CS 189 Introduction to Machine Learning
Fall 2017 HW3

This homework is due **Friday, September 15 at 12pm noon.**

1 Getting Started

You may typeset your homework in latex or submit neatly handwritten and scanned solutions. Please make sure to start each question on a new page, as grading (with Gradescope) is much easier that way! Deliverables:

1. Submit a PDF of your writeup to assignment on Gradescope, "HW[n] Write-Up"
2. Submit all code needed to reproduce your results, "HW[n] Code".
3. Submit your test set evaluation results, "HW[n] Test Set".

After you've submitted your homework, be sure to watch out for the self-grade form.

(a) Before you start your homework, write down your team. Who else did you work with on this homework? List names and email addresses. In case of course events, just describe the group. How did you work on this homework? Any comments about the homework?

None
Comments: HW is long

(b) Please copy the following statement and sign next to it:

I certify that all solutions are entirely in my words and that I have not looked at another student's solutions. I have credited all external sources in this write up.

I certify that all solutions are entirely in my words and that I have not looked at another students' solutions. I have credited all external sources in this write up.

Problem # 2

(a) $Y = Xw + b + z$

$P(Y|X) = P(Xw + b + z \mid X)$

We know $P(z) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$

$\Rightarrow P(Y|X) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-f(x))^2}{2\sigma^2}}$ where $f(x) = Xw + b$

(b) We want to maximize:

$P(Y_1, Y_2, \dots, Y_n \mid X_1, X_2, \dots, X_n; w, b)$

$= P(Y_1 \mid X_1; w, b) \times P(Y_2 \mid X_2; w, b) \times \dots \times P(Y_n \mid X_n; w, b)$

Take the log:

$\max \log P(Y_1, Y_2, \dots, Y_n \mid X_1, X_2, \dots, X_n; w, b)$

$= \max \sum_1^n \log P(Y_i \mid X_i; w, b)$

$= \min \sum_1^n \frac{(y_i - f(x_i))^2}{2\sigma^2} - n \log(\sqrt{2\pi}\sigma)$

The maximum likelihood estimator (MLE)

$\arg\min_{w,b} \sum_1^n (y_i - f(x_i))^2$

(c) $Z \sim U[-0.5, 0.5]$

$$\Rightarrow P(z) = \begin{cases} 1 & -0.5 \le z \le 0.5 \\ 0 & \text{otherwise} \end{cases}$$

$$P(Y|X) = P(z = y - xw \mid x) = \begin{cases} 1 & -0.5 \le y - xw \le 0.5 \\ 0 & \text{otherwise} \end{cases}$$

$$\Rightarrow P(Y|X) = P(-0.5 + xw \le y \le 0.5 + xw)$$

(d) we want to maximize:

$$P(y_1, y_2, \dots, y_n \mid x_1, x_2, \dots, x_n; w)$$

$$= \prod_1^n P(y_i \mid x_i; w)$$

$$= \prod_1^n P(-0.5 \le y_i - x_i w \le 0.5)$$

The maximum likelihood estimator:

$$\underset{w}{\operatorname{argmax}} \prod_1^n P(-0.5 \le y_i - x_i w \le 0.5)$$

we know that $P(-0.5 \le y_i - x_i w \le 0.5) = \begin{cases} 1 & -0.5 \le y_i - x_i w \le 0.5 \\ 0 & \text{otherwise} \end{cases}$

$$\Rightarrow \max \prod_1^n P(-0.5 \le y_i - x_i w \le 0.5) = 1$$

iff $\quad -0.5 \le y_i - x_i w \le 0.5 \quad \forall x_i, y_i$

$$\frac{y_i - 0.5}{x_i} \le w \le \frac{y_i + 0.5}{x_i} \quad \forall x_i, y_i$$

assuming that $x_i > 0 \quad \forall i$

more conservative

$$\frac{\max(y) - 0.5}{\min(x)} \le w \le \frac{\min(y) + 0.5}{\max(x)}$$

e.) See appendix

(f) The Bayes' theorem gives:

$$P(W|Y,X) = \frac{P(Y|W,X)\,P(W|X)}{\int_W P(Y|W,X)\,P(W,X)\,dW}$$

($P(W|X) \to P(W)$, $P(W,X) \to P(W)$)

$$= \frac{P(Y|W,X)\,P(W)}{\int_W P(Y|W,X)\,P(W)\,dW}$$

$$W \sim N(0,\sigma^2) \Rightarrow P(W) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{W^2}{2\sigma^2}}$$

$$Z_i \sim N(0,1) \Rightarrow P(Z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{Z^2}{2}}$$

$$Y_i = X_i W + Z_i \Rightarrow Y_i \sim N(X_i W, 1)$$

$$\Rightarrow P(Y|W,X) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(Y-XW)^2}{2}}$$

data, prior

$$\Rightarrow P(W|Y,X) = \frac{\frac{1}{\sqrt{2\pi}} e^{-\frac{(Y-XW)^2}{2}} \times \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{W^2}{2\sigma^2}}}{\int_W \frac{1}{\sqrt{2\pi}} e^{-\frac{(Y-XW)^2}{2}} \times \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{W^2}{2\sigma^2}}\,dW}$$

The denominator is constant. $C =$

$$\Rightarrow P(W|Y,X) = \frac{1}{2\pi\sigma C} e^{-\frac{(Y-XW)^2}{2} - \frac{W^2}{2\sigma^2}}$$

We consider the exponent:

$$-\frac{(Y-XW)^2}{2} - \frac{W^2}{2\sigma^2} = -\frac{\sigma^2 Y^2 - 2\sigma^2 XYW + \sigma^2 X^2 W^2 + W^2}{2\sigma^2}$$

$$= -\frac{W^2(1+\sigma^2X^2) - 2W\sqrt{1+\sigma^2X^2}\,\frac{\sigma^2 XY}{\sqrt{1+\sigma^2X^2}} + \frac{\sigma^4X^2Y^2}{1+\sigma^2X^2} + \sigma^2Y^2 - \frac{\sigma^4X^2Y^2}{1+\sigma^2X^2}}{2\sigma^2}$$

$$= -\frac{\left(w\sqrt{1+\sigma^2 x^2} - \frac{\sigma^2 xy}{\sqrt{1+\sigma^2 x^2}} \right)^2 + \text{const}}{2\sigma^2}$$

$$\Rightarrow \quad E[w \mid y, x] = \frac{\sigma^2 xy}{1+\sigma^2 x^2} \qquad \text{where} \quad xy = \frac{1}{n}\sum x_i y_i$$

$$x^2 = \frac{1}{n}\sum x_i^2$$

(g) $Z_i \sim N(0,1)$

$$Y_i = w^T X_i + Z_i \Rightarrow Y_i \sim N(w^T X_i, 1)$$

$$\Rightarrow P(Y_i) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(Y_i - w^T X_i)^2}{2}}$$

MLE is to maximize :

$$P(Y_1, Y_2, \dots, Y_n \mid X_1, X_2, \dots, X_n ; w)$$

$$= P(Y_1 \mid X_1; w)\, P(Y_2 \mid X_2; w) \dots P(Y_n \mid X_n; w)$$

take log :

$$\underset{w}{\operatorname{argmax}} \log P = \underset{w}{\operatorname{argmax}} \sum_1^n \log P(Y_i \mid X_i; w)$$

$$= \underset{w}{\operatorname{argmin}} \sum_1^n \frac{(y_i - w^T x_i)^2}{2} + n \log \sqrt{2\pi}$$

$$= \underset{w}{\operatorname{argmin}} \sum_1^n (y_i - w^T x_i)^2$$

$$= \underset{w}{\operatorname{argmin}} \| Y^T - w^T X \|_2^2$$

↑

Least Square Regression

(h) Similar to problem (f):

$$P(W|Y,X) = \frac{P(Y|W,X)\,P(W|X)}{\int_W P(Y|W,X)\,P(W,X)\,dW}$$

($P(W|X) \to P(W)$, $P(W,X) \to P(W)$)

$$= \frac{P(Y|W,X)\,P(W)}{\int_W P(Y|W,X)\,P(W)\,dW}$$

$$W_i \sim N(0,\sigma^2) \Rightarrow P(W) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\|W\|_2^2}{2\sigma^2}}$$

$$Z_i \sim N(0,1) \Rightarrow P(Z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\|Z\|_2^2}{2}}$$

$$Y_i = W^T X_i + Z_i \Rightarrow Y_i \sim N(W^T X_i, 1)$$

$$\Rightarrow P(Y|W,X) = \frac{1}{\sqrt{2\pi}} e^{-\frac{\|Y-XW\|_2^2}{2}}$$

$$\Rightarrow P(W|Y,X) = \frac{\frac{1}{\sqrt{2\pi}} e^{-\frac{\|Y-XW\|_2^2}{2}} \cdot \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\|W\|_2^2}{2\sigma^2}}}{\int_W \frac{1}{\sqrt{2\pi}} e^{-\frac{(Y-XW)^2}{2}} \times \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{W^2}{2\sigma^2}} dW}$$

(constant)

To calculate the mean, similarly we have:

$$E[W|Y,X] = (I + \sigma^2 X^T X)^{-1} \sigma^2 XY$$

(i) See appendix

Problem #3

(a) 1. $\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n}$

$\Rightarrow E[\hat{X}] = \frac{E[x_1] + E[x_2] + \dots + E[x_n]}{n} = \frac{n\mu}{n} = \mu$

$\Rightarrow$ bias $E[\hat{X} - \mu] = E[\hat{X}] - \mu = \mu - \mu = 0$

2. $\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n+1}$

$\Rightarrow E[\hat{X}] = \frac{n\mu}{n+1}$

$\Rightarrow$ bias $E[\hat{X} - \mu] = E[\hat{X}] - \mu = \frac{n\mu}{n+1} - \mu = -\frac{\mu}{n+1}$

3. $\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n + n_0}$

$\Rightarrow E[\hat{X}] = \frac{n\mu}{n+n_0}$

$\Rightarrow$ bias $E[\hat{X} - \mu] = E[\hat{X}] - \mu = \frac{n\mu}{n+n_0} - \mu = -\frac{n_0\mu}{n+n_0}$

4. $\hat{X} = 0$

$\Rightarrow E[\hat{X}] = 0$

$\Rightarrow$ bias $E[\hat{X} - \mu] = E[\hat{X}] - \mu = 0 - \mu = -\mu$

(b) 1. $\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n}$

$\Rightarrow Var[\hat{X}] = \frac{1}{n^2} Var[x_1 + x_2 + \dots + x_n] =$

$= \frac{Var(x_1) + Var(x_2) + \dots + Var(x_n)}{n^2} = \frac{n\sigma^2}{n^2} = \frac{\sigma^2}{n}$

2. $$\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n+1}$$

$$\Rightarrow \text{Var}[\hat{x}] = \frac{1}{(n+1)^2} \text{Var}[x_1 + x_2 + \dots + x_n] = \frac{n\sigma^2}{(n+1)^2}$$

3. $$\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n + n_0}$$

$$\Rightarrow \text{Var}[\hat{X}] = \frac{1}{(n+n_0)^2} \text{Var}[x_1 + x_2 + \dots + x_n] = \frac{n\sigma^2}{(n+n_0)^2}$$

4. $$\hat{X} = 0$$

$$\Rightarrow \text{Var}[\hat{X}] = 0$$

(c)

1. $$E[\hat{x} - \mu] = 0 \qquad \text{Var}[\hat{x}] = \frac{\sigma^2}{n}$$

$$\Rightarrow \text{ETE} = 0^2 + \frac{\sigma^2}{n} = \frac{\sigma^2}{n}$$

2. $$E[\hat{x} - \mu] = -\frac{\mu}{n+1} \qquad \text{Var}[\hat{x}] = \frac{n\sigma^2}{(n+1)^2}$$

$$\Rightarrow \text{ETE} = \frac{\mu^2}{(n+1)^2} + \frac{n\sigma^2}{(n+1)^2} = \frac{\mu^2 + n\sigma^2}{(n+1)^2}$$

3. $$E[\hat{X} - \mu] = -\frac{n_0\mu}{n+n_0} \qquad \text{Var}[\hat{x}] = \frac{n\sigma^2}{(n+n_0)^2}$$

$$\Rightarrow \text{ETE} = \frac{n_0^2\mu^2}{(n+n_0)^2} + \frac{n\sigma^2}{(n+n_0)^2} = \frac{n_0^2\mu^2 + n\sigma^2}{(n+n_0)^2}$$

4. $E[\hat{X} - \mu] = -\mu \qquad Var[\hat{X}] = 0$

$\Rightarrow \quad ETE = \mu^2$

(d) The 3rd estimator is the general case:

$$\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n + n_0}$$

with bias $E[\hat{X} - \mu] = -\frac{n_0 \mu}{n + n_0} \qquad Var[\hat{X}] = \frac{n\sigma^2}{(n+n_0)^2}$

$n_0 = 0$: $\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n}$

$E[\hat{X} - \mu] = 0 \qquad Var[\hat{X}] = \frac{\sigma^2}{n}$

→ case 1

$n_0 \to \infty$: $\hat{X} = \lim_{n_0 \to \infty} \frac{x_1 + x_2 + \dots + x_n}{n + n_0} = 0$

$$E[\hat{X} - \mu] = \lim_{n_0 \to \infty} -\frac{n_0 \mu}{n + n_0} = \lim_{n_0 \to \infty} -\frac{\mu}{\frac{n}{n_0} + 1} = -\mu$$

$$Var[\hat{X}] = \lim_{n_0 \to \infty} \frac{n\sigma^2}{(n+n_0)^2} = 0$$

→ case 4

$n_0 = 1$: $\hat{X} = \frac{x_1 + x_2 + \dots + x_n}{n + 1}$

$E[\hat{X} - \mu] = -\frac{1 \times \mu}{n + n_0} \qquad Var[\hat{X}] = \frac{n\sigma^2}{(n+1)^2}$

→ case 2

(e) $n_0 = \alpha n$

$$ETE = \frac{n_0^2\mu^2 + n\sigma^2}{(n+n_0)^2} = \frac{\alpha^2 n^2\mu^2 + n\sigma^2}{(n+\alpha n)^2}$$

$$\frac{d\,ETE}{d\alpha} = \frac{2\alpha n^2\mu^2(n+\alpha n)^2 - (2n^2 + 2\alpha n^2)(\alpha^2 n^2\mu^2 + n\sigma^2)}{(n+\alpha n)^4} = 0$$

$$\Rightarrow \alpha n\mu^2(1+\alpha)^2 - (1+\alpha)(\alpha^2 n\mu^2 + \sigma^2) = 0$$

$$\alpha n\mu^2(1+\alpha) - \alpha^2 n\mu^2 - \sigma^2 = 0$$

$$\alpha^2 n\mu^2 + \alpha n\mu^2 - \alpha^2 n\mu^2 - \sigma^2 = 0$$

$$\Rightarrow \alpha = \frac{\sigma^2}{n\mu^2}$$

(f) μ small & σ large

$\Rightarrow \alpha \nearrow^{\infty}$ to minimize ETE

(g) $\mu = \mu_0 \Rightarrow \alpha = \frac{\sigma^2}{n\mu_0^2} \Rightarrow n_0 = \alpha n = \frac{\sigma^2}{\mu_0^2}$

We have two approaches:

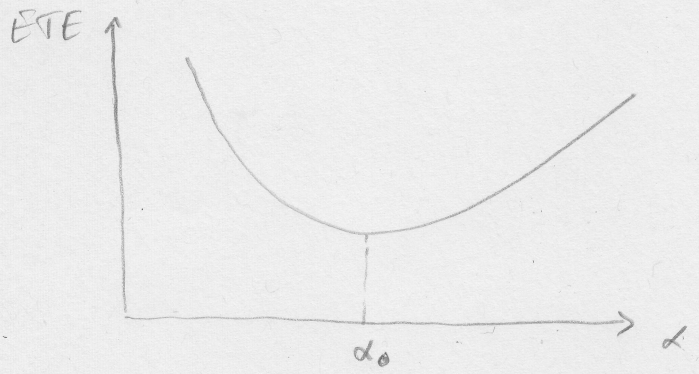
1. Estimate the mean of the sampling distribution of mean $\frac{x_1 + x_2 + \cdots + x_n}{n + n_0}$ where $n_0 = \frac{\sigma^2}{\mu_0^2}$

   by doing this, the mean has minimum value.

2. Refine X before estimating mean, that is

$$X' = X - \mu_0$$

(h) From question (e), we have the diagram of ETE vs α

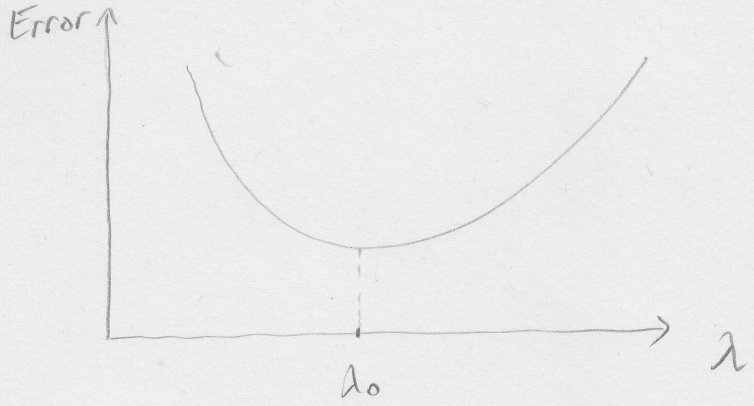


that is there is a value α_0 minimizing the expected total error ETE

$\alpha < \alpha_0$: under training, # of samples is not enough

$\alpha > \alpha_0$: over training, # of sample is too large

From lecture : Error v.s. λ in Ridge regression



$\lambda < \lambda_0$: over-fitting

$\lambda > \lambda_0$: under fitting, the regularization term dominates

$\Rightarrow$ α and λ share the same meaning in opposite direction.

Problem #4

(a) $\hat{X} = \underset{x}{\operatorname{argmin}} \| Y - Ax \|_2^2 = \underset{x}{\operatorname{argmin}} \| Ax^* + W - Ax \|_2^2$

$= \underset{x}{\operatorname{argmin}} \| W - A(x - x^*) \|_2^2$

$\Rightarrow \hat{X} - x^* = (A^TA)^{-1}A^TW \Rightarrow \hat{X} = (A^TA)^{-1}A^TW + x^*$

$\| A\hat{X} - y^* \|_2^2 = \| A(A^TA)^{-1}A^TW + \underbrace{Ax^*}_{y^*} - y^* \|_2^2$

$= \| A(A^TA)^{-1}A^TW \|_2^2$

(b) From (a):

$\| A\hat{X} - y^* \|_2^2 = \| A(A^TA)^{-1}A^TW \|_2^2$

$A = U\Sigma V^T$

$\| A(A^TA)^{-1}A^TW \|^2 = \| U\Sigma V^T (V\Sigma^TU^TU\Sigma V^T)^{-1} V\Sigma^TU^TW \|^2$

$= \| U \underbrace{\Sigma V^T(\Sigma V^T)^{-1}}_{I} \underbrace{(U^TU)^{-1}}_{I} \underbrace{(V\Sigma^T)^{-1}}_{I} V\Sigma^TU^TW \|^2$

$U^TU = I$ because $U_{full}U_{full}^T = U_{full}^TU = I$, $U_{full} = [u_1\ u_2 \dots u_n]$

U is U_{full} reduced by omitting some collumn

$\Rightarrow U^TU = I$ but $UU^T \neq I$

$\Rightarrow \| A\hat{X} - y^* \|_2^2 = \| UU^TW \|_2^2$

$= (UU^TW)^T(UU^TW)$

$= W^TU\underbrace{U^TU}_{I}U^TW = W^TUU^TW$

$= (U^TW)^TU^TW = \| U^TW \|_2^2$

(c) $W_i \sim N(0, \sigma^2) \Rightarrow U^T W = \begin{bmatrix} -u_1^T- \\ -u_2^T- \\ \vdots \\ -u_n^T- \end{bmatrix} W \sim N\left(0, \begin{bmatrix} \|u_1\|_2^2 \sigma^2 \\ \|u_2\|_2^2 \sigma^2 \\ \vdots \\ \|u_d\|_2^2 \sigma^2 \end{bmatrix}\right)$

but U is a reduced orthonormal matrix $\Rightarrow \|u_i\|_2^2 = 1 \quad \forall i$

$\Rightarrow U^T W \sim \left(0, \begin{bmatrix} \sigma^2 \\ \sigma^2 \\ \vdots \\ \sigma^2 \end{bmatrix}\right) \Rightarrow E[\|U^T W\|_2^2] - \mu(U^T W)^{\to 0}$

$= \underbrace{\sigma^2 + \sigma^2 + \dots + \sigma^2}_{d \text{ rows}} = d\sigma^2$

$$\frac{1}{n} E[\|A\hat{x} - y^*\|_2^2] = \frac{1}{n} E[\|U^T W\|_2^2]$$

$$= \frac{1}{n} d\sigma^2$$

d.) The problem set up is:

$$\min \|Ax - y\|_2^2$$

where $A \in R^{n \times (D+1)}$ is Vandermonde matrix

The average squared error:

$$\frac{1}{n} E[\|A\hat{x} - y^*\|_2^2] = \sigma^2 \frac{D+1}{n} \leq \varepsilon$$

$$\Rightarrow n \geq \frac{\sigma^2 (D+1)}{\varepsilon}$$

Conclusion: $D \uparrow \rightarrow n \uparrow$ proportionally w/ slope $\frac{\sigma^2}{\varepsilon}$

e.) See appendix

Problem # 5

See appendix

Problem #6

Prove that $E[X+Y] = E[X] + E[Y]$

$Var[X+Y] = Var[X] + Var[Y] + 2\,Cov(X,Y)$

Solution:

$$E[X] = \frac{\sum X_i}{n} \qquad E[Y] = \frac{\sum Y_i}{n}$$

$$E[X+Y] = \frac{\sum (X_i + Y_i)}{n}$$

$$\Rightarrow E[X] + E[Y] = \frac{\sum X_i}{n} + \frac{\sum Y_i}{n} = \frac{\sum (X_i + Y_i)}{n} = E[X+Y]$$

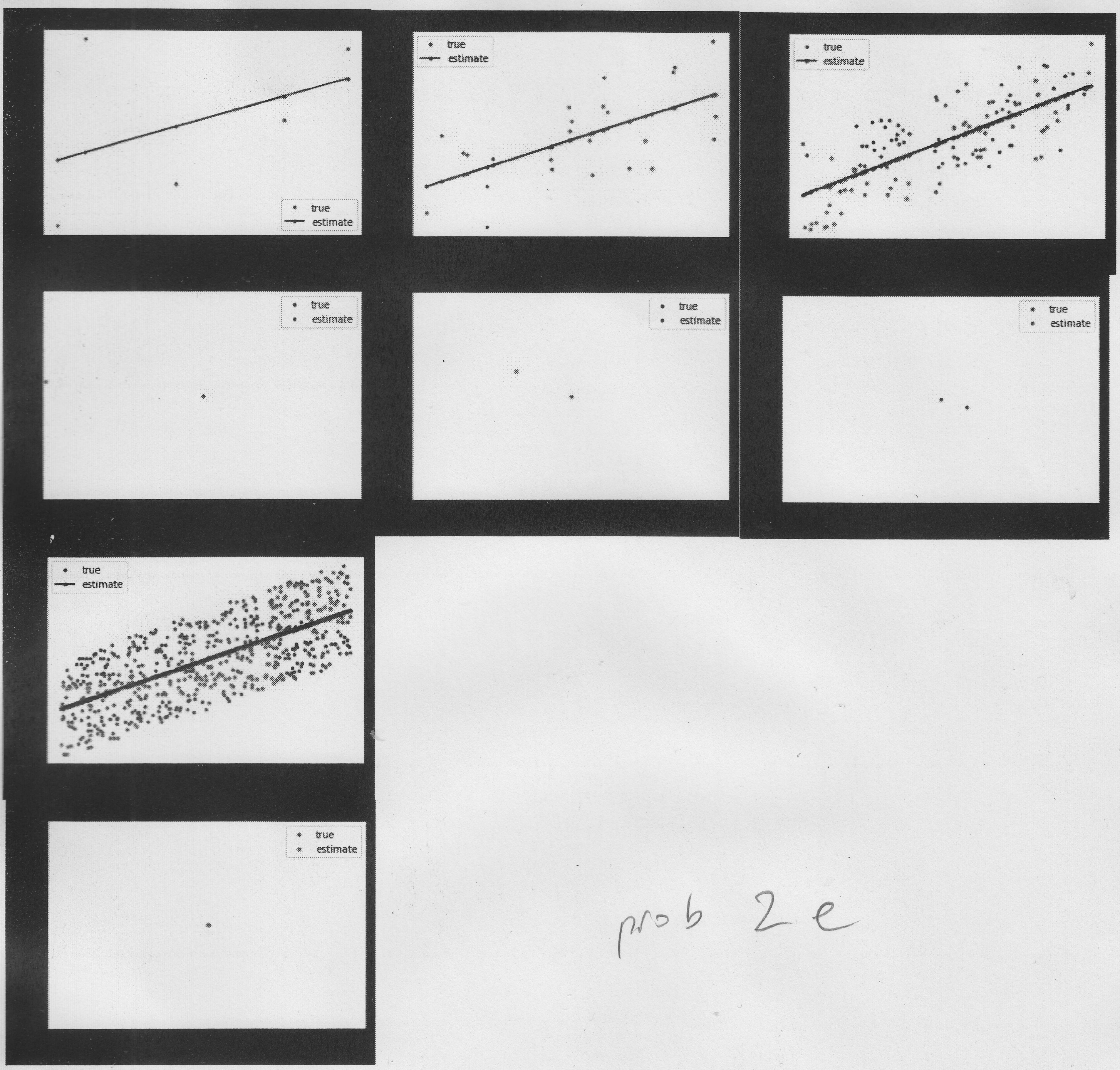
$$Var[X+Y] = E[(X+Y-\mu_{X+Y})^2]$$

$$= E[(X+Y)^2] - E[X+Y]^2$$

$$= E[X^2] + E[Y^2] + 2E[XY]$$

$$+ E[X]^2 + E[Y]^2 + 2E[X]E[Y]$$

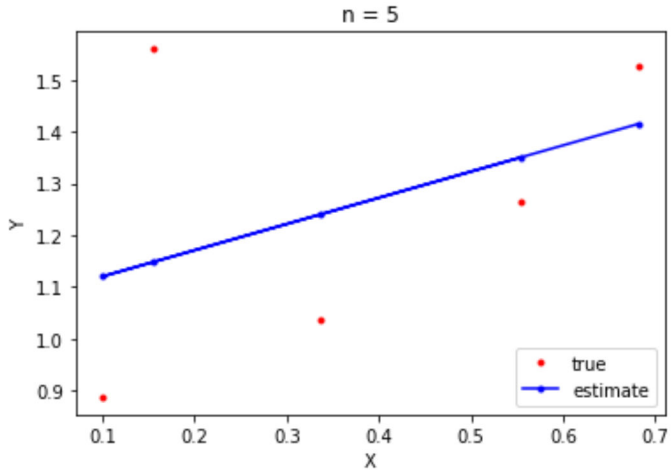
$$= Var[X] + Var[Y] + 2\,Cov(X,Y)$$

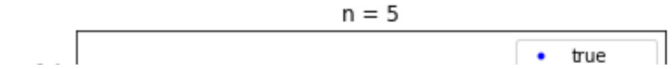


prob 2e

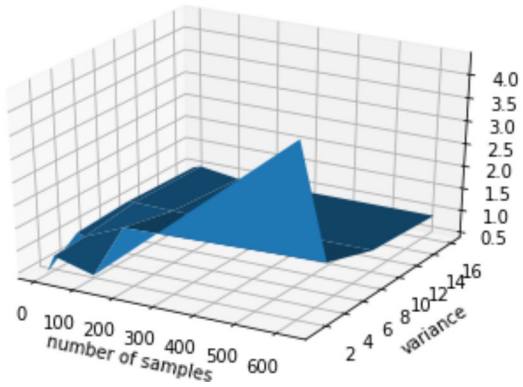
In [40]:
```
import pickle
import numpy as np
import os
import random
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm

def linear_regression_solver(X, Y):
    coeffs = np.matmul( np.linalg.inv( np.matmul(X.T, X) ), np.matmul(X.T, Y) )
    y_estimate = np.matmul(X, coeffs)
    return coeffs, y_estimate


def ridge_regression_solver(X, y, lamb):
    n, m = X.shape
    I = np.identity(m)
    norm_list = []
    features = np.matmul( np.linalg.inv( np.matmul(X.T, X) + lamb * I), np.matmul(X.T,
    for i in range(n):
        norm_list.append( np.linalg.norm(np.matmul(X[i,:], features) - y[i,:]) ** 2 )
    error = np.mean(norm_list)
    return features, error

def ridge_regression_solver_2(X, y, lamb):
    n, m = X.shape
    I = np.identity(m)
    norm_list = []
    features = np.matmul( np.linalg.inv( np.matmul(X.T, X) + lamb * I), np.matmul(X.T,
    for i in range(n):
        norm_list.append( np.linalg.norm(np.matmul(X[i,:], features) - y[i]) ** 2 )
    error = np.mean(norm_list)
    return features, error
```

In [41]:
```
# Problem 2
# Question 2.e

num = [5, 25, 125, 625]
w_star, b_star = 1, 1

for i in num:
    X = np.ones( (i, 2), "float64" )
    Y = np.zeros( (i, 1), "float64" )

    X[:, 0] = np.random.random(i)
    Z = np.random.uniform(-0.5, 0.5, i)
    Y = X[:, 0] * w_star + b_star + Z
    coeffs, y_estimate = linear_regression_solver(X, Y)
    w, b = coeffs
    plt.figure()
    plt.plot(X[:, 0], Y, "r.", label="true")
    plt.plot(X[:, 0], y_estimate , "b.-", label="estimate")
    plt.xlabel("X")
    plt.ylabel("Y")
    plt.legend()
    plt.title("n = " + str(i))

    plt.figure()
    plt.plot(w_star, b_star, "b.", label="true")
    plt.plot(w, b , "r.", label="estimate")
    plt.xlabel("slope")
    plt.ylabel("intercept")
    plt.xlim([0.5, 1.5])
    plt.ylim([0.5, 1.5])
    plt.legend()
    plt.title("n = " + str(i))
```



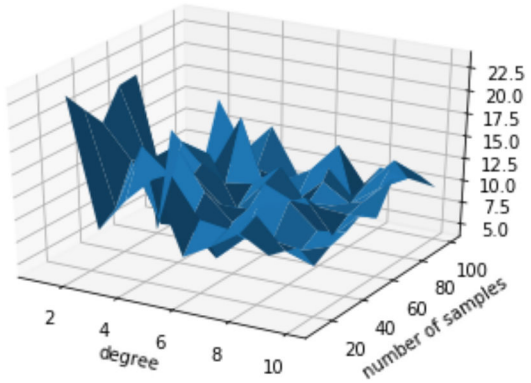


In [42]:
```
# Question 2.i

num = [5, 25, 125, 625]
var = [1, 4, 9, 16]
error_grid = np.zeros( (len(num), len(var)), "float64")

for i in range(len(num)):
    s = int(num[i])
    for j in range(len(var)):
        v = int(var[j])
        X = np.zeros( (s, 2), "float64" )
        Y = np.zeros( (s, 1), "float64" )

        X[:, 0] = np.random.random(s)
        X[:, 1] = np.random.random(s)
        W = np.random.normal(0, v, 2)
        Z = np.random.normal(0, 1, s)
        Y = np.matmul(X, W) + Z
        features, error = ridge_regression_solver_2(X, Y, 1/v)
        error_grid[i, j] = error

num_grid, var_grid = np.meshgrid(num, var)
fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot_surface(num_grid, var_grid, error_grid)
plt.xlabel("number of samples")
plt.ylabel("variance")
plt.show()
```



In [43]:
```
# Problem 4
# Question 4.e

m, c = 1, 1

n = np.linspace(10, 100, num=10, endpoint=True)
D = np.linspace(1, 10, num=10, endpoint=True)

error_grid = np.zeros( (len(D), len(n)), "float64")

for i in range(len(D)):
    d = int(D[i])
    for j in range(len(n)):
        s = int(n[j])
        W = np.random.normal(0, 1, s)
        x_vector = np.random.uniform(-1, 1, s)
        Y = m * x_vector + c + W

        X = np.zeros( (s, d + 1), "float64" )
        for k in range(d + 1):
            X[:, k] = x_vector ** k
        coeffs, y_estimate = linear_regression_solver(X, Y)
        error = np.linalg.norm(y_estimate - Y) ** 2 / j
        error_grid[i, j] = error

D_grid, n_grid = np.meshgrid(D, n)
fig = plt.figure()
ax = fig.gca(projection='3d')
ax.plot_surface(D_grid, n_grid, error_grid)
plt.xlabel("degree")
plt.ylabel("number of samples")
plt.show()
```

```
c:\users\nin\appdata\local\programs\python\python36-32\lib\site-packages\ipykernel
_launcher.py:23: RuntimeWarning: divide by zero encountered in double_scalars
c:\users\nin\appdata\local\programs\python\python36-32\lib\site-packages\numpy\cor
e\numeric.py:1795: RuntimeWarning: invalid value encountered in multiply
  multiply(a1, b2, out=cp0)
c:\users\nin\appdata\local\programs\python\python36-32\lib\site-packages\numpy\cor
e\numeric.py:1800: RuntimeWarning: invalid value encountered in subtract
  cp1 -= tmp
c:\users\nin\appdata\local\programs\python\python36-32\lib\site-packages\mpl_toolk
its\mplot3d\proj3d.py:141: RuntimeWarning: invalid value encountered in true_divid
e
  txs, tys, tzs = vecw[0]/w, vecw[1]/w, vecw[2]/w
```



In [47]:
```
DATA_DIR = "hw03-data"

def condition_number(X, lamb):
    n, m = X.shape
    I = np.identity(m)
    eigs, _ = np.linalg.eig( np.matmul(X.T, X) + lamb * I)
    eig_min, eig_max = np.amin(eigs), np.amax(eigs)
    return eig_max ** 2 / eig_min ** 2


def evaluate(features, X, y):
    n, m = X.shape
    y_estimate = np.zeros( y.shape, "float64" )
    error_list = []
    bias_list = []
    y_estimate_list = []
    for i in range(n):
        y_estimate[i,:] = np.matmul(X[i,:], features)
        error_list.append( np.linalg.norm(y_estimate[i,:]  - y[i,:]) ** 2 )
        bias_list.append( np.linalg.norm(y_estimate[i,:]  - y[i,:]) )
        y_estimate_list.append( np.linalg.norm(y_estimate[i,:]) )
    error = np.mean(error_list)
    bias = np.mean(bias_list)
    variance = np.var(y_estimate_list)
    return error, bias, variance


# Problem 5
# Question 5.a

fid = open(os.path.join(DATA_DIR, "x_train.p"), "rb")
x_train_list = pickle.load(fid, encoding="latin1")
x_train = np.array( [i.flatten() for i in x_train_list], dtype="float64" )
fid.close()

fid = open(os.path.join(DATA_DIR, "y_train.p"), "rb")
y_train = np.array(pickle.load(fid, encoding="latin1"), dtype="float64")
fid.close()

n, m = x_train.shape
print("Question 5.a:")

try:
    policy = np.matmul( np.linalg.inv( np.matmul(x_train.T, x_train) ), np.matmul(x_tr
    print(policy)
    print()
except np.linalg.LinAlgError as e:
    print(str(e))
    print()
```

```
Question 5.a:
Singular matrix
```

In [48]:
```
# Question 5.b:

lamd = [0.1, 1, 10, 100, 1000]
error_list = []
policy_list = []

print("Question 5.b:")
for l in lamd:
    print("l = " + str(l), end="    ")
    policy, error = ridge_regression_solver(x_train, y_train, l)
    policy_list.append(policy)
    print("error = " + str(error))
print()
```

```
Question 5.b:
l = 0.1    error = 8.48330028526e-11
l = 1    error = 5.95495690254e-13
l = 10    error = 1.25678241957e-11
l = 100    error = 1.25561464086e-09
l = 1000    error = 1.24593399019e-07
```

In [49]:
```
# Question 5.c:

x_train_standardized = np.zeros( x_train.shape, "float64")
x_train_standardized = x_train * 2.0 / 255.0 - 1
policy_standardized_list = []

print("Question 5.c:")
for l in lamd:
    print("l = " + str(l), end="    ")
    policy, error = ridge_regression_solver(x_train_standardized, y_train, l)
    policy_standardized_list.append(policy)
    print("error = " + str(error))
```

```
Question 5.c:
l = 0.1    error = 3.25574750613e-07
l = 1    error = 2.91051229057e-05
l = 10    error = 0.0015903814573
l = 100    error = 0.0347731220424
l = 1000    error = 0.254402961468
```

In [4]:
```
# Question 5.d:

print("Question 5.d:")
print("Without standardization: k = " + str(condition_number(x_train, 100)))
print("With standardization: k = " + str(condition_number(x_train_standardized, 100)))
```

```
Question d:
Without standardization: k = (2.77852259576e+15+0j)
With standardization: k = (197781.154336+0j)
```

In [26]:
```
# Question 5.e:

fid = open(os.path.join(DATA_DIR, "x_test.p"), "rb")
x_test_list = pickle.load(fid, encoding="latin1")
x_test = np.array( [i.flatten() for i in x_test_list], dtype="float64" )
x_test_standardized = x_test * 2.0 / 255.0 - 1
fid.close()

fid = open(os.path.join(DATA_DIR, "y_test.p"), "rb")
y_test = np.array(pickle.load(fid, encoding="latin1"), dtype="float64")
fid.close()


print("Question 5.e:")
print("Without standardization:")

for i in range(len(lamd)):
    print("l = " + str(lamd[i]), end="    ")
    error, bias, variance = evaluate(policy_list[i], x_test, y_test)
    print("error = " + str(error), end="    ")
    print("bias = " + str(bias), end="    ")
    print("variance = " + str(variance))
print()

print("With standardization:")

for i in range(len(lamd)):
    print("l = " + str(lamd[i]), end="    ")
    error, bias, variance = evaluate(policy_standardized_list[i], x_test_standardized,
    print("error = " + str(error), end="    ")
    print("bias = " + str(bias), end="    ")
    print("variance = " + str(variance))
```

```
Question 5.e:
Without standardization:
l = 0.1    error = 0.774020351184    bias = 0.794957745631    variance = 0.0926162
974979
l = 1    error = 0.774020464897    bias = 0.794960320813    variance = 0.092616580
61
l = 10    error = 0.77401720455    bias = 0.794959257922    variance = 0.092616413
845
l = 100    error = 0.773983137313    bias = 0.79494690441    variance = 0.09261218
39865
l = 1000    error = 0.773644256139    bias = 0.79482359378    variance = 0.0925700
727932

With standardization:
l = 0.1    error = 0.86807706878    bias = 0.818454305227    variance = 0.10772113
3288
l = 1    error = 0.862102932992    bias = 0.816155480128    variance = 0.106153068
37
l = 10    error = 0.827507615938    bias = 0.801265107685    variance = 0.09595062
74099
l = 100    error = 0.72465308533    bias = 0.759527924947    variance = 0.06709249
28275
l = 1000    error = 0.725014200512    bias = 0.815597260013    variance = 0.048578
9030662
```

In [ ]: