DSC 140A - Homework 04

Due: Wednesday, February 7

Write your solutions to the following problems by either typing them up or handwriting them on another piece of paper. Unless otherwise noted by the problem's instructions, show your work or provide some justification for your answer. Homeworks are due via Gradescope at 11:59 PM.

Problem 1.

In this problem, you will derive the solution to the ridge regression optimization problem.

Recall that the ridge regression regularized risk function is

$$\tilde{R}(\vec{w}) = \frac{1}{n}\sum_{i=1}^{n}(\vec{\phi}(\vec{x}^{(i)}) \cdot \vec{w} - y_i)^2 + \lambda\|\vec{w}\|^2.$$

Here, $\vec{\phi}(\vec{x})$ is a feature map. Also recall that this risk function can be equivalently written in matrix-vector form as

$$\tilde{R}(\vec{w}) = \frac{1}{n}\|\Phi\vec{w} - \vec{y}\|^2 + \lambda\|\vec{w}\|^2,$$

where Φ is the design matrix; its ith row is $\vec{\phi}(\vec{x}^{(i)})$.

a) Show that $\tilde{R}(\vec{w}) = \frac{1}{n}\left(\vec{w}^T\Phi^T\Phi\vec{w} - 2\vec{w}^T\Phi^T\vec{y} + \vec{y}^T\vec{y}\right) + \lambda\vec{w}^T\vec{w}$.

Solution: For $\|\vec{w}\|^2 = \vec{w}^T\vec{w}$, we know that the norm $\|\vec{w}\|$ is defined as follows,

$$\|\vec{w}\| = \sqrt{\vec{w}_1^2 + \vec{w}_2^2 + \cdots + \vec{w}_n^2}$$

so that $\|\vec{w}\|^2 = \vec{w}_1^2 + \vec{w}_2^2 + \cdots + \vec{w}_n^2$. Which is equivalent to $\vec{w}^T\vec{w}$, as

$$\vec{w}^T\vec{w} = \begin{bmatrix}\vec{w}_1 & \vec{w}_2 & \cdots & \vec{w}_n\end{bmatrix}\begin{bmatrix}\vec{w}_1 \\ \vec{w}_2 \\ \vdots \\ \vec{w}_n\end{bmatrix} = \vec{w}_1^2 + \vec{w}_2^2 + \cdots + \vec{w}_n^2$$

So now looking at $\|\Phi\vec{w} - \vec{y}\|^2$, we can expand this as follows,

$$\|\Phi\vec{w} - \vec{y}\|^2 = (\Phi\vec{w} - \vec{y})^T(\Phi\vec{w} - \vec{y})$$

Using the properties of the transpose, we can expand this as follows,

$$(\Phi\vec{w} - \vec{y})^T(\Phi\vec{w} - \vec{y}) = (\vec{w}^T\Phi^T - \vec{y}^T)(\Phi\vec{w} - \vec{y})$$

Then we can expand this as follows,

$$(\vec{w}^T\Phi^T - \vec{y}^T)(\Phi\vec{w} - \vec{y}) = \vec{w}^T\Phi^T\Phi\vec{w} - \vec{w}^T\Phi^T\vec{y} - \vec{y}^T\Phi\vec{w} + \vec{y}^T\vec{y}$$

Thus this yields

$$\frac{1}{n}\|\Phi\vec{w} - \vec{y}\|^2 = \frac{1}{n}\left(\vec{w}^T\Phi^T\Phi\vec{w} - \vec{w}^T\Phi^T\vec{y} - \vec{y}^T\Phi\vec{w} + \vec{y}^T\vec{y}\right)$$

We can also see that by the properties of transposes, $\vec{y}^T\Phi\vec{w} = \vec{w}^T\Phi^T\vec{y}$, so that we can simplify this to

$$\frac{1}{n}\|\Phi\vec{w} - \vec{y}\|^2 = \frac{1}{n}\left(\vec{w}^T\Phi^T\Phi\vec{w} - 2\vec{w}^T\Phi^T\vec{y} + \vec{y}^T\vec{y}\right)$$

Thus we have shown that

$$\tilde{R}(\vec{w}) = \frac{1}{n}\left(\vec{w}^T\Phi^T\Phi\vec{w} - 2\vec{w}^T\Phi^T\vec{y} + \vec{y}^T\vec{y}\right) + \lambda\vec{w}^T\vec{w}$$

b) So far this quarter, we have seen a few vector calculus identities. For example, we know that $\frac{d}{d\vec{w}}(\vec{w}^T\vec{w}) = 2\vec{w}$.

Using these identities, show that

$$\frac{d}{d\vec{w}}\tilde{R}(\vec{w}) = \frac{1}{n}\left(2\Phi^T\Phi\vec{w} - 2\Phi^T\vec{y}\right) + 2\lambda\vec{w}.$$

Solution: For $\lambda\vec{w}^T\vec{w}$ it is clear that $\frac{d}{d\vec{w}}\lambda\vec{w}^T\vec{w} = 2\lambda\vec{w}$. For $\frac{d}{d\vec{w}}\vec{w}^T\Phi^T\Phi\vec{w}$, for the first term, we can let $\Phi^T\Phi = A$, so that we have $\frac{d}{d\vec{w}}\vec{w}^TA\vec{w}$. Then by the product rule, we have

$$\frac{d}{d\vec{w}}\vec{w}^TA\vec{w} = 2A\vec{w}$$

This is because teh matrix A is symmetric, so that $A^T = A$ which is true as $\Phi^T\Phi = \Phi\Phi^T$. Thus we have shown the first term is $2A\vec{w}$. For the second term, we look at $\frac{d}{d\vec{w}}2\vec{w}^T\Phi^T\vec{y}$, which is simply $2\Phi^T\vec{y}$. Thus we have shown that

$$\frac{d}{d\vec{w}}\tilde{R}(\vec{w}) = \frac{1}{n}\left(2\Phi^T\Phi\vec{w} - 2\Phi^T\vec{y}\right) + 2\lambda\vec{w}$$

c) Show that the minimizer of $\tilde{R}(\vec{w})$ is $\vec{w}^* = (\Phi^T\Phi + n\lambda I)^{-1}\Phi^T\vec{y}$.

Solution: So to minimize the equation $\tilde{R}(\vec{w})$, we take the derivative and set it equal to zero. So we have

$$\frac{d}{d\vec{w}}\tilde{R}(\vec{w}) = \frac{1}{n}\left(2\Phi^T\Phi\vec{w} - 2\Phi^T\vec{y}\right) + 2\lambda\vec{w} = 0$$

Then we can solve for $\vec{w}$ as follows,

$$\begin{aligned}
&\frac{1}{n}(2\Phi^T\Phi\vec{w} - 2\Phi^T\vec{y}) + 2\lambda\vec{w} = 0 \\
\implies\ &\Phi^T\Phi\vec{w} - \Phi^T\vec{y} + n\lambda\vec{w} = 0 \\
\implies\ &\vec{w}(\Phi^T\Phi + n\lambda I) = \Phi^T\vec{y} \\
\implies\ &\vec{w} = (\Phi^T\Phi + n\lambda I)^{-1}\Phi^T\vec{y}
\end{aligned}$$

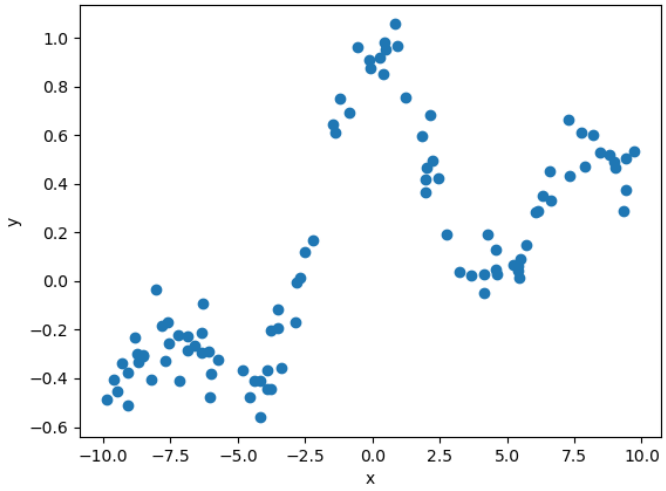
Thus we have shown that the minimizer of $\tilde{R}(\vec{w})$ is $\vec{w}^* = (\Phi^T\Phi + n\lambda I)^{-1}\Phi^T\vec{y}$.

Problem 2.

The data set linked below contains data for performing non-linear regression. The first column is x (the independent variable), and the second column is y (the dependent variable).

`https://f000.backblazeb2.com/file/jeldridge-data/010-nonlinear_regression/data.csv`

Plotting the data shows that there is a non-linear relationship between x and y:



a) Fit a function of the form $H(\vec{x}) = w_1\phi_1(\vec{x}) + w_2\phi_2(\vec{x}) + \ldots + w_{50}\phi_{50}(\vec{x})$, where each $\phi_i(\vec{x})$ is a Gaussian basis function. Your 50 Gaussian basis functions should be equally-spaced, with the first at $\mu_1 = -10$ and the last at $\mu_{50} = 10$. The width of each Gaussian should be $\sigma = 1$. You should *not* augment $\vec{x}$.

For your answer, report only w_1 (the first component of $\vec{w}$), and show your code.

Code

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression


data = np.loadtxt('data.csv', delimiter=',')
data = data[data[:, 0].argsort()]

x = data[:, 0]
y = data[:, 1]


mus = np.linspace(-10, 10, 50)
sigma = 1

gaussians = []
for mu in mus:
    gaussians.append(np.exp(-0.5 * (x - mu)**2 / sigma**2))


np.random.seed(0) # set the random seed for reproducibility

model = LinearRegression()
model.fit(np.array(gaussians).T, y)


print(model.coef_[0])
```

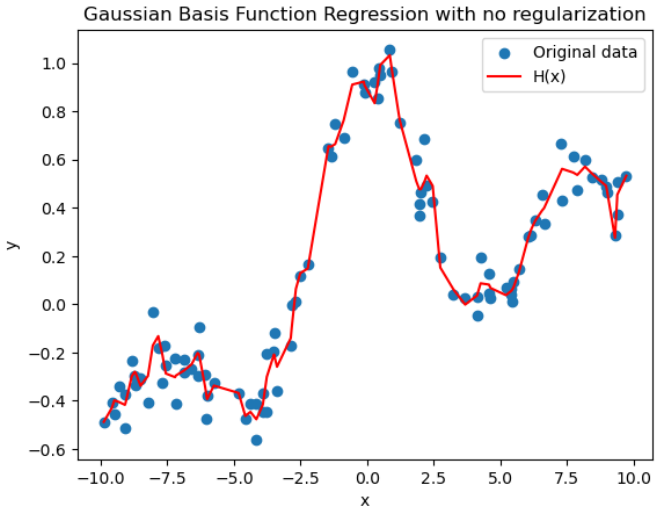
Solution: The value of w_1 is 1390758.6881607738.

b) Plot the prediction function $H(\vec{x})$ that you trained in the previous part, on top of the data. Provide your plot and the code used to generate it.

Code

```
# Calculate H(x) using the trained model
hx = model.predict(np.array(gaussians).T)

# Plot the original data
plt.scatter(x, y)

# Plot H(x) on top of the original data
plt.plot(x, hx, color='red')


plt.xlabel('x')
plt.ylabel('y')
plt.title('Gaussian Basis Function Regression with no regularization')
plt.legend(['Original data', 'H(x)'])
plt.show()
```

Solution: The plot is



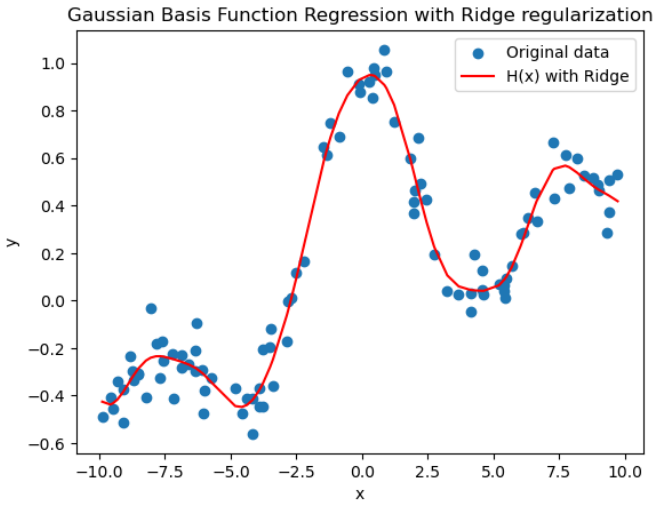
c) You should see that the prediction function $H(\vec{x})$ slighly overfits the data. Now perform ridge regression on the same data, using the same Gaussian basis functions. Choose the regularization parameter λ to reduce overfitting (you may do so by trial and error – no need to perform cross-validation). For your answer, state λ and plot your new prediction function on top of the data. Also provide your code.

Code

```
from sklearn.linear_model import Ridge
```

```
np.random.seed(0)
model = Ridge(alpha=0.1)
model.fit(np.array(gaussians).T, y)

hx_ridge = model.predict(np.array(gaussians).T)

# Plot the original data
plt.scatter(x, y)

# Plot H(x) using Ridge on top of the original data
plt.plot(x, hx_ridge, color='red')

# Show the plot
plt.xlabel('x')
plt.ylabel('y')
plt.title('Gaussian Basis Function Regression with Ridge regularization')
plt.legend(['Original data', 'H(x) with Ridge'])
plt.show()
```

Solution: The plot is



using a λ of 0.1.