DSC 140A - Homework 04

Due: Wednesday, February 8

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 p.m.

Note! Because the midterm is on Thursday, we would like to release the solutions to this homework immediately after the due date on Wednesday at midnight. Since this is before the late due date, we **cannot accept slip days for this homework**.

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: Using the fact that, for any vector $\vec{u}$, $\|vecu\|^2 = \vec{u}^T\vec{u}$, we have:

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

$\vec{y}^T\Phi\vec{w}$ is a scalar, and any scalar transposed is just itself. So $(y^T\Phi\vec{w})^T = \vec{w}^T\Phi^T\vec{y}$, and we have:

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

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:

$$\begin{aligned}
&\frac{d}{d\vec{w}}\left[\frac{1}{n}\left(\vec{w}^T\Phi^T\Phi\vec{w} - 2\vec{w}^T\Phi^T\vec{y} - \vec{y}^T\vec{y}\right) + \lambda\vec{w}^T\vec{w}\right] \\
&\qquad = \left[\frac{1}{n}\left(\frac{d}{d\vec{w}}\vec{w}^T\Phi^T\Phi\vec{w} - \frac{d}{d\vec{w}}2\vec{w}^T\Phi^T\vec{y} - \frac{d}{d\vec{w}}\vec{y}^T\vec{y}\right) + \frac{d}{d\vec{w}}\lambda\vec{w}^T\vec{w}\right] \\
&\qquad = \frac{1}{n}\left(2\Phi^T\Phi\vec{w} - \Phi^T\vec{y} - \vec{0}\right) + 2\lambda\vec{w}
\end{aligned}$$

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

Note: the lecture slides had originally stated the solution without the n in $n\lambda I$. This was a typo and has been fixed.

Solution: Setting the gradient to sero and solving for $\vec{w}$, we have:

$$\begin{aligned}
&\frac{1}{n}\left(2\Phi^T\Phi\vec{w} - 2\Phi^T\vec{y}\right) + 2\lambda\vec{w} = \vec{0} && \text{(multiply both sides by } n\text{)} \\
&\Rightarrow 2\Phi^T\Phi\vec{w} - 2\Phi^T\vec{y} + 2\lambda n\vec{w} = \vec{0} && \text{(grouping terms with } \vec{w}\text{)} \\
&\Rightarrow 2\Phi^T\Phi\vec{w} + 2\lambda n\vec{w} = 2\Phi^T\vec{y} && \text{(dividing both sides by 2)} \\
&\Rightarrow \Phi^T\Phi\vec{w} + \lambda n\vec{w} = \Phi^T\vec{y} && \text{(factoring out } \vec{w}\text{)} \\
&\Rightarrow (\Phi^T\Phi + n\lambda I)\vec{w} = \Phi^T\vec{y} && \text{(solving for } \vec{w}\text{)} \\
&\Rightarrow \vec{w} = (\Phi^T\Phi + n\lambda I)^{-1}\Phi^T\vec{y}
\end{aligned}$$

Problem 2.

In this problem, you will demonstrate that kernel ridge regression is equivalent to ridge regression performed in feature space by showing that they make the same predictions on a concrete data set.

We'll use the toy data set:

i	$\vec{x}^{(i)}$	y_i
1	$(0, 2)^T$	1
2	$(1, 0)^T$	1
3	$(0, -2)^T$	1
4	$(-1, 0)^T$	1
5	$(0, 0)^T$	-1

We will define

$$\vec{\phi}(\vec{x}) = (1, x_1^2, x_2^2, \sqrt{2}\,x_1, \sqrt{2}\,x_2, \sqrt{2}\,x_1x_2)^T.$$

It turns out that $\kappa(\vec{x}, \vec{x}') = (1 + \vec{x}\cdot\vec{x}')^2$ is a kernel for $\vec{\phi}$.

You can write code to perform any and all calculations in this problem. If you do, please show your code.

a) Learn a prediction rule $H_1(\vec{x})$ by performing ridge regression in the 6-dimensional feature space and report the optimal parameter vector $\vec{w}$. Use a regularization parameter of $\lambda = 2$.

Solution:

To perform ridge regression in feature space, we compute $\vec{w}^* = (\Phi^T\Phi + n\lambda I)^{-1}\Phi^T\vec{y}$, where Φ is the design matrix whose ith row is $\vec{\phi}(\vec{x})$.

The below code does exactly that:

```
import numpy as np

def phi(x):
    x_1, x_2 = x
    c = np.sqrt(2)
    return np.array([
        1,
        x_1**2,
        x_2**2,
        c * x_1,
        c * x_2,
        c * x_1 * x_2
    ])

X = np.array([
    [0, 2],
    [1, 0],
    [0, -2],
    [-1, 0],
    [0, 0]
])

y = np.array([1, 1, 1, 1, -1])

n = len(X)

ell = 2
Phi = np.array([phi(x) for x in X])
w = np.linalg.inv((Phi.T @ Phi + n * ell * np.eye(Phi.shape[1]))) @ Phi.T @ y
```

We find:

$$\vec{w} \approx (0.086,\ 0.152,\ 0.174,\ 0,\ 0,\ 0)^T$$

b) Given a new point, $\vec{x} = (1, 1)^T$, what is the prediction made by your ridge regressor? That is, what is $H_1(\vec{x})$? Show your calculations / code.

Solution: To predict $H_1(\vec{x})$, we need to map $\vec{x}$ to feature space with $\vec{\phi}(\vec{x})$ and dot it with $\vec{w}$. That is, we need to compute $\vec{phi}(\vec{x}, \vec{w})$.

To do so in code, we first define `x = np.array([1, 1])`, and then compute `phi(x) @ w`. Our answer is approximately 0.413.

c) Compute the kernel matrix, K, for this data.

Solution: We define a python function for computing the kernel and apply it to every pair of training points:

```
def k(x, z):
    return (1 + x @ z)**2

K = np.zeros((n, n))
for i in range(n):
    for j in range(i, n):
        K[i,j] = K[j,i] = k(X[i], X[j])his is done with the code

>>> K
array([[25.,  1.,  9.,  1.,  1.],
       [ 1.,  4.,  1.,  0.,  1.],
       [ 9.,  1., 25.,  1.,  1.],
       [ 1.,  0.,  1.,  4.,  1.],
       [ 1.,  1.,  1.,  1.,  1.]])
```

d) Learn a prediction function $H_2(\vec{x})$ by solving the kernel ridge regression dual problem; that is, by finding the optimal vector $\vec{\alpha}$. Recall that there is an exact solution: $\vec{\alpha} = (K + n\lambda I)^{-1}\vec{y}$. Report the $\vec{\alpha}$ that you find.

Note: the lecture slides had originally stated the solution without the n in $n\lambda I$. This was a typo and has been fixed.

Solution: `alpha = np.linalg.inv(K + n * ell * np.eye(n)) @ y`

We find $\vec{\alpha} \approx (0.022, 0.077, 0.022, 0.077, -0.11)^T$

e) Let $\vec{x} = (1, 1)^T$, as before. What does your kernel ridge regressor predict for this point? That is, what is $H_2(\vec{x})$?

Hint: $H_2(\vec{x})$ should be the same as $H_1(\vec{x})$.

Solution: To compute $H_2(\vec{x})$, we use the dual solution $\vec{\alpha}$. From lecture, we have:

$$H_2(\vec{x}) = \sum_{i=1}^{n} \alpha_i k(\vec{x}^{(i)}, x)$$

In code: `[k(x_i, x) for x_i in X] @ alpha`

We get 0.41, which is the same as $H_1(\vec{x})$, as expected.