamma} + \mathbf{y}^\top\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\boldsymbol{\gamma}.\end{aligned}$$

One can see this by $\mathbf{X}^\top(\mathbf{I} - \mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top) = \mathbf{0}$. Hence, the dual program is

$$\max_{\boldsymbol{\gamma}^+, \boldsymbol{\gamma}^- \ge 0} \frac{n+1}{n}\boldsymbol{\gamma}^\top(\mathbf{X}^\top\mathbf{X})^{-1}\boldsymbol{\gamma} + \mathbf{y}^\top\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\boldsymbol{\gamma} \text{ s.t. } \boldsymbol{\gamma}_{C_+} = -\boldsymbol{\gamma}^+, \boldsymbol{\gamma}_{C_-} = \boldsymbol{\gamma}^-, \boldsymbol{\gamma}_{(C_+\cup C_-)^C} = \mathbf{0}.$$

(c) The KKT conditions: the stationarity condition is (1), the remaining conditions are

$$\begin{aligned}&\boldsymbol{\gamma}^+, \boldsymbol{\gamma}^- \ge 0, \\ &\hat{\beta}_j \le 0, j \in C_-, \quad \hat{\beta}_j \ge 0, j \in C_+ \\ &\gamma_j^+\hat{\beta}_{s_+(j)} = 0, j \in C_+, \quad \gamma_j^-\hat{\beta}_{s_-(j)} = 0, j \in C_-.\end{aligned}$$

The complementary slackness says that for $j \in C_+$ if $\gamma_j^+ \neq 0$ then $\hat{\beta}_{s_+(j)} = 0$ necessarily, and similarly for C_-.

3. Look at the dataset which can be found here: `https://archive.ics.uci.edu/ml/datasets/Blood+Transfusion+Service+Center`.

   (a) You will predict the final column in the dataset, which is an indicator if the person has made a blood donation. You probably want to rescale your design matrix. Apply linear SVM and kernel SVM with two kernels (you may use the built in kernels for you package).

   (b) Apply k-nearest neighbors with the usual Euclidean distance. (You can try it also using your kernels to define the distance by $d(i,j) = k(\mathbf{x}_i, \mathbf{x}_i) + k(\mathbf{x}_j, \mathbf{x}_j) - 2k(\mathbf{x}_i, \mathbf{x}_j)$).

   (c) Tune any parameters based on what you have learned about validation, and compare these methods with test errors (there are 4 different methods to compare, kNN and SVM with each kernel).

Solution:

```
import numpy as np
import scipy as sci
from sklearn import linear_model, neighbors, preprocessing, cross_validation, svm
from matplotlib import pyplot as plt

D = np.loadtxt("transfusion.data",dtype=np.str_,delimiter=",")

X = np.array(D[1:,0:4],dtype=float)
y = np.array(D[1:,4],dtype=float)

X = preprocessing.scale(X)
X, Xte, y, yte = cross_validation.train_test_split(X, y, test_size=0.3)
Xtr, Xva, ytr, yva = cross_validation.train_test_split(X, y, test_size=0.3)

## PART A - training and validating the SVMs
G = np.arange(10.)*.1 + .1
poly_TE = []
for gamma in G:
    clf = svm.SVC(kernel="poly",degree=3,gamma=gamma,C=1.)
    clf.fit(Xtr,ytr)
    yhat = clf.predict(Xva)
    poly_TE.append(np.sum(np.abs(yhat - yva)))

gamma_poly = G[np.argmin(poly_TE)]

G = np.arange(10.)*.2 + .1
rbf_TE = []
for gamma in G:
    clf = svm.SVC(kernel="rbf",gamma=gamma,C=3.)
    clf.fit(Xtr,ytr)
    yhat = clf.predict(Xva)
    rbf_TE.append(np.sum(np.abs(yhat - yva)))

gamma_rbf = G[np.argmin(rbf_TE)]

rbf = svm.SVC(kernel="rbf",gamma=gamma_rbf,C=3.)
poly = svm.SVC(kernel="poly",degree=3,gamma=gamma_poly,C=1.)
lin = svm.SVC(kernel='linear')
```

```
## PART B - validating knn
K = range(2,10)
knn_TE = []
for k in K:
    nbrs = neighbors.KNeighborsClassifier(n_neighbors=k)
    nbrs.fit(Xtr,ytr)
    yhat = nbrs.predict(Xva)
    knn_TE.append(np.sum(np.abs(yhat - yva)) - k/100.)

k_va = K[np.argmin(knn_TE)]
knn = neighbors.KNeighborsClassifier(n_neighbors=k_va)

## PART C - testing the methods
methods = [rbf,poly,lin,knn]
TE = []
for meth in methods:
    meth.fit(X,y)  #here I retrain on the whole training set
    yhat = meth.predict(Xte)
    TE.append(np.sum(np.abs(yhat - yte)))

print TE
```

The test errors for SVM-rbf, SVM-poly, SVM-linear, kNN are $43.0, 53.0, 50.0, 63.0$ respectively. The method seems to be sensitive to test and training set selection, but not choice of tuning parameters. These misclassification errors are nearly as large as the size of positively labelled class, which indicates that nothing is working particularly well.