CSE 512 Machine Learning

HW #3

Enbo Yu

1130947114

12 Oct (3 days extension)

1. Gradient properties.

(a) Linearity. If $h(x)=\alpha f(x)+\beta g(x)$, then $\nabla h(x)=\alpha\nabla f(x)+\beta\nabla g(x)$.

$h(x)=\alpha f(x)+\beta g(x)$,

we know that $\nabla f(x)$ is the gradient of f at x.

so for $\nabla h(x)$:

$$\nabla h(x)=\begin{bmatrix}\frac{\partial}{\partial x_1}h(x_1)\\ \frac{\partial}{\partial x_2}h(x_2)\\ \cdots\ \cdots\\ \frac{\partial}{\partial x_n}h(x_n)\end{bmatrix}=\begin{bmatrix}\frac{\partial}{\partial x_1}(\alpha f(x_1)+\beta g(x_1))\\ \frac{\partial}{\partial x_2}(\alpha f(x_2)+\beta g(x_2))\\ \cdots\ \cdots\\ \frac{\partial}{\partial x_n}(\alpha f(x_n)+\beta g(x_n))\end{bmatrix}$$

by the definition of Linearity of differentiation.

$$\frac{d}{dx}(\alpha\cdot f(x)+\beta\cdot g(x))=\alpha\cdot f'(x)+\beta\cdot g(x).$$

$$\text{so }\nabla h(x)=\begin{bmatrix}\alpha\cdot\frac{\partial}{\partial x_1}f(x_1)+\beta\frac{\partial}{\partial x_1}g(x_1)\\ \alpha\cdot\frac{\partial}{\partial x_2}f(x_2)+\beta\frac{\partial}{\partial x_2}g(x_2)\\ \cdots\ \cdots\\ \alpha\frac{\partial}{\partial x_n}f(x_n)+\beta\frac{\partial}{\partial x_n}g(x_n)\end{bmatrix}=\begin{bmatrix}\alpha\cdot\frac{\partial}{\partial x_1}f(x_1)\\ \alpha\cdot\frac{\partial}{\partial x_2}f(x_2)\\ \cdots\\ \alpha\frac{\partial}{\partial x_n}f(x_n)\end{bmatrix}+\begin{bmatrix}\beta\cdot\frac{\partial}{\partial x_1}g(x_1)\\ \beta\cdot\frac{\partial}{\partial x_2}g(x_2)\\ \cdots\\ \beta\frac{\partial}{\partial x_n}g(x_n)\end{bmatrix}$$

$$=\alpha\cdot\begin{bmatrix}\frac{\partial}{\partial x_1}f(x_1)\\ \frac{\partial}{\partial x_2}f(x_2)\\ \cdots\\ \frac{\partial}{\partial x_n}f(x_n)\end{bmatrix}+\beta\cdot\begin{bmatrix}\frac{\partial}{\partial x_1}g(x_1)\\ \frac{\partial}{\partial x_2}g(x_2)\\ \cdots\\ \frac{\partial}{\partial x_n}g(x_n)\end{bmatrix}=\alpha\cdot\nabla f(x)+\beta\cdot\nabla g(x)$$

so $\nabla h(x)=\alpha\nabla f(x)+\beta\nabla g(x)$.

(b) Chain rule. If $g(v) = f(Av)$, then $\nabla g(v) = A^T \nabla f(Av)$.

Based on question (a).

$$\nabla g(v) = \begin{bmatrix} \frac{\partial}{\partial v_1} g(v_1) \\ \frac{\partial}{\partial v_2} g(v_2) \\ \cdots \\ \frac{\partial}{\partial v_n} g(v_n) \end{bmatrix} \overset{\text{by chain rule}}{\underset{\frac{dy}{dx} = f'(g(x)) \cdot g'(x)}{=\!=\!=}} \begin{bmatrix} \frac{\partial}{\partial (Av_1)} f(Av_1) \cdot \frac{\partial}{\partial v_1}(Av_1) \\ \frac{\partial}{\partial (Av_2)} f(Av_2) \cdot \frac{\partial}{\partial v_2}(Av_2) \\ \cdots \\ \frac{\partial}{\partial (Av_n)} f(Av_n) \cdot \frac{\partial}{\partial v_n}(Av_n) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\partial}{\partial (Av_1)} f(Av_1) \cdot A_{\cdot 1} \\ \frac{\partial}{\partial (Av_2)} f(Av_1) A_2 \\ \cdots \\ \frac{\partial}{\partial (Av_n)} f(Av_n) \cdot A_n \end{bmatrix} = [A_1 A_2 \ldots A_n] \begin{bmatrix} \frac{\partial}{\partial (Av_1)} f(Av_1) \\ \frac{\partial}{\partial (Av_2)} f(Av_2) \\ \cdots \\ \frac{\partial}{\partial (Av_n)} f(Av_n) \end{bmatrix} = A^T \nabla f(Av).$$

so we can get $\nabla g(v) = A^T \nabla f(Av)$.

2. Gradients.

(a) Quadratic function. $f(x) = \frac{1}{2} x^T Q x + p^T x + r$, $Q \in R^{12 \times 12}$ and Q is symmetric ($Q_{[i,j]} = Q_{[j,i]}$)

$$\nabla f(x) = \begin{bmatrix} \frac{\partial}{\partial x_1}(\frac{1}{2} x_1^T Q x_1 + p^T x_1 + r) \\ \frac{\partial}{\partial x_2}(\frac{1}{2} x_2^T Q x_2 + p^T x_2 + r) \\ \cdots \cdots \\ \frac{\partial}{\partial x_n}(\frac{1}{2} x_n^T Q x_n + p^T x_n + r) \end{bmatrix}, \text{ by the linearity in Question 1 (a), we can get:}$$

$$= \begin{bmatrix} \frac{\partial}{\partial x_1}(\frac{1}{2} x_1^T Q x_1) \\ \frac{\partial}{\partial x_2}(\frac{1}{2} x_2^T Q x_2) \\ \cdots \\ \frac{\partial}{\partial x_n}(\frac{1}{2} x_n^T Q x_n) \end{bmatrix} + \begin{bmatrix} \frac{\partial}{\partial x_1}(p^T x_1) \\ \frac{\partial}{\partial x_2}(p^T x_2) \\ \cdots \\ \frac{\partial}{\partial x_n}(p^T x_n) \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{\partial}{\partial x_1} r \\ \frac{\partial}{\partial x_2} r \\ \cdots \\ \frac{\partial}{\partial x_n} r \end{bmatrix}}_{\hookrightarrow \frac{\partial}{\partial x}(\text{constant}) = 0} = \frac{1}{2} \begin{bmatrix} \frac{\partial}{\partial x_1}(x_1^T Q x_1) \\ \frac{\partial}{\partial x_2}(x_2^T Q x_2) \\ \cdots \\ \frac{\partial}{\partial x_n}(x_n^T Q x_n) \end{bmatrix} + \underbrace{\begin{bmatrix} \frac{\partial}{\partial x_1} p^T x_1 \\ \frac{\partial}{\partial x_2} p^T x_2 \\ \cdots \\ \frac{\partial}{\partial x_n} p^T x_n \end{bmatrix}}_{\Downarrow \ \frac{\partial}{\partial x}(px) = p}$$

$Q \in R^{12 \times 12}$, and $Q_{[i][j]} = Q_{[j][i]}$

$$Q = \begin{bmatrix} Q_{[1][1]} & \cdots & Q_{[1][12]} \\ \cdots & \cdots & \cdots \\ Q_{[12][1]} & \cdots & Q_{[12][12]} \end{bmatrix}, \quad \frac{\partial}{\partial \underline{x}} \underline{x}^T Q \underline{x} = \sum^{12} x_i Q_{[i][j]} + \sum^{12} x_j Q_{[j][i]}$$

$$\text{so } \nabla f(x) = \frac{1}{2} \cdot (Qx + Qx) + p + 0$$

$$= Qx + p$$

the dimension is 12.

(b) $f(x) = \frac{1}{\mu}\log\left(\sum_{i=1}^{8}\exp(\mu x_{[i]})\right)$. $x \in R^8$. μ is a positive scalar.

let $g(x) = \sum_{i=1}^{8}\exp(\mu x_{[i]})$

$$\nabla f(x) = \begin{bmatrix} \frac{\partial}{\partial g(x_1)} f(x_1)\cdot\frac{\partial}{\partial x_1} g(x_1) \\ \frac{\partial}{\partial g(x_1)} f(x_2)\cdot\frac{\partial}{\partial x_2} g(x_2) \\ \cdots \quad \cdots \\ \frac{\partial}{\partial g(x_n)} f(x_n)\cdot\frac{\partial}{\partial x_n} g(x_n) \end{bmatrix} = = \begin{bmatrix} \frac{1}{\mu}\frac{\mu\cdot\exp(\mu x_{[1]})}{\sum_{i=1}^{8}\exp(\mu x_{[1]})} \\ \frac{1}{\mu}\cdot\frac{\mu\cdot\exp(\mu x_{[2]})}{\sum_{i=1}^{8}\exp(\mu x_{[i]})} \\ \cdots \\ \frac{1}{\mu}\cdot\frac{\mu\cdot\exp(\mu x_{[n]})}{\sum_{i=1}^{8}\exp(\mu x_{[8]})} \end{bmatrix} = \begin{bmatrix} \frac{\exp(\mu x_{[1]})}{\sum\exp(\mu x_{[i]})} \\ \frac{\exp(\mu x_{[2]})}{\sum\exp(\mu x_{[i]})} \\ \cdots \\ \frac{\exp(\mu x_{[8]})}{\sum\exp(\mu x_{[8]})} \end{bmatrix}$$

we can set the sum as a term $S(x)$.

so $\nabla f(x) = \frac{1}{S(x)}\begin{bmatrix} \exp(\mu x) \\ \cdots \\ \exp(\mu x_8) \end{bmatrix}$, the dimension is 8.

3. Convex or not convex.

(a) $S = \{x : \Sigma_i x_i = 0\}$.

Let $m = \{m : \Sigma_i m_i = 0\} \in S$.

$n = \{n : \Sigma_i n_i = 0\} \in S$,

set α, $\alpha \in [0,1]$.

$\alpha m + (1-\alpha)n = \alpha \Sigma_i m_i + (1-\alpha)\Sigma_i n_i = \alpha \cdot 0 + (1-\alpha)\cdot 0 = 0$

so $\alpha m + (1-\alpha)n \in S$,

so $S = \{x : \Sigma_i x_i = 0\}$ is convex.

(b) $S = \{(x, y) : x^2 + y^2 = 1\}$

$x^2+y^2=1$ shows the side of a circle whose radius is 1.

so we can assume that

$m = (x=0, y=1)$ and $n = (m=1, n=0)$.

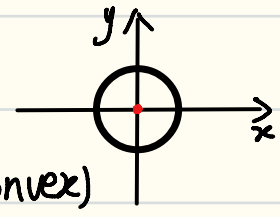
$\alpha \in [0,1]$, so we can assume $\alpha = 1/3$,

so $\alpha m + (1-\alpha)n = 1/3(0,1) + 2/3(1,0)$

$= (0, 1/3) + (2/3, 0)$

$x^2 + y^2 = (1/3)^2 + (2/3)^2 = 1/9 + 4/9 = 5/9 \neq 1$

so $\alpha m + (1-\alpha)n \notin S$.

so S is not convex. (Based on the graph of $x^2+y^2=1$, 

, we also can easily know that S is not convex)

(c) $S = \{x : |x| \leq 1\}$.

Let $m = \{m : |m| \leq 1\}$.

$n = \{n : |m| \leq 1\}$.

set α, $\alpha \in [0,1]$

$|\alpha m + (1-\alpha)n| \leq |\alpha m| + |(1-\alpha)n| = \alpha|m| + (1-\alpha)|n|$

$\leq \alpha \cdot 1 + (1-\alpha)\cdot 1 = 1$

So $|\alpha m + (1-\alpha)n| \in S$.

So S is convex.

4. Am I positive semidefinte?

(a) $X = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 4 \\ 3 & 4 & 1 \end{bmatrix}$, X is positive semidefinite for all v, $v^T H v \geq 0$.

so $v^T X v = [v_1\ v_2\ v_3]\begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 4 \\ 3 & 4 & 1 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = v_1^2 + 2v_1v_2 + 3v_1v_3 + 2v_1v_2 + v_2^2 + 4v_2v_3 + 3v_1v_3 + 4v_2v_3 + v_3^2$

$= v_1^2 + v_2^2 + v_3^2 + 4v_1v_2 + 6v_1v_3 + 8v_2v_3$

we can set $v = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$, so $v^T = [-1\ \ 1\ \ -1]$.

so $v^T X v = 1 + 1 + 1 - 4 + 6 - 8 = -3 < 0$

so X is not positive semidefinite. a counter example is $v = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$

(b) $X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$.

$v^T X v = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}[v_1\ v_2\ v_3] = v_1^2 + 2v_2^2 + 3v_3^2$

we know that, for any v, $v^2 \geq 0$. so $v_1^2 + 2v_2^2 + 3v_3^2 \geq 0$

so $v^T X v \geq 0$, so X is positive semidefinite.

5. Convex or not convex.

(a) $f(x) = 1/x$, for $x > 0$.

By definition, let $x > 0$, $y > 0$, and $\alpha \in [0, 1]$,

$f(\alpha x + (1-\alpha)y) = \frac{1}{\alpha x + (1-\alpha)y}$

$\alpha f(x) + (1-\alpha) f(y) = \frac{\alpha}{x} + \frac{(1-\alpha)}{y} = \frac{\alpha y + (1-\alpha)x}{xy}$

We can assume that $f(\alpha x+(1-\alpha)y) > \alpha f(x)+(1-\alpha)f(y)$

So $\frac{1}{\alpha x+(1-\alpha)y} > \frac{\alpha y+(1-\alpha)x}{xy}$ $(x>0,\ y>0)$

$$xy > (\alpha y+(1-\alpha)x)\cdot(\alpha x+(1-\alpha)y)$$

$$xy > \alpha^2 xy + \alpha y\cdot(1-\alpha)y + (1-\alpha)x\cdot\alpha x + (1-\alpha)^2 xy$$

$$xy > \alpha^2 xy + \alpha y^2 - \alpha^2 y^2 + \alpha x^2 - \alpha^2 x^2 + xy - \alpha^2 xy$$

$$xy > xy + \alpha(1-\alpha)y^2 + \alpha(1-\alpha)x^2$$

we know that $\alpha(1-\alpha)y^2+\alpha(1-\alpha)x^2 > 0$, so the above is wrong.

So $xy \le xy+\alpha(1-\alpha)y^2+\alpha(1-\alpha)x^2$ for $x>0,\ y>0,\ \alpha\in[0,1]$

so $f(x)$ is convex, by definition of convexity.

(b) $f(x)=\|x\|_\infty$

let $x,\ y,\ \alpha\in[0,1]$.

$f(\alpha x+(1-\alpha)y) = \|\alpha x+(1-\alpha)y\|_\infty$

by subordinate matrix infinity norm.

$$\|x\|_\infty = \max\sum^n |a_{ij}|$$

So $\|\alpha x+(1-\alpha)y\| \le \sum^n(\alpha x_{ij}+(1-\alpha)y_{ij})$

$$= \sum^n(\alpha x_{ij}) + \sum^n(1-\alpha)y_{ij}.$$

So $\|\alpha x+(1-\alpha)y\|_\infty \le \|\alpha x\|_\infty + \|(1-\alpha)y\|_\infty = \alpha\|x\|_\infty + (1-\alpha)\|y\|_\infty$

So $f(\alpha x+(1-\alpha)y) \le \alpha f(x)+(1-\alpha)f(y)$

so $f(x)$ is convex, by definition of convexity.

(C) $f(x)= x_{[3]}^3 + x_{[2]}^2 + x_{[1]}$

let $H = \nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 x_2} & \frac{\partial^2 f}{\partial x_1 x_3} \\ \frac{\partial^2 f}{\partial x_2 x_1} & \frac{\partial^2 f}{\partial x_2^2} & \frac{\partial^2 f}{\partial x_2 x_3} \\ \frac{\partial^2 f}{\partial x_3 x_1} & \frac{\partial f}{\partial x_3 x_2} & \frac{\partial^2 f}{\partial x_3^2} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 6x_3 \end{bmatrix}$

for each part:

$g_3(x)= x^3,\ g_3''(x) = 6x;$

$g_2(x)= x^2,\ g_2''(x) = 2;$

$g_1(x) = x,\ g_1''(x) = 0.$

$$v^T H v = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix}\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 6x_3 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = 2v_2^2 + 6v_3^2 x_3$$

for all v, $v^2 \ge 0$, but for all x, $2v_2^2+6v_3^2x_3$ could be negative

so $f(x)$ is not convex, by second order condition.

(d) $f(x) = \|Ax - b\|_2^2$

assume A is symmetic matric, we can set that

assume matrix A is [m, m] and symmetric, assume $g(x) = Ax - b$, $x = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_n \end{bmatrix}$

So $H = \nabla^2 f(x) = \begin{bmatrix} \frac{\partial}{\partial g(x)} f(x) \frac{\partial}{\partial x} g(x) \\ \cdots \cdots \\ \frac{\partial}{\partial g(x)} f(x) \frac{\partial}{\partial x} g(x) \end{bmatrix}$ by chain rule: $((ax-b)^2)'' = (2(ax-b)' = 2a$

$$= \begin{bmatrix} 2A[1.1](A[1.1] + A[2.1] + \ldots + A[m.1]) \\ \cdots \cdots \\ 2A[1.n](A[1.n] + A[2.n] + \cdots + A[m.n]) \end{bmatrix}$$

$$= 2A^T A$$

so, for all v, we have $V^T \cdot H \cdot V = V^T \cdot 2A^T A \cdot V \geq 0$

so $H(\nabla^2 f(x))$ is positive semidefinite, so $f(x)$ is convex, by second order condition.

6. (a) Linear regression.

i. normal equations for problem (2): $\underset{\theta \in \mathbb{R}^p}{\text{minimize}} \frac{1}{2}\|X\theta - y\|_2^2$.

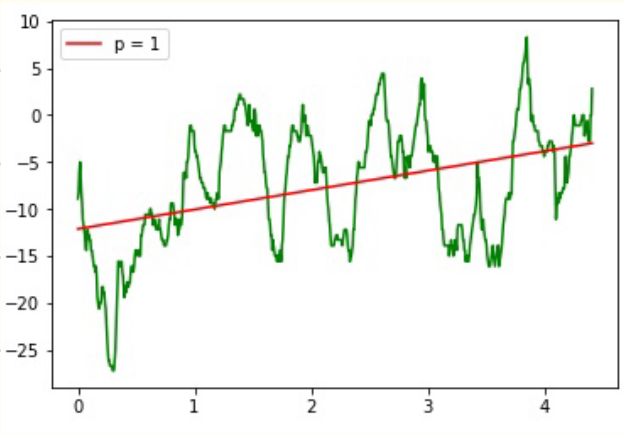
$(X^T X)\theta = X^T y$, $\theta = (X^T X)^{-1} X^T y$

ii.

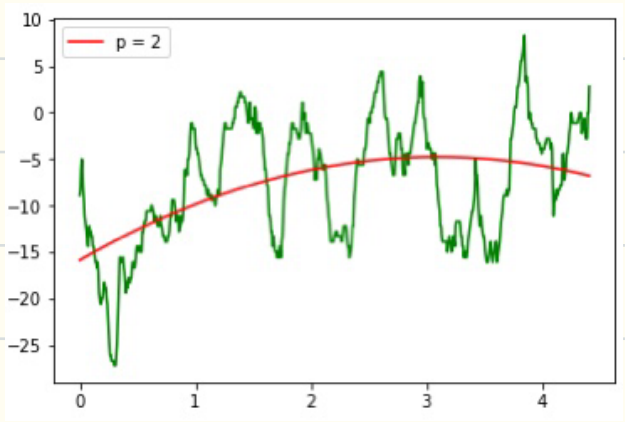
```
[13] #Question (a)
     def packX(z,poly_order):
         X = np.zeros((len(z),poly_order+1))

         for i in range(poly_order+1):
           X[:, i] = np.array(z)**i

         return X


     def solveLinearSystem(X,y):
         theta = np.linalg.solve(np.dot(X.T, X), np.dot(X.T, y)) #normal equations
         return theta

     # TEST SCRIPT. DO NOT MODIFY!
     X = packX(range(100),3)
     y = np.sqrt(np.array(range(100)))
     theta = solveLinearSystem(X,y)
     print('Check number is', np.sum(theta))

     Check number is 1.341270179610586
```

Check number ≈ 1.341

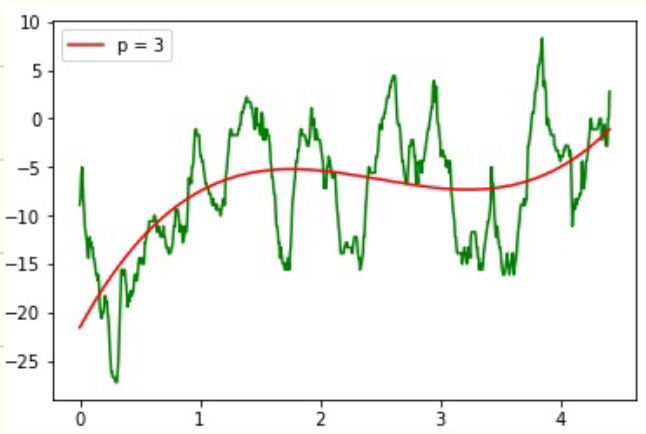
iii. p=1.



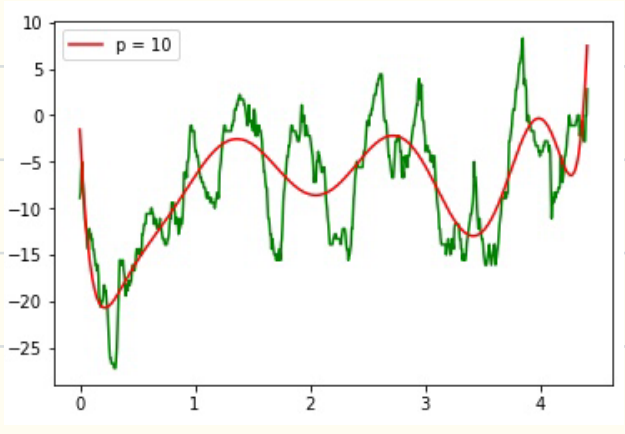
p=2:



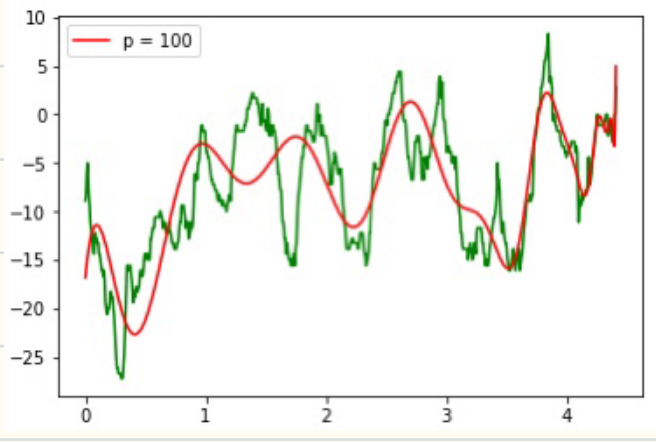
p=3:



p=10:



p=100:



(b) Ridge regression.

i. normal equation for problem (3). $\underset{\theta \in R^p}{\text{minimize}} \frac{1}{2}\|X\theta - y\|_2^2 + \frac{\rho}{2}\|\theta\|_2^2$

$(X^TX + \rho)\theta = X^Ty$

$\theta = (X^TX + \rho)^{-1}X^Ty$

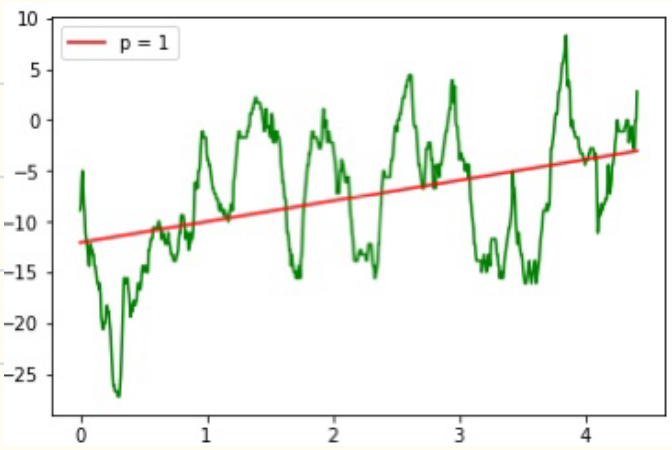
ii.

```
def solveRidgeRegressionSystem(X,y,rho):
    theta = np.linalg.solve(np.matmul(X.T, X) + rh
    return theta

# TEST SCRIPT. DO NOT MODIFY!
X = packX(range(100),3)
y = np.sqrt(np.array(range(100)))
theta = solveRidgeRegressionSystem(X,y,1)
print('Check number is', np.sum(theta))
```

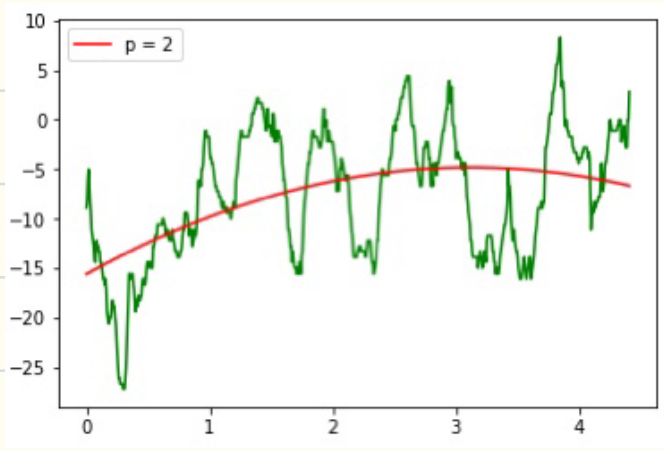
```
Check number is 1.2061712965226425
```

Check number ≈ 1.206.

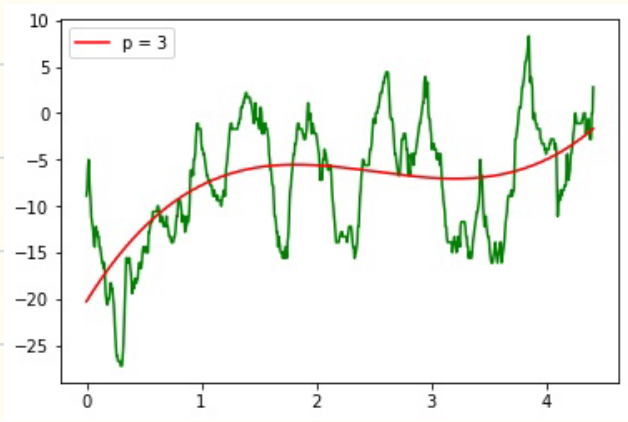
iii. p=1:



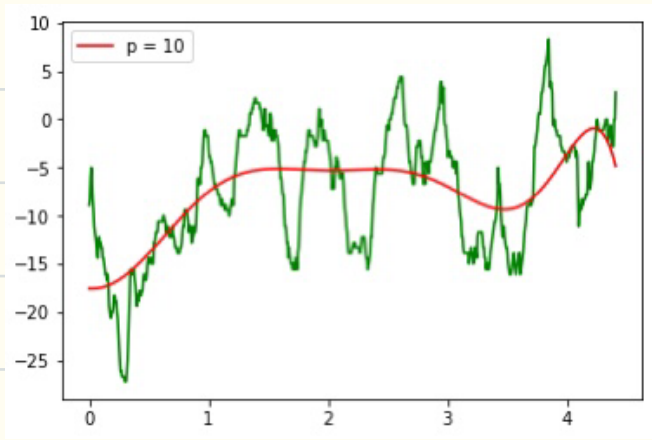
p=2:



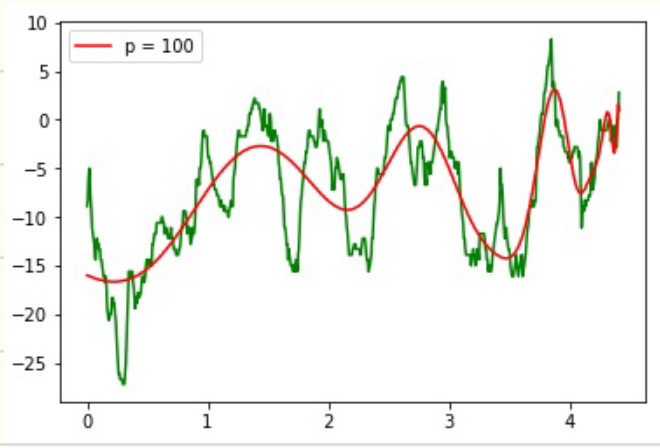
p=3:



p= 10:



p=100.



(c) conditioning.

```
[ ]  pLst1 = [1, 2, 5, 10]
```

```
for p in pLst1:
  for rho in [0, N, N*10, N*100]:
    X = packX(weeks_after_start, p)
    Q = np.dot(X.T, X) + rho*np.identity(X.shape[1])
    KQ = np.linalg.cond(Q)
    print('P'+ ' || ' + str(p) + ' || ' + 'rho'+ ' || ' + str(rho) + ' || ' + 'K(Q)'+ ' || ' + str(KQ))
```

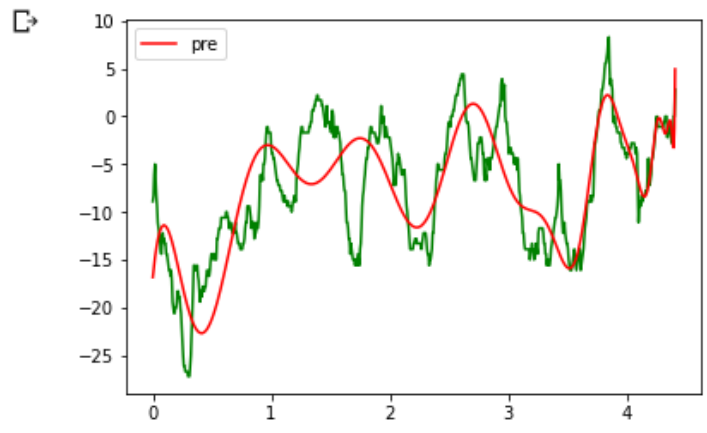
```
P || 1 || rho || 0 || K(Q) || 32.472521666398684
P || 1 || rho || 742 || K(Q) || 6.754235401156341
P || 1 || rho || 7420 || K(Q) || 1.6887590213669572
P || 1 || rho || 74200 || K(Q) || 1.0702597386922308
P || 2 || rho || 0 || K(Q) || 1476.3924507440943
P || 2 || rho || 742 || K(Q) || 79.11371158831943
P || 2 || rho || 7420 || K(Q) || 9.202205647238008
P || 2 || rho || 74200 || K(Q) || 1.8243450966088195
P || 5 || rho || 0 || K(Q) || 730851302.3775175
P || 5 || rho || 742 || K(Q) || 271693.24960615137
P || 5 || rho || 7420 || K(Q) || 27179.318088769916
P || 5 || rho || 74200 || K(Q) || 2718.922773636268
P || 10 || rho || 0 || K(Q) || 7.157429647449161e+18
P || 10 || rho || 742 || K(Q) || 397031762841.3795
P || 10 || rho || 7420 || K(Q) || 39703178289.463005
P || 10 || rho || 74200 || K(Q) || 3970317849.5276527
```

```
    y_pre = np.dot(X, theta)

    mean_squared_error = np.sum((y_pre - dewtemp) ** 2)/N

    print('P'+ ' || ' + str(p) + ' || ' + 'rho'+ ' || ' + str(rho) + ' || ' + 'mean
```

```
P || 1 || rho || 0 || mean squard error || 36.38217361905994
P || 1 || rho || 742 || mean squard error || 58.80283013948351
P || 1 || rho || 7420 || mean squard error || 78.34463001407917
P || 1 || rho || 74200 || mean squard error || 96.31928655068175
P || 2 || rho || 0 || mean squard error || 33.53185241449123
P || 2 || rho || 742 || mean squard error || 57.61883781375108
P || 2 || rho || 7420 || mean squard error || 77.25615112057318
P || 2 || rho || 74200 || mean squard error || 86.84731522758038
P || 5 || rho || 0 || mean squard error || 27.0671045152648
P || 5 || rho || 742 || mean squard error || 57.23483324805714
P || 5 || rho || 7420 || mean squard error || 71.9915827133517
P || 5 || rho || 74200 || mean squard error || 77.48731086603902
P || 10 || rho || 0 || mean squard error || 16.707135186130255
P || 10 || rho || 742 || mean squard error || 55.38371919899731
P || 10 || rho || 7420 || mean squard error || 71.3923471023423
P || 10 || rho || 74200 || mean squard error || 76.7444564683475
```

(d)

```
pM = 10
rhoM = 0
X = packX(weeks_after_start, p)
theta = solveRidgeRegressionSystem(X, dewtemp, rhoM)
y_pre = np.dot(X, theta)


plt.plot(weeks_after_start, dewtemp, c = 'green')
plt.plot(weeks_after_start, y_pre, label = 'pre', c = 'red')

plt.legend()
plt.show()
```



I pick (p=10, ρ=0), now the mean squard erro is the minimum (≈16.707). Less erro means our predicate will be more accurate.

7. (a) $f(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\log(\sigma(y_i x_i^T\theta))$, $\sigma(s) = \frac{1}{1+e^{-s}}$

$$f_i(\theta) = \log\sigma(y_i x_i^T\theta)$$

Gradient: $\nabla f(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\nabla f_i(\theta)$

$$\nabla f_i(\theta) = (\sigma(y_i x_i^T\theta) - 1)\cdot y_i x_i$$

$$\nabla f(\theta) = -\frac{1}{m}\sum_{i=1}^{m}(\sigma(y_i x_i^T\theta) - 1) y_i x_i$$

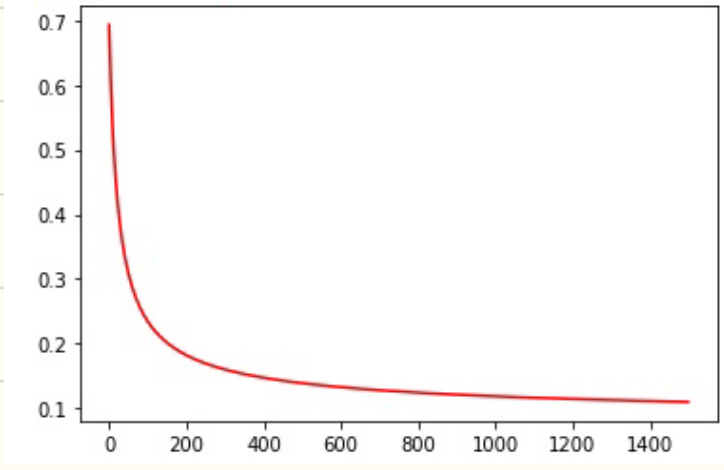
(b)

```
    theta = theta.reshape((n, 1))
    y = np.copy(ytrain)
    y[y==-1] = 0
    y = y.reshape((m, 1))
    h = sigmoid(X.dot(theta))
    res = ((1 / m) * X.T.dot(h - y))
    res = res.reshape(-1)
    # print(res.shape)
    return res


# TEST SCRIPT. DO NOT MODIFY!
theta = np.linspace(-.1,.1,n)
print('Check number is', getLossFunction(theta),np.sum(getGradient(theta)))
```

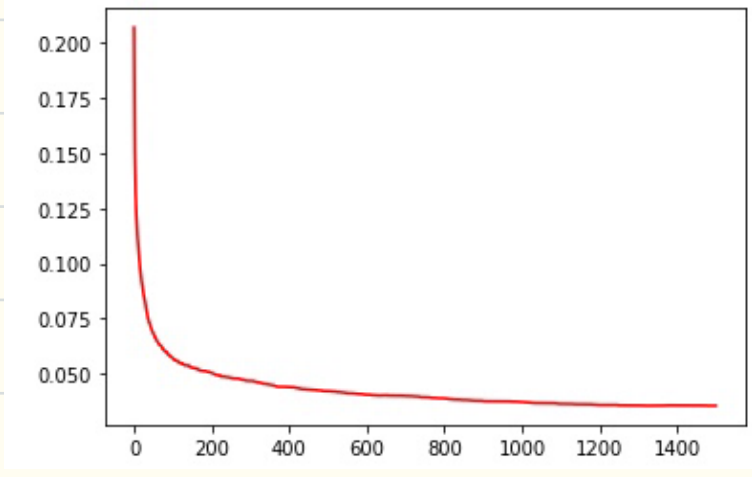
```
Check number is 45.19215648734918 12343.176947604472
```

- check number ≒ 45.192 12343.177

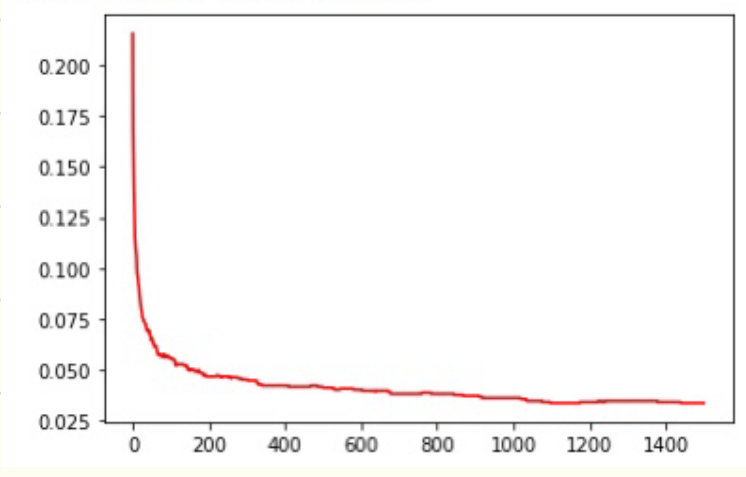
- training loss:



training misclassification rate:



test misclassification rate:



final train accuracy : 0.9645;
final test accuracy : 0.9663.

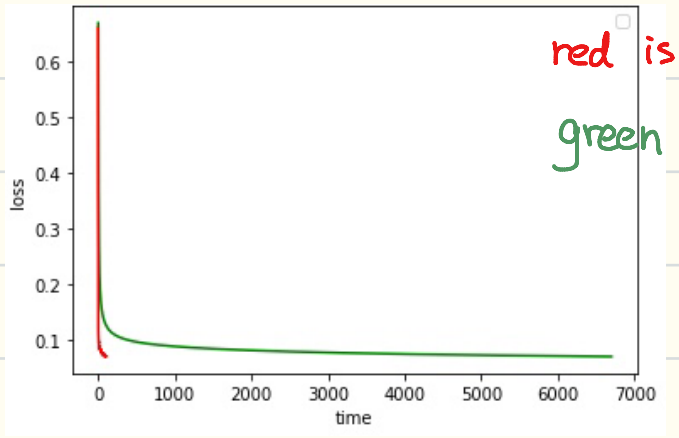
(c)

```
#Question(c)
def getStochGradient(theta, minibatch):
    thetaC = np.zeros(n)

    for i in minibatch:
        thetaC = thetaC + (sigmoid(ytrain[i] * np.dot(Xtrai

    thetaC = thetaC / len(minibatch)
    return thetaC

# TEST SCRIPT. DO NOT MODIFY!
theta = np.linspace(-.1,.1,n)
print('Check number is',np.sum(getStochGradient(theta,[1,4,
```

```
Check number is 5803.5
```

check number is 5803.5.

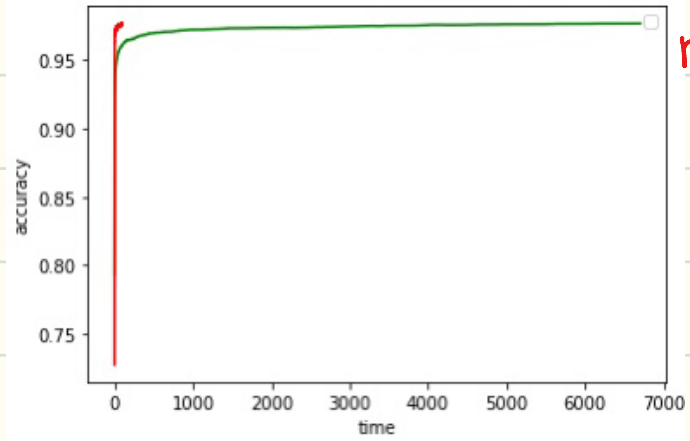
(d) objective loss:



red is stochastic gradient descent

green is gradient descent

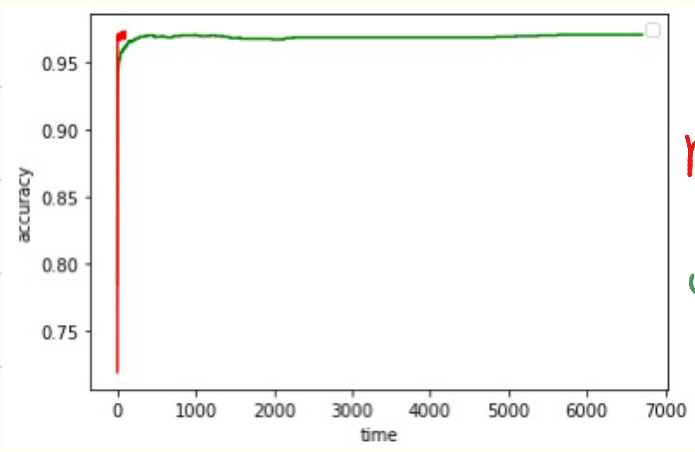
train accuracy:



red is stochastic gradient descent

green is gradient descent

test accuracy:



red is stochastic gradient descent

green is gradient descent

By running 50000 iterations, stochastic gradient descent is much slower than gradient descent. (more than 30 mins difference)

But the test accuracy of stochastic gradient descent is a bit higher than gradient descent. (about 0.002 difference)