CPSC 340 Assignment 5 (due Friday November 15 at 11:55pm)

Zijia Zhang 42252965

If you are planning to do a poster project, make sure to register for the poster session as soon as possible. Details regarding registration are here:
`https://piazza.com/class/k02x04b6o524?cid=358`

1 Kernel Trick

In this question you will revisit questions from previous assignments, this time implementing the same (or similar) models using the "kernel trick".

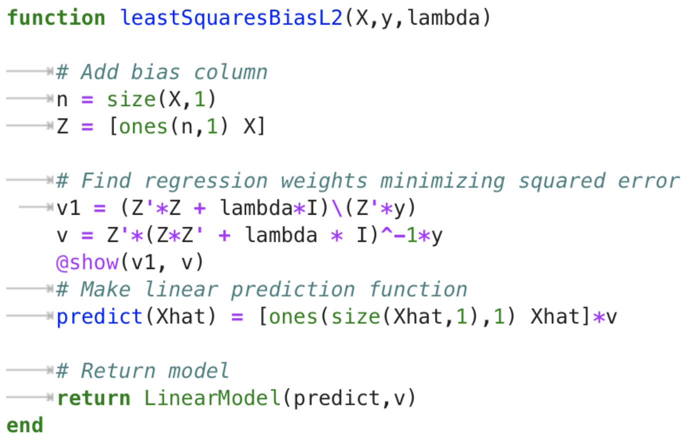
1.1 "Other" Normal Equations

The script *example_nonLinear* loads a dataset from a previous assignment, and fits an L2-regularized least squares model with a bias term (the regularization leads to small improvement in the test error, even for this 2-variable problem). Modify the *leastSquaresBiasL2* function so that it uses the "other" normal equations we discussed in class,

$$v = Z^T \underbrace{(ZZ^T + \lambda I)^{-1} y}_{u}.$$

In particular, the training should result in an $n \times 1$ vetor u of parameters, and predictions are made based on the training Z and the vector u. Hand in your code for the modified function.

Hint: you should get the same predictions as the original funciton. To help debugging, you may want to first re-write the calculation of v using the above formula, to verify that this gives you the same vector v.

Answer: 

```
function leastSquaresBiasL2(X,y,lambda)

    # Add bias column
    n = size(X,1)
    Z = [ones(n,1) X]

    # Find regression weights minimizing squared error
    v1 = (Z'*Z + lambda*I)\(Z'*y)
    v = Z'*(Z*Z' + lambda * I)^-1*y
    @show(v1, v)
    # Make linear prediction function
    predict(Xhat) = [ones(size(Xhat,1),1) Xhat]*v

    # Return model
    return LinearModel(predict,v)
end
```

1.2 Polynomial Kernel

Write a new function, *leastSquaresKernelBasis*, taking a degree p and a regularization parameter λ. It should fit a degree-p polynomial to the data, using the kernel trick to avoid ever forming the matrix Z. Hand in your code and the plot obtained with $p = 3$ and $\lambda = 10^{-6}$.

Hint: you may find it helpful to write a function *polyKernel* that takes two matrices as inputs (either X and X or $\tilde{X}$ and X) and a degree p, and computes the polynomial kernel between all pairs of rows in the matrices.

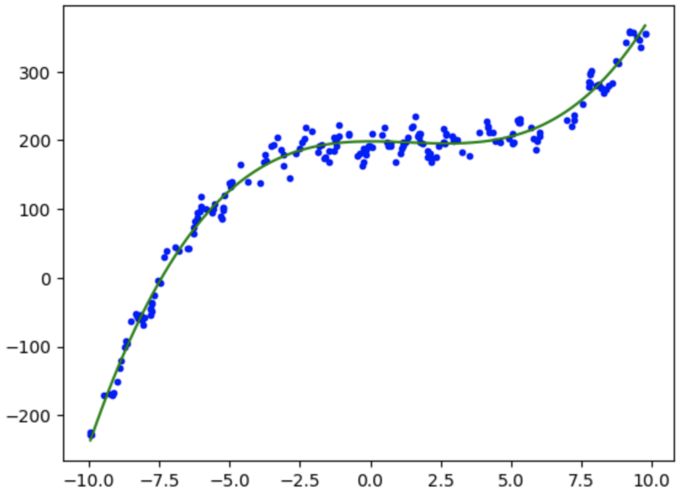
Answer:

```
function leastSquaresKernalBias(X,y,lambda, p)

    # Find regression weights minimizing squared error
    k = polyKernal(X,X, p)
    # Make linear prediction function
    predict(Xhat) = polyKernal(Xhat, X, p) * ((k + lambda * I) \ y)

    # Return model
    return GenericModel(predict)
end

function polyKernal(X1, X2, p)
    t = size(X1,1)
    n = size(X2,1)
    k = ones(t,n)
    for i = 1:t
        for j = 1:n
            k[i,j] = (1 + X1[i,:]'*X2[j,:])^p
        end
    end
    return k
end
```



```
Squared train Error with least squares: 252.017
Squared test Error with least squares: 242.791
```

1.3 Gaussian-RBF Kernel

Repeat the previous question and report the same quantities, but using the Gaussian RBF kernel. You can use $\sigma = 1$ and $\lambda = 10^{-6}$ for the plot.

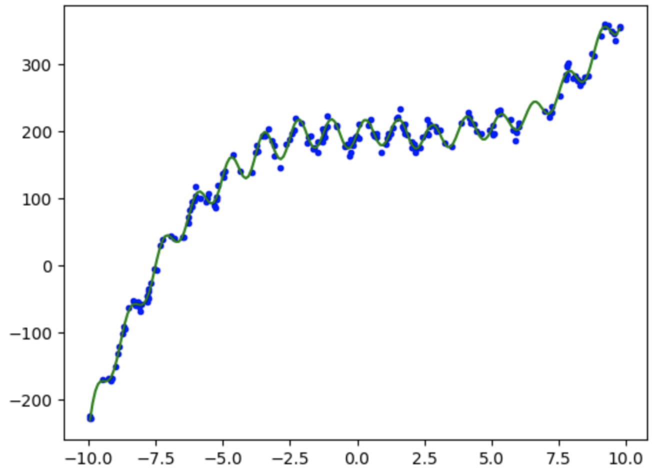
Answer:

```
function RBFKernalBias(X,y,lambda, sigma)

    # Find regression weights minimizing squared error
    k = RBFKernal(X,X, sigma)
    # Make linear prediction function
    predict(Xhat) = RBFKernal(Xhat, X, sigma) * ((k + lambda * I) \ y)

    # Return model
    return GenericModel(predict)
end

function RBFKernal(X1, X2, sigma)
    t = size(X1,1)
    n = size(X2,1)
    k = ones(t,n)
    for i = 1:t
        for j = 1:n
            k[i,j] = exp(- norm(X1[i,:] - X2[j,:])^2/(2*sigma^2))
        end
    end
    return k
end
```



```
Squared train Error with least squares: 39.163
Squared test Error with least squares: 70.580
1]: 1-element Array{PyCall.PyObject,1}:
     PyObject <matplotlib.lines.Line2D object at 0x14333eac8>
```

2 MAP Estimation

In class, we considered MAP estimation in a regression model where we assumed that:

- The likelihood $p(y_i \mid x_i, w)$ for each example i is a normal distribution with a mean of $w^T x_i$ and a variance of 1.
- The prior $p(w_j)$ for each variable j is a normal distribution with a mean of zero and a variance of λ^{-1}.

Under these assumptions we showed that computing the MAP estimate with n training examples leads to the standard L2-regularized least squares objective function:

$$f(w) = \frac{1}{2}\|Xw - y\|^2 + \frac{\lambda}{2}\|w\|^2.$$

For each of the alternate assumptions below, write down the objective function that results (from minimizing the negative log-posterior, and simplifying as much as possible):

1. We use a Laplace likelihood with a mean of $w^T x_i$ and a scale of 1, and we use a zero-mean Laplace

prior for each variable with a scale parameter of λ^{-1},

$$p(y_i \mid x_i, w) = \frac{1}{2}\exp(-|w^T x_i - y_i|), \quad p(w_j) = \frac{\lambda}{2}\exp(-\lambda|w_j|).$$

Answer:

$$\begin{aligned}
f(w) &= -\sum_{i=1}^{n}\log(p(y_i \mid x_i, w)) - \sum_{j=0}^{d}\log(p(w_j)) \\
&= -\sum_{i=1}^{n}\log(\frac{1}{2}\exp(-|w^T x_i - y_i|)) - \sum_{j=0}^{d}\log(\frac{\lambda}{2}\exp(-\lambda|w_j|)) \\
&= -\sum_{i=1}^{n}\log(\frac{1}{2}) - \sum_{i=1}^{n}\log(\exp(-|w^T x_i - y_i|)) - \sum_{j=0}^{d}\log(\frac{\lambda}{2}) - \sum_{j=0}^{d}\log(\exp(-\lambda|w_j|)) \\
&= contant - \sum_{i=1}^{n} -|w^T x_i - y_i| + \sum_{j=0}^{d}\lambda|w_j| \\
&= constant + \|w^T X - y\|_1 + \lambda\|w_j\|_1
\end{aligned}$$

2. We use a normal likelihood with a mean of $w^T x_i$ but where each example i has its own positive variance σ_i^2, and a normal prior with a variance of λ^{-1} and a mean that is some "guess" w^0 of the optimal parameter vector,

$$p(y_i \mid x_i, w) = \frac{1}{\sqrt{2\sigma_i^2\pi}}\exp\left(-\frac{(w^T x_i - y_i)^2}{2\sigma_i^2}\right), \quad p(w_j) \propto \exp\left(-\frac{\lambda(w_j - w_j^0)^2}{2}\right).$$

Answer:

$$\begin{aligned}
f(w) &= -\sum_{i=1}^{n}\log(p(y_i \mid x_i, w)) - \sum_{j=0}^{d}\log(p(w_j)) \\
&= -\sum_{i=1}^{n}\log(\frac{1}{\sqrt{2\sigma_i^2\pi}}\exp\left(-\frac{(w^T x_i - y_i)^2}{2\sigma_i^2}\right)) - \sum_{j=0}^{d}\log(\exp\left(-\frac{\lambda(w_j - w_j^0)^2}{2}\right)) \\
&= -\sum_{i=1}^{n}(\log(\frac{1}{\sqrt{2\sigma_i^2\pi}}) + \log(\exp\left(-\frac{(w^T x_i - y_i)^2}{2\sigma_i^2}\right))) + \sum_{j=0}^{d}\frac{\lambda(w_j - w_j^0)^2}{2} \\
&= -\sum_{i=1}^{n}(constant + \left(-\frac{(w^T x_i - y_i)^2}{2\sigma_i^2}\right))) + \lambda/2\|w - w^0\|_2 \\
&= constant + \|\Sigma^{-1}(w^T x - y)\|_2^2 + \lambda\|w - w^0\|_2
\end{aligned}$$

The standard notation for this case is to use Σ as a diagonal matrix with the σ_i^2 values along the diagonal.

3. We use a Poisson likelihood with a mean of $\exp(w^T x_i)$,[1] and we use a uniform prior for some constant κ,

$$p(y_i \mid x_i, w) = \frac{\exp(y_i w^T x_i)\exp(-\exp(w^T x_i))}{y_i!}, \quad p(w_j) \propto \kappa$$

[1]This is one way to use regression to model *counts*, like "number of Facebook likes".

Answer:

$$\begin{aligned}
f(w) &= -\sum_{i=1}^{n} \log(p(y_i \mid x_i, w)) - \sum_{j=0}^{d} \log(p(w_j)) \\
&= -\sum_{i=1}^{n} \log(\frac{\exp(y_i w^T x_i)\exp(-\exp(w^T x_i))}{y_i!}) - \sum_{j=0}^{d} \log(\kappa) \\
&= -\sum_{i=1}^{n} y_i w^T x_i + \exp(w^T x_i) + \log(y_i!) + constant \\
&= -\sum_{i=1}^{n} y_i w^T x_i + \exp(w^T x_i) + \log(y_i!)
\end{aligned}$$

For this sub-question you don't need to put likelihood in matrix notation.

4. We use a Laplace likelihood with a mean of $w^T x_i$ where each example i has its own positive scale paramater v_i^{-1}, and a student t prior (which is very robust to irrelevant features) with ν degrees of freedom,

$$p(y_i \mid x_i, w) = \frac{1}{2}\exp\left(-v_i|w^T x_i - y_i|\right), \quad p(w_j) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1 + \frac{w_j^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

where you use can V as a diagonal matrix with the v_i along the diagonal and Γ is the "gamma" function (which is always non-negative). You do not need to put the log-prior in matrix notation.

Answer:

$$\begin{aligned}
f(w) &= -\sum_{i=1}^{n} \log(p(y_i \mid x_i, w)) - \sum_{j=0}^{d} \log(p(w_j)) \\
&= -\sum_{i=1}^{n} \log(\frac{1}{2}\exp\left(-v_i|w^T x_i - y_i|\right)) - \sum_{j=0}^{d} \log(\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1 + \frac{w_j^2}{\nu}\right)^{-\frac{\nu+1}{2}}) \\
&= -\sum_{i=1}^{n} \log(1/2) + \log(\exp\left(-v_i|w^T x_i - y_i|\right)) - \sum_{j=0}^{d} \log(\frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}) - \sum_{j=0}^{d} \log(\left(1 + \frac{w_j^2}{\nu}\right)^{-\frac{\nu+1}{2}}) \\
&= constant - \sum_{i=1}^{n} \left(-v_i|w^T x_i - y_i|\right) - \sum_{j=0}^{d} \log(\left(1 + \frac{w_j^2}{\nu}\right)^{-\frac{\nu+1}{2}}) \\
&= constant + \|(w^T X - y)V\|_1 + \frac{\nu+1}{2}\sum_{j=0}^{d} \log\left(1 + \frac{w_j^2}{\nu}\right)
\end{aligned}$$

3 Principal Component Analysis

3.1 PCA by Hand

Consider the following dataset, containing 5 examples with 2 features each:

x_1	x_2
-4	3
0	1
-2	2
4	-1
2	0

Recall that with PCA we usually assume that the PCs are normalized ($\|w\| = 1$), that we need to center the data before we apply PCA, and that the direction of the first PC is the one that minimizes the orthogonal distance to all data points.

1. What is the first principal component?

   Answer: Center X: $(X_2 - 1)$

x_1	x_2
-4	2
0	0
-2	1
4	-2
2	-1

   Then we can see the line is $x_2 = -\frac{x_1}{2}$.

   We then can have $W = [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}]$

2. What is the (L2-norm) reconstruction error of the point (-3, 2.5)? (Show your work.)

   Answer: We first do the "center" that is $x_2 - 1$. We get data: (-3,1.5)
   $Z = [-3, 1.5] * [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}]'([\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}] * [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}]')^{-1} = \frac{7.5}{\sqrt{5}}$
   $\hat{X} = [0, 1] + \frac{7.5}{\sqrt{5}} * [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}] = [-3, 2.5]$
   So the reconstruction error is $\|\frac{7.5}{\sqrt{5}} * [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}] - [-3, 1.5]\|_F^2 = 0$.

3. What is the (L2-norm) reconstruction error of the point (-3, 2)? (Show your work.)

   Answer: We first do the "center". We get data: (-3,1) $Z = [-3, 1] * [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}]'([\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}] * [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}]')^{-1} = \frac{7}{\sqrt{5}}$
   Thus the reconstruction is $\hat{X} = [0, 1] + \frac{7}{\sqrt{5}} * [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}] = [-2.8, 2.4]$
   So the reconstruction error is $\|\frac{7}{\sqrt{5}} * [\frac{-2}{\sqrt{5}}, \frac{1}{\sqrt{5}}] - [-3, 1]\|_F^2 = \|[0.2, 0.4]\|_F^2 = 0.2$

3.2 Data Visualization

The script *example_PCA* will load a dataset containing 50 examples, each representing an animal. The 85 features are traits of these animals. The script standardizes these features and gives two unsatisfying visualizations of it. First it shows a plot of the matrix entries, which has too much information and thus gives little insight into the relationships between the animals. Next it shows a scatterplot based on two random features and displays the name of 10 randomly-chosen animals. Because of the binary features even a scatterplot matrix shows us almost nothing about the data.

The function *PCA* applies the classic PCA method (orthogonal bases via SVD) for a given k. Using this function, modify the demo so that the scatterplot uses the latent features z_i from the PCA model with $k = 2$. Make a scatterplot of the two columns in Z, and use the *annotate* function to label a bunch of the points in the scatterplot.

1. Hand in your modified demo and the scatterplot.

   Answer:

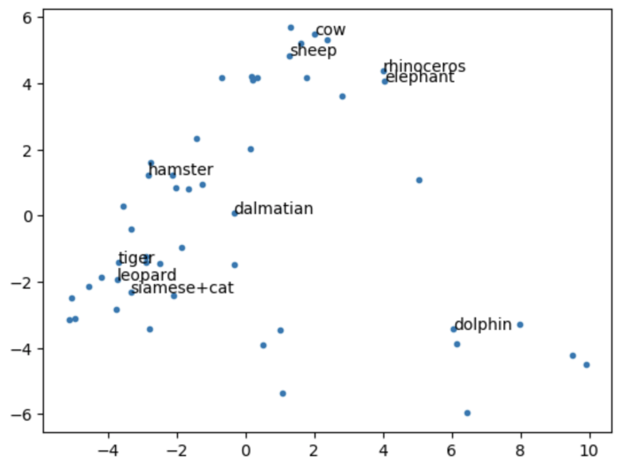
```julia
using DelimitedFiles

# Load data
dataTable = readdlm("animals.csv",',')
X = float(real(dataTable[2:end,2:end]))
(n,d) = size(X)

# Standardize columns
include("misc.jl")
(X,mu,sigma) = standardizeCols(X)

# Plot matrix as image
using PyPlot
figure(1)
clf()
imshow(X)

include("PCA.jl")
model = PCA(X,2)
compressed = model.compress(X);
# Show scatterplot of 2 random features
# j1 = rand(1:d)
# j2 = rand(1:d)
figure(2)
clf()
plot(compressed[:,1],compressed[:,2],".")
for i in rand(1:n,10)
    annotate(dataTable[i+1,1],
    xy=[compressed[i,1],compressed[i,2]],
    xycoords="data")
end
```



```
argmax(W[1, :]) = 13
argmax(W[2, :]) = 58
```

2. Which trait of the animals has the largest influence (absolute value) on the first principal component? (Make sure not to forget the "+1" when looking for the name of the trait in the *dataTable*).

   Answer: "hairless"

3. Which trait of the animals has the largest influence (absolute value) on the second principal component?

   Answer: "grazer"

3.3 Data Compression

It is important to know how much of the information in our dataset is captured by the low-dimensional PCA representation. In class we discussed the "analysis" view that PCA maximizes the variance that is explained by the PCs, and the connection between the Frobenius norm and the variance of a centered data matrix X. Use this connection to answer the following:

1. How much of the variance is explained by our two-dimensional representation from the previous question?

   Answer: $1 - \frac{\|ZW-X\|^2}{\|x\|^2} = 0.3019$

2. How many PCs are required to explain 50% of the variance in the data?

   Answer: 5

Note: you can compute the Frobenius norm of a matrix using the function *norm*. Also, note that the "variance explained" formula from class assumes that X is already centered.

4 Very-Short Answer Questions

1. What is the difference between multi-label and multi-class classification?

   Answer: In a multi-label classification, one object can have many labels, but in multi-class classification, one object can only have one class.

2. We discussed "global" vs. "local" features for e-mail classification. What is an advantage of using global features, and what is advantage of using local features?

   Answer: "Global Features are useful when we have an important email for all users. And Local features are useful for identifying emails for specific users.

3. Assuming we want to use the original features (no change of basis) in a linear model, what is an advantage of the "other" normal equations over the original normal equations?

   Answer: It is faster when $n < d$.

4. What is the key advantage of stochastic gradient methods over gradient descent methods?

   Answer: Solve the issue when "n" is really large and it will take a long time for gradiant decent to take one step.

5. Which of the following step-size sequences lead to convergence of stochastic gradient to a stationary point?

   (a) $\alpha^t = 1/t^2$.

   (b) $\alpha^t = 1/t$.

   (c) $\alpha^t = \gamma/t$ (for $\gamma > 0$).

   (d) $\alpha^t = 1/(t+\gamma)$ (for $\gamma > 0$).

   (e) $\alpha^t = 1/\sqrt{t}$.

   (f) $\alpha^t = 1$.

Answer: b,c,d

6. In the language of loss functions and regularization, what is the difference between MLE and MAP?

   Answer: MAP will give a score of the likelihood of w as perior, which is like the regularization, what gives a very samll value for the w that is likely to be overfitting.

7. What is the difference between a generative model and a discriminative model?

   Answer: generative model : Often need strong assumption as it model "X", and discriminative model: can use complicated features as it don't model "X".

8. In the MLE framework, what is the connection between the logistic loss and the sigmoid function in linear models.

   Answer: If we use sigmoid function, we will get logistic loss after applying the MLE.

9. What is the significance of choosing $k = 2$ for visualizing with PCA.

   Answer: We can plot it using the 2 "features" to get a better visualization, than 2 original features.

10. With PCA, is it possible for the loss to increase if k is increased? Briefly justify your answer.

    Answer: No. Because you can always set Z value to 0 to the k that is more than the previous one and get the same loss.

11. Why doesn't it make sense to do PCA with $k > d$?

    Answer: Because you will always get a perfect result because the dimention of the hyper plane is more than the dimention of the original data. Doesn't make sence. And if k ¿ d the "compression" effect does not occur.

12. In terms of the matrices associated with PCA (X, W, Z, $\hat{X}$), where would an "eigenface" be stored?

    Answer: W

Project Update (OPTIONAL for 340 STUDENTS)

Alongside Assignment 5, we are also asking you to submit a *project update*. It's ok if you haven't yet made much progress on your project, but we're making you submit this document as an excuse to get together with your project and plan. In particular, the project update should be a 1-page report covering the following:

1. **What has been done already**: for example, you might have already found a group of appropriate size, picked a topic, got access to appropriate data, read some related papers, or maybe even ran some preliminary experiments.

2. **What still needs to be**: what are the "action items" that need to be finished in order to complete the project. The purpose here is that you make a plan to finishing the different things that will need to come together in order to finish your project.

It's ok if the report is fairly short, and just written as series of bullet items addressing the two issues above.