Machine Learning HW2

Bohan Wu

February 2020

1 Problem1

1.1 (a)

Target variable: the total profit made by certain product.

1.2 (b)

The features contain:

- The average of numeric scores
- The average number of ”good” exists per text review
- The average number of ”bad” exists per text review
- The average number of ”broke” exists per text review
- The average number of ”bargain” exists per text review
- The possibility of the same user reorder the same product

1.3 (c)

We can normalize the numeric scores feature into the scale of 0 to 1. For example, if a product get scored 3 out of 5, the new score will be 0.6; if a product get scored 9 out of 10, the new score will be 0.9

1.4 (d)

We can add the same precondition to the related features such that whether the review is a valid review. The valid review has to contain both a numeric score and valid text review which for instance has at least 5 words. For each related feature, for example the numeric scores, if user didn't provide the score at all, we can set the value equal to NULL. Therefore, we can simply drop those 'NULLs' while doing data training in order to prevent from unnecessary influences caused by those invalid reviews.

2 Problem2

2.1 (a)

Mean centering may improve the performance of our linear model. But it will only help for some cases, for example, we have the issues of multicollinearity when the interaction terms are included. Different predictors may have effects on the others. Therefore, by using mean centering would help us to improve our model.

2.2 (b)

By normalizing each columns to have unit standard deviation could be really improve our linear model, if we use different units to represent our features. Because by using this method will help us to remove the unit of measure and rescale the variable to standard deviation. Therefore, I believe normalize my column data to have unit standard deviation for all cases will improve the model.

2.3 (c)

If we are fitting the model with ridge regulation, then as for the part(a) I will change the answer to use mean centering method anyway. And keep using unit standard deviation method for each column. Because as we learned in lecture 5, Ridge regulation method is not invariant to data scaling. Typically when using ridge regulation we mean center and scale columns to have unit variance.

3 Problem3

3.1 (a)

$$\frac{\partial}{\partial z_i}(\|z\|_p^p) = pz_i^{p-1} \tag{1}$$

Based on the equation1 we get, we can conclude:

$$\nabla g(z) = pz^{**p-1} \tag{2}$$

$**p-1$ in here, means we raise p-1 power of each value in matrix z rather than power of the whole matrix.

3.2 (b)

Let's assume $r(w) = \|w\|_p^p$, therefore, $r(y - X\beta) = \|y - X\beta\|_p^p$.
By applying chain rule, we can get $\nabla L_p(\beta) = -X^T \nabla r(y - X\beta)$. Based on what we get on part a, we can conclude the final answer will be:

$$\nabla L_p(\beta) = -X^T * p(y - X\beta)^{**p-1} \tag{3}$$

4 Problem4

4.1 (a)

When $x_i = \lambda$, the first portion of our $f(x_i)$ becomes $a_1 + s_1\lambda$, the second portion becomes $a_1 + s_1\lambda - s_2\lambda + s_2\lambda$. So we can see in second portion $s_2\lambda$ cancel out and the rest part is exactly same as the first portion. Therefore, this unconstrained model also ensures that two linear models actually 'meet' at $x = \lambda$ and equivalent to the given constrained model.

4.2 (b)

1) Data transformation:

- The first column will be a column of 1s
- The second column will be the list $l1$ containing if $x_i < \lambda$, $l1_i will be x_i$, otherwise $l1_i = \lambda$
- The third column will be the list $l2$ containing if $x_i < \lambda$, $l2_i will be 0$, otherwise $l2_i = x_i - \lambda$

2) There will be three value in matrix β. $\beta 0, \beta 1, and \beta 2$.

- $\beta 0$ is mapping to $a0$
- $\beta 1$ is mapping to s_1
- $\beta 2$ is mapping to s_2

4.3 (c)

```
x = df2['horsepower'].values
y = df2['mpg'].values
secondList = []
for item in x:
    if item < 100:
        secondList.append(item)
    else:
        secondList.append(100)

thirdList = []
for item in x:
    if item < 100:
        thirdList.append(0)
    else:
```

```
        thirdList.append(item-100)

ones = np.ones((392,1))
X0 = np.column_stack((ones,secondList))
X = np.column_stack((X0,thirdList))

Xt = np.transpose(X)
beta=np.dot(np.linalg.inv(np.dot(Xt,X)),np.dot(Xt,y))
print(beta)
x1 = x[x< 100]
yp = beta[0] + beta[1]*x1

x2 = x[x>= 100]
yp2 = beta[0] + beta[1]*100 - beta[2]*100 + beta[2]*x2

plt.plot(x,y,'o')
plt.plot(x1,yp,'-')
plt.plot(x2,yp2,'-')
plt.xlabel('horsepower')
plt.ylabel('mpg')
plt.grid(True)
```

```
[53.57724087 -0.32638817 -0.09142217]
```