COMP 540 Assignment #1

Yunda Jia
Yu Wu

January 24, 2020

0 Background refresher(30 points)

- Plot the categorical distribution.

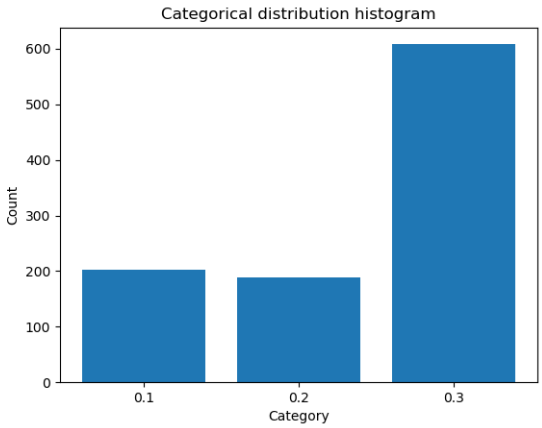


Figure 1: Categorical distribution

- Plot the Univariate normal distribution with mean of and standard deviation of 1.

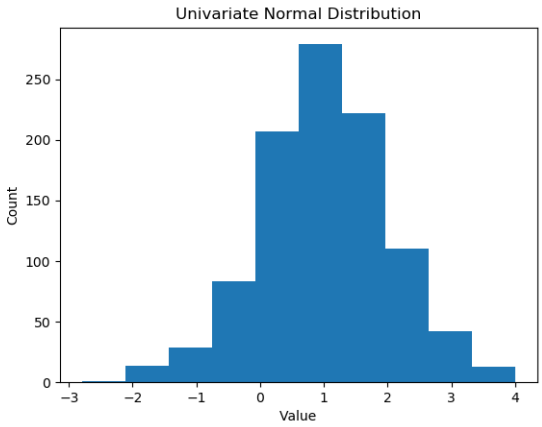


Figure 2: Univariate Normal Distribution

- Produce a scatter plot of the samples for a 2-D Gaussian.

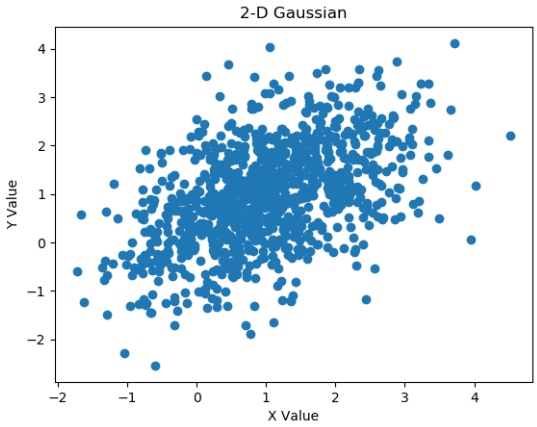


Figure 3: Univariate Normal Distribution

- Test mixture sampling code Code can be seen in sampler.py. Mixture Gaussian plot is shown below

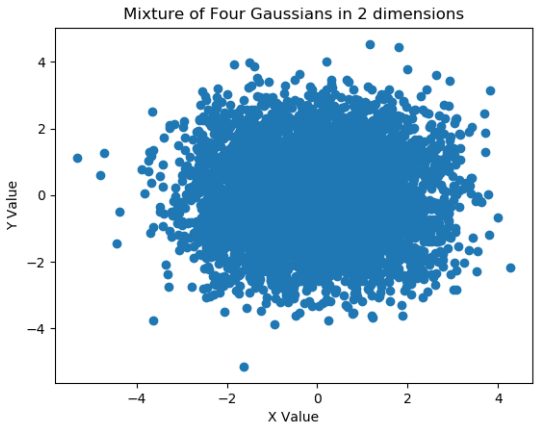


Figure 4: Univariate Normal Distribution

- Prove that the sum of two independent Poisson random variables is also a Poisson random variable. Suppose $X \sim \mathcal{P}(\lambda)$ and $Y \sim \mathcal{P}(\mu)$. Now Prove that $X + Y \sim \mathcal{P}(\lambda + \mu)$.

$$
\begin{aligned}
P(X+Y=k) &= \sum_{i=0}^{k} P(X+Y=k, X=i) \\
&= \sum_{i=0}^{k} P(Y=k-i, X=i) \\
&= \sum_{i=0}^{k} P(Y=k-i)P(X=i) \\
&= \sum_{i=0}^{k} e^{-\mu}\frac{\mu^{k-i}}{(k-i)!}e^{-\lambda}\frac{\lambda^{i}}{i!} \\
&= e^{-(\mu+\lambda)}\frac{1}{k!}\sum_{i=0}^{k}\frac{k!}{i!(k-i)!}\mu^{k-i}\lambda^{i} \\
&= e^{-(\mu+\lambda)}\frac{1}{k!}\sum_{i=0}^{k}\binom{k}{i}\mu^{k-i}\lambda^{i} \\
&= \frac{(\mu+\lambda)^{k}}{k!}\cdot e^{-(\mu+\lambda)}
\end{aligned}
$$

So $X+Y \sim \mathcal{P}(\lambda+\mu)$.

- Find α, μ_1 and σ_1.
  We have X_0 and X_1 be continuous random variables. If

$$
\begin{aligned}
p(X_0 = x_0) &= \alpha_0 e^{-\frac{(x_0-\mu_0)^2}{2\sigma_0^2}} \\
P(X_1 = x_1 | X_0 = x_0) &= \alpha_1 e^{-\frac{(x_1-x_0)^2}{2\sigma^2}}
\end{aligned}
$$

$$
\begin{aligned}
p(X_1 = x_1) &= \int P(X_1 = x_1 | X_0 = x_0)\cdot p(X_0 = x_0)dx_0 \\
&= \alpha_0\alpha_1 \int e^{-\frac{\sigma^2(x_0-\mu_0)^2+\sigma_0^2(x_1-x_0)^2}{2\sigma_0^2\sigma^2}} dx_0 \\
&= \alpha_0\alpha_1 \int e^{-\frac{(\sigma^2+\sigma_0^2)x_0^2-2(\sigma^2\mu_0+\sigma_0^2x_1)x_0+\sigma^2\mu_0^2+\sigma_0^2x_1^2}{2\sigma_0^2\sigma^2}} dx_0 \\
&= \alpha_0\alpha_1 \int e^{-\frac{1}{2\sigma_0^2\sigma^2}[(\sqrt{\sigma^2+\sigma_0^2}x_0-\frac{-\sigma^2\mu_0+\sigma_0^2x_1}{\sqrt{\sigma^2+\sigma_0^2}})^2+\sigma^2\mu_0^2+\sigma_0^2x_1^2-\frac{\sigma^2\mu_0^2+\sigma_0^2x_1^2}{\sigma^2+\sigma_0^2}]} dx_0
\end{aligned}
$$

Since

$$
\int \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx = 1
$$

$$
\begin{aligned}
p(X_1 = x_1) &= \frac{\alpha\alpha_0\sqrt{2\pi}\sigma\sigma_0}{\sqrt{\sigma^2+\sigma_0^2}} e^{-\frac{1}{2\sigma_0^2\sigma^2}(\sigma^2\mu_0^2+\sigma_0^2x_1^2-\frac{\sigma^2\mu_0^2+\sigma_0^2x_1^2}{\sigma^2+\sigma_0^2})} \\
&= \alpha e^{-\frac{(x_1-\mu_0)^2}{2(\sigma^2+\sigma_0^2)}}
\end{aligned}
$$

Thus we can solve that:

$$\begin{aligned}\alpha &= \frac{\alpha\alpha_0\sqrt{2\pi}\sigma\sigma_0}{\sqrt{\sigma^2+\sigma_0^2}}\\ \mu_1 &= \mu_0\\ \sigma_1 &= \sqrt{\sigma^2+\sigma_0^2}\end{aligned}$$

- Show that if $P(A|B,C) > P(A|B)$ then $P(A|B,C^C) < P(A|B)$

$$P(A|B,C) = \frac{P(A\bigcap B\bigcap C)}{P(B\bigcap C)}$$

$$\begin{aligned}\frac{P(A\bigcap B\bigcap C)}{P(B\bigcap C)} &> \frac{P(A\bigcap B)}{P(B)}\\ \frac{P(C|A\bigcap B)}{P(C|B)} &> 1\\ P(C|A\bigcap B) &> P(C|B)\\ 1-P(C|A\bigcap B) &< 1-P(C|B)\\ P(C^C|A\bigcap B) &< P(C^C|B)\\ \frac{P(C^C|A\bigcap B)}{P(C^C|B)} &< 1\end{aligned}$$

$$\begin{aligned}P(A|B,C^C) &= \frac{P(A\bigcap B\bigcap C^C)}{P(B\bigcap C^C)}\\ &= \frac{P(C^C|A\bigcap B)P(A\bigcap B)}{P(C^C|B)P(B)}\\ &= \frac{P(C^C|A\bigcap B)}{P(C^C|B)P(B)}P(A|B)\\ &< P(A|B)\end{aligned}$$

Thus, if $P(A|B,C) > P(A|B)$ then $P(A|B,C^C) < P(A|B)$.

- Consider the vectors $\boldsymbol{u} = \begin{bmatrix}1 & 2\end{bmatrix}^{\mathrm{T}}$ and $\boldsymbol{v} = \begin{bmatrix}2 & 3\end{bmatrix}^{\mathrm{T}}$. Define the matrix $\boldsymbol{M} = \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}}$. Compute the eigenvalues and eigenvectors of $\boldsymbol{M}$.

$$\begin{aligned}\boldsymbol{M} &= \boldsymbol{u}\boldsymbol{v}^{\mathrm{T}}\\ &= \begin{bmatrix}1 & 2\end{bmatrix}^{\mathrm{T}}\cdot\begin{bmatrix}2 & 3\end{bmatrix}\\ &= \begin{bmatrix}2 & 3\\ 4 & 6\end{bmatrix}\end{aligned}$$

$$\left|\lambda\boldsymbol{I}-\boldsymbol{M}\right| = \begin{vmatrix}\lambda-2 & -3\\ -4 & \lambda-6\end{vmatrix} = 0$$

Then we can solve it as

$$
\begin{aligned}
(\lambda-2)(\lambda-6)-12&=0\\
\lambda^2-8\lambda&=0\\
\lambda(\lambda-8)&=0\\
\lambda_1=0,\lambda_2&=8
\end{aligned}
$$

Let $\lambda=0$:

$$
\begin{aligned}
(\lambda\boldsymbol{I}-\boldsymbol{M})\cdot\begin{bmatrix}x_1 & x_2\end{bmatrix}^{\mathrm{T}}&=0\\
\begin{bmatrix}-2 & -3\\ -4 & -6\end{bmatrix}\cdot\begin{bmatrix}x_1\\ x_2\end{bmatrix}&=\begin{bmatrix}0\\0\end{bmatrix}\\
-2x_1-3x_2&=0
\end{aligned}
$$

Let $x_1=3$, Then $x_2=-2$. So eigenvector is $\begin{bmatrix}3 & -2\end{bmatrix}^{\mathrm{T}}$.
Let $\lambda=8$:

$$
\begin{aligned}
(\lambda\boldsymbol{I}-\boldsymbol{M})\cdot\begin{bmatrix}x_1 & x_2\end{bmatrix}^{\mathrm{T}}&=0\\
\begin{bmatrix}6 & -3\\ -4 & 2\end{bmatrix}\cdot\begin{bmatrix}x_1\\ x_2\end{bmatrix}&=\begin{bmatrix}0\\0\end{bmatrix}\\
6x_1-3x_2&=0\\
-4x_1+2x_2&=0
\end{aligned}
$$

Let $x_1=1$, Then $x_2=2$. So eigenvector is $\begin{bmatrix}1 & 2\end{bmatrix}^{\mathrm{T}}$.
Thus, when eigenvalue $\lambda=0$, eigenvector is $\begin{bmatrix}3 & -2\end{bmatrix}^{\mathrm{T}}$, when eigenvalue $\lambda=8$, eigenvector is $\begin{bmatrix}1 & 2\end{bmatrix}^{\mathrm{T}}$.

- Show that if A is positive semi-definite, then all eigenvalues of A are non-negative.
  Definition of an eigenvalue and eigenvector is:

$$A\boldsymbol{v}=\lambda\boldsymbol{v}$$

If A is positive semi-definite, $\boldsymbol{x}^TA\boldsymbol{x}\geq 0$ for all $\boldsymbol{x}$.

$$\boldsymbol{v}^TA\boldsymbol{v}=\boldsymbol{v}^T\boldsymbol{v}\lambda$$

Since $\boldsymbol{v}^T\boldsymbol{v}$ is necessarily a positive number, in order for $\boldsymbol{v}^TA\boldsymbol{v}$ to be greater than or equal to 0, λ must be $\lambda\geq 0$.

- Provide one example for each of the following cases.
  As for $(A+B)^2\neq A^2+2AB+B^2$. Suppose $A=\begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}$ and $B=\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}$. Since $A\cdot B=\begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}$ and $B\cdot A=\begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}$.
  So $(A+B)^2=A^2+AB+BA+B^2\neq A^2+2AB+B^2$.
  As for $AB=0, A\neq 0, B\neq 0$, Suppose $A=\begin{bmatrix}0 & 1\\ 0 & 0\end{bmatrix}$ and $B=\begin{bmatrix}1 & 0\\ 0 & 0\end{bmatrix}$.

- Show that $\boldsymbol{A}$ is orthogonal.
  Given $\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u} = 1$ and $\boldsymbol{A} = \boldsymbol{I} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}$.

$$\begin{aligned}\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A} &= (\boldsymbol{I} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}})^{\mathrm{T}}(\boldsymbol{I} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}) \\ &= (\boldsymbol{I} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}})(\boldsymbol{I} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}) \\ &= \boldsymbol{I} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}} + 4\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}} \\ &= \boldsymbol{I} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}} - 2\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}} + 4\boldsymbol{u}\boldsymbol{u}^{\mathrm{T}} \\ &= \boldsymbol{I}\end{aligned}$$

  So $\boldsymbol{A}$ is orthogonal.

- Prove the following assertions.
  –As for $f(x) = x^3$ for $x \geq 0$.

$$\begin{aligned}f(x) &= x^3 \\ f^{''}(x) &= 6x\end{aligned}$$

  Since $x \geq 0$, $f^{''}(x) = 6x \geq 0$. Thus $f(x) = x^3$ is convex for $x \geq 0$.
  –As for $f(x_1, x_2) = max(x_1, x_2)$. Let $\boldsymbol{x} = (x_1, x_2)$ and $\boldsymbol{y} = (y_1, y_2)$ and $\lambda \in [0, 1]$.

$$\begin{aligned}f(\lambda\boldsymbol{x} + (1-\lambda)\boldsymbol{y}) &= f(\lambda x_1 + (1-\lambda)y_1, \lambda x_2 + (1-\lambda)y_2) \\ &= max(\lambda x_1 + (1-\lambda)y_1, \lambda x_2 + (1-\lambda)y_2) \\ &\leq max(\lambda x_1, \lambda x_2) + max((1-\lambda)y_1, (1-\lambda)y_2) \\ &= \lambda max(x_1, x_2) + (1-\lambda)max(y_1, y_2) \\ &= \lambda f(\boldsymbol{x}) + (1-\lambda)f(\boldsymbol{y})\end{aligned}$$

  So $f(x_1, x_2) = max(x_1, x_2)$ is convex on R^2.
  –As for function $f + g$. If univariate functions f and g are convex on S, then $f^{''}(x) \geq 0$ and $g^{''}(x) \geq 0$.
  So $(f+g)^{''}(x) = f^{''}(x) + g^{''}(x) \geq 0$. Thus if univariate functions f and g are convex on S, then $f + g$ is convex on S.
  –As for fg. If univariate functions f and g are convex and non-negegative on S.

$$\begin{aligned}(fg)^{''}(x) &= (f^{'}g + fg^{'})^{'}(x) \\ &= (f^{''}g + f^{'}g^{'} + f^{'}g^{'} + fg^{''})(x) \\ &= (f^{''}g + 2f^{'}g^{'} + fg^{''})(x)\end{aligned}$$

  Since f and g have their minimum within S at the same point. Before the minimum point, both f and g are decreasing. After the minimum point, both f and g are increasing. So $f^{'}g^{'} \geq 0$, $f^{''} \geq 0$, $g^{''} \geq 0$, $f \geq 0$ and $g \geq 0$. Thus $(fg)^{''}(x) = (f^{''}g + 2f^{'}g^{'} + fg^{''})(x) \geq 0$. Then fg is convex on S.

- Find the highest entropy of categorical distribution.
  The entropy of a categorical distribution on K values is defined as

$$H(p) = -\sum_{i=1}^{K} p_i log(p_i)$$

  The probability and constrains are defined below:

$$P(X = x_i) = p_i \quad for \quad i = 1, 2, ..., K$$

$$s.p. \begin{cases} \sum_{i=1}^{K} p_i = 1 \\ p_i \geq 0 \quad for \quad i = 1, 2, ..., K \end{cases}$$

Constrain function is

$$\varphi(p_i) = \sum_{i=1}^{K} p_i - 1 = 0$$

Define Lagrange multipliers:

$$\begin{aligned} \mathcal{L} &= H(p) + \lambda\varphi(p_i) \\ &= -\sum_{i=1}^{K} p_i log(p_i) + \lambda(\sum_{i=1}^{K} p_i - 1) \end{aligned}$$

To find the highest entropy, we should find the point where derivative is 0.

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial p_i} &= -log(p_i) - 1 + \lambda = 0 \\ p^* &= e^{\lambda - 1} \end{aligned}$$

Thus when all $p_i, i = 1, 2, ..., K$ are equally equal to $e^{\lambda-1}$, the categorical distribution has the highest entropy.

1 Locally weighted linear regression(20 points)

- Find an appropriate diagonal matrix W.

$$J(\theta) = (X\theta - y)^T W (X\theta - y)$$

Let W be

$$W = \begin{bmatrix} \frac{1}{2}w^{(1)} & & 0 \\ & \frac{1}{2}w^{(2)} & \\ 0 & & \ddots \\ & & & \frac{1}{2}w^{(i)} \end{bmatrix}$$

X is the $m \times d$ input matrix and y is a $m \times 1$ vector.

$$X = \begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & \cdots & x_d^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ x_1^{(m)} & x_2^{(m)} & x_3^{(m)} & \cdots & x_d^{(m)} \end{bmatrix}$$

$$y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

$$\theta = \begin{bmatrix} \theta_{(1)} \\ \theta_{(2)} \\ \vdots \\ \theta_{(d)} \end{bmatrix}$$

$$X\theta - y = \begin{bmatrix} (\theta_1 x_1^{(1)} + \theta_2 x_2^{(1)} + \cdots + \theta_d x_d^{(1)}) - y^{(1)} \\ (\theta_1 x_1^{(2)} + \theta_2 x_2^{(2)} + \cdots + \theta_d x_d^{(2)}) - y^{(2)} \\ \vdots \\ (\theta_1 x_1^{(m)} + \theta_2 x_2^{(m)} + \cdots + \theta_d x_d^{(m)}) - y^{(m)} \end{bmatrix}$$

$$W(X\theta - y) = \begin{bmatrix} \frac{1}{2} w^{(1)} \times ((\theta_1 x_1^{(1)} + \theta_2 x_2^{(1)} + \cdots + \theta_d x_d^{(1)}) - y^{(1)}) \\ \frac{1}{2} w^{(2)} \times ((\theta_1 x_1^{(2)} + \theta_2 x_2^{(2)} + \cdots + \theta_d x_d^{(2)}) - y^{(2)}) \\ \vdots \\ \frac{1}{2} w^{(m)} \times ((\theta_1 x_1^{(m)} + \theta_2 x_2^{(m)} + \cdots + \theta_d x_d^{(m)}) - y^{(m)}) \end{bmatrix}$$

$$\begin{aligned} (X\theta - y)^T W(X\theta - y) = &\frac{1}{2} w^{(1)} \times ((\theta_1 x_1^{(1)} + \theta_2 x_2^{(1)} + \cdots + \theta_d x_d^{(1)}) - y^{(1)})^2 \\ &+ \frac{1}{2} w^{(2)} \times ((\theta_1 x_1^{(2)} + \theta_2 x_2^{(2)} + \cdots + \theta_d x_d^{(2)}) - y^{(2)})^2 + \cdots \\ &+ \frac{1}{2} w^{(m)} \times ((\theta_1 x_1^{(m)} + \theta_2 x_2^{(m)} + \cdots + \theta_d x_d^{(m)}) - y^{(m)})^2 \end{aligned}$$

So $J(\theta) = (X\theta - y)^T W(X\theta - y)$ can be written in the from $J(\theta) = (X\theta - y)^T W(X\theta - y)$ when choosing W as above.

- Solve for θ

$$\begin{aligned} J(\theta) &= (X\theta - y)^T W(X\theta - y) \\ &= ((X\theta)^T - y^T) W(X\theta - y) \\ &= ((X\theta)^T W - y^T W)(X\theta - y) \\ &= \theta^T X^T W X \theta - 2\theta^T X^T W y + y^T W y \end{aligned}$$

$$\begin{aligned} \nabla_\theta J(\theta) &= 2X^T W X \theta - 2X^T W y = 0 \\ \theta &= (X^T W X)^{-1} X^T W y \\ &= (X^T X)^{-1} X^T y \end{aligned}$$

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

- Write down an algorithm for calculating θ by batch gradient descent for locally weighted linear regression.
We know that $\theta = (X^T X)^{-1} X^T y$.
Algorithm is

Algorithm 1: Calculating θ by batch gradient descent

Input: $\boldsymbol{x}$
Output: θ
for $i < 10000$ **do**
  $\theta_j \leftarrow \theta_j - \alpha \sum_i w^{(i)} (\sum_k (\theta_k x_x^{(i)}) - y^{(i)}), j = 0$
  $\underline{\theta_j} \leftarrow \theta_j - \alpha \sum_i w^{(i)} (\sum_k (\theta_k x_x^{(i)}) - y^{(i)}) x_j^{(i)}, j \neq 0$

Locally weighted linear regression is a non-parametric method.

2 Properties of the linear regression estimator(10 points)

- Show that $E[\theta] = \theta^*$

$$\begin{aligned}\theta &= (X^T X)^{-1} X^T y \\ &= (X^T X)^{-1} X^T (X\theta^* + \epsilon) \\ &= (X^T X)^{-1} X^T X\theta^* + (X^T X)^{-1} X^T \epsilon\end{aligned}$$

$$\begin{aligned}E(\theta) &= E((X^T X)^{-1} X^T X\theta^* + (X^T X)^{-1} X^T \epsilon) \\ &= E((X^T X)^{-1} X^T X\theta^*) + E(X^T X)^{-1} X^T \epsilon) \\ &= \theta^* + (X^T X)^{-1} X^T E(\epsilon) \\ &= \theta^*\end{aligned}$$

- Show the variance of the least squares estimator is $Var(\theta) = (X^T X)^{-1}\sigma^2$. Suppose $\boldsymbol{b}$ is an estimator of θ

$$\begin{aligned}Var(\theta) &= E[(\boldsymbol{b} - \theta)(\boldsymbol{b} - \theta)^T | X] \\ &= E[(X^T X)^{-1} X^T \epsilon\epsilon^T X (X^T X)^{-1} | X] \\ &= (X^T X)^{-1} X^T E[\epsilon\epsilon^T | X] X (X^T X)^{-1} \\ &= (X^T X)^{-1}\sigma^2\end{aligned}$$

3 Implementing linear regression and regularized linear regression(90 points)

3.1 Implementing linear regression(45 points)

- A1: Computing the cost function $J(\theta)$
  See the implementation for the *loss* function in the file *linear_regressor.py*.
- A2: Implementing gradient descent
  See the implementation for the *train* function in the file *linear_regressor.py*.

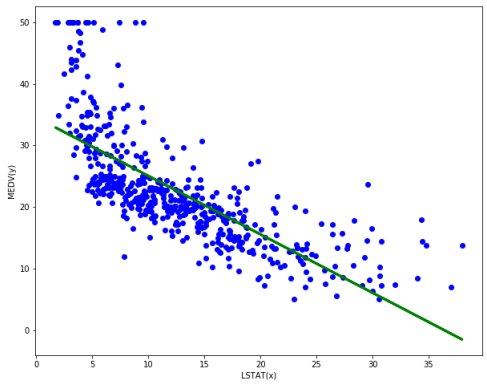


Figure 5: Fitting a linear model

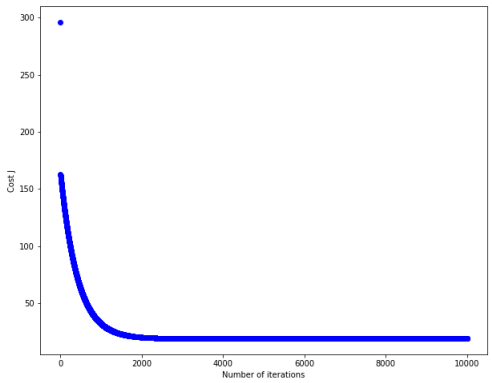


Figure 6: Convergence of gradient descent

- A3: Predicting on unseen data
  For lower status percentage = 5, we predict a median home value of 298034.49
  For lower status percentage = 50, we predict a median home value of -129482.13

- B1: Feature normalization
  See the implementation for the $feature_normalize$ function in the file *utils.py*.
- B2: Loss function and gradient descent
  See the implementation for the *train* and *loss* function in the file *linear_regressor_multi.py*.

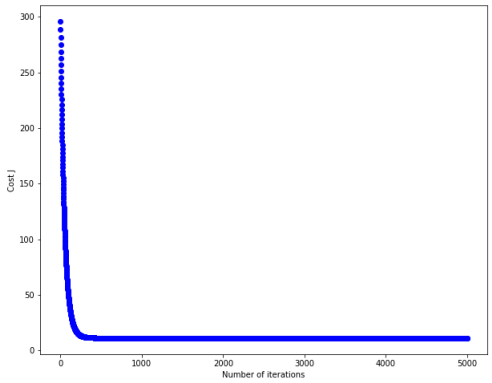
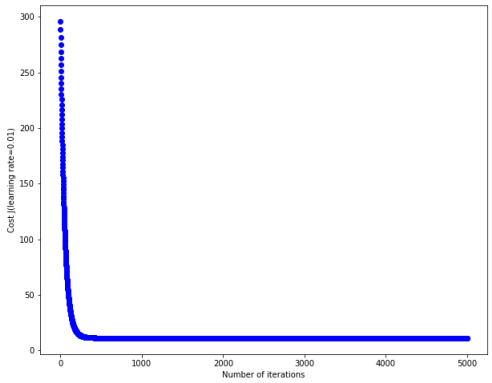


Figure 7: Convergence of gradient descent with multiple variables

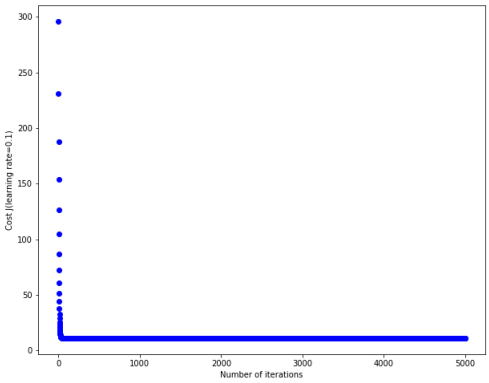
- B3: Making predictions on unseen data
  For average home in Boston suburbs, we predict a median home value of 230467.11

- B4: Normal equations
  See the implementation for the $normal_eqn$ function in the file *linear_regressor_multi.py*.
  Theta computed by direct solution is: $[3.64594884e+01 - 1.08011358e-01 - 4.64204584e-02 - 2.05586264e-02 - 2.68673382e+00 - 1.77666112e+01 + 3.80986521e+00 - 6.92224640e-04 - 1.47556685e+00 + 3.06049479e-01 - 1.23345939e-02 - 9.52747232e-01 - 9.31168327e-03 - 5.24758378e-01]$
  For average home in Boston suburbs, we predict a median home value of 230406.54.

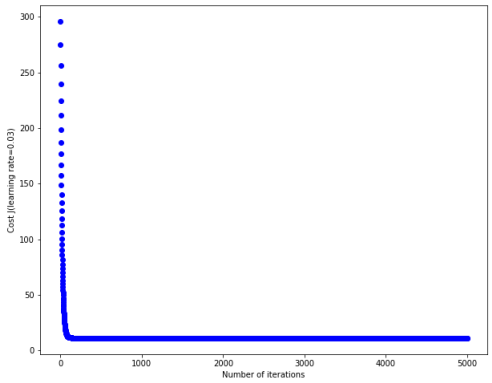
- B5: Exploring convergence of gradient descent



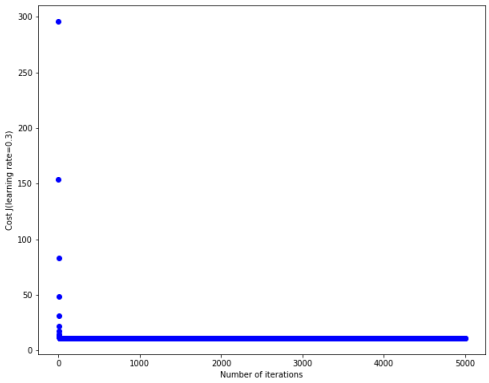
((a)) learning rate=0.01



((b)) learning rate=0.03



((c)) learning rate=0.1

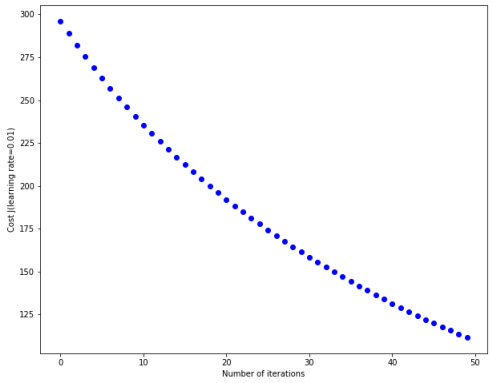


((d)) learning rate=0.3

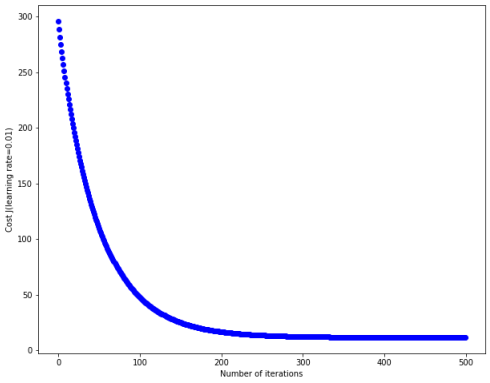
Figure 8: Convergence of gradient descent with multiple variables according to different learning rates

According to the above figures, when learning $rate = 0.01$ is a good choice.

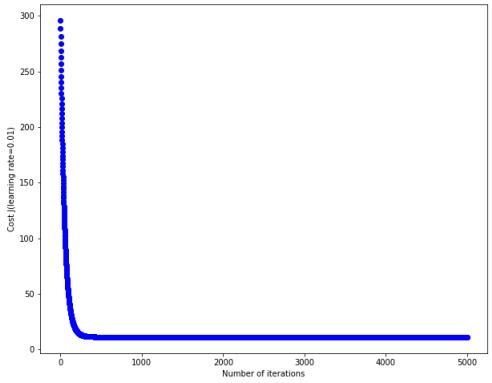
When learning rate = 0.01, chosing different iteration times.



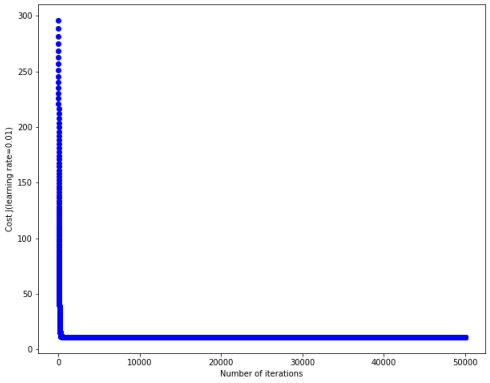
((a)) iteration time = 50



((b)) iteration time = 500



((c)) iteration time = 5000



((d)) iteration time = 50000

Figure 9: Convergence of gradient descent with multiple variables according to different iteration times

According to the above figures, iteration time is 500 is a good choice.

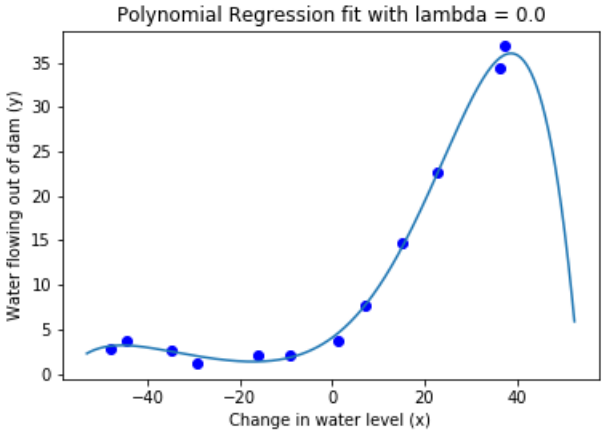
3.2 Implementing regularized linear regression(45 points)

- A1: Regularized linear regression cost function
  See the implementation for the *loss* function in the class *Reg_LinearRegression_SquaredLoss* in the file *reg_linear_regressor_multi.py*.

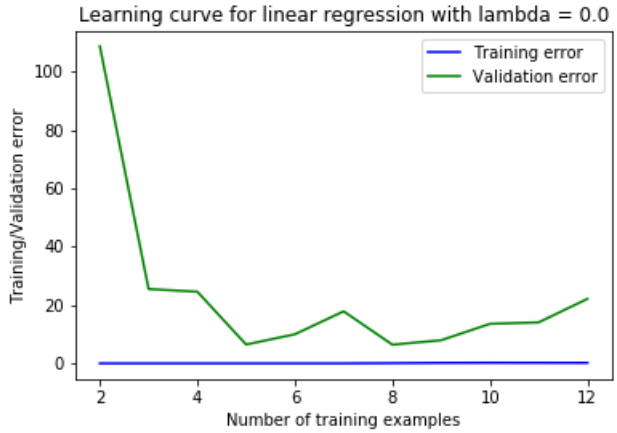
- A2: Gradient of the Regularized linear regression cost function
  See the implementation for the *grad_loss* function in the class *Reg_LinearRegression_SquaredLoss* in the file *reg_linear_regressor_multi.py*.

- A3: Learning curves

- A4: Adjusting the regularization parameter

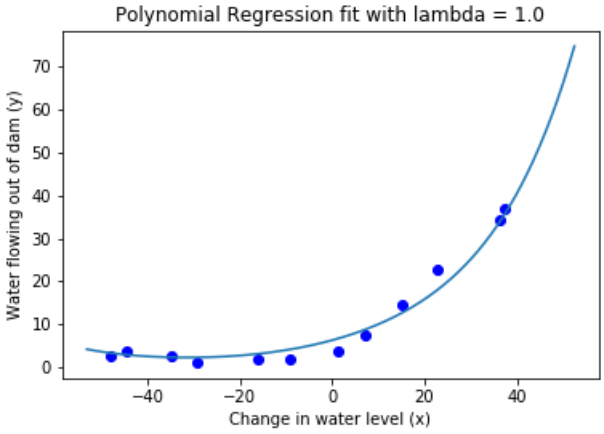


((a)) Polynomial Regression fit

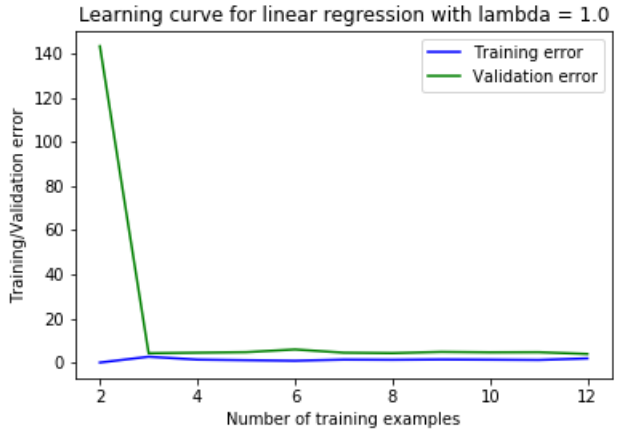


((b)) Learning curve for linear regression

Figure 10: $\lambda = 0.0$

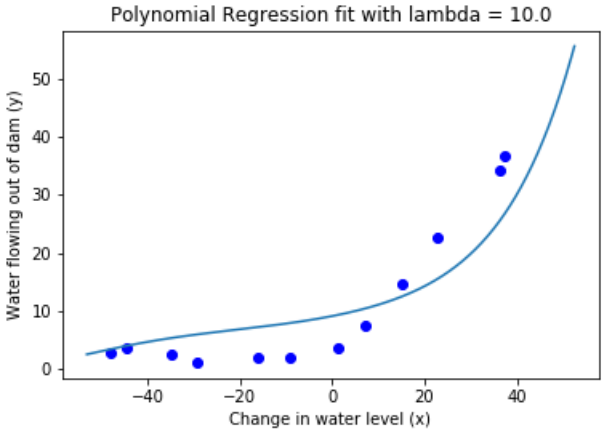


((a)) Polynomial Regression fit

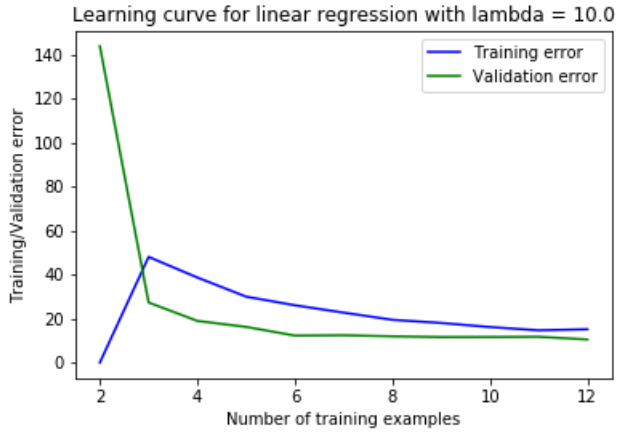


((b)) Learning curve for linear regression

Figure 11: $\lambda = 1.0$

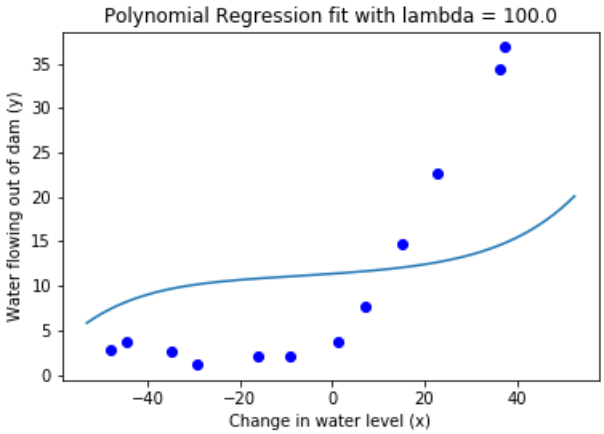


((a)) Polynomial Regression fit

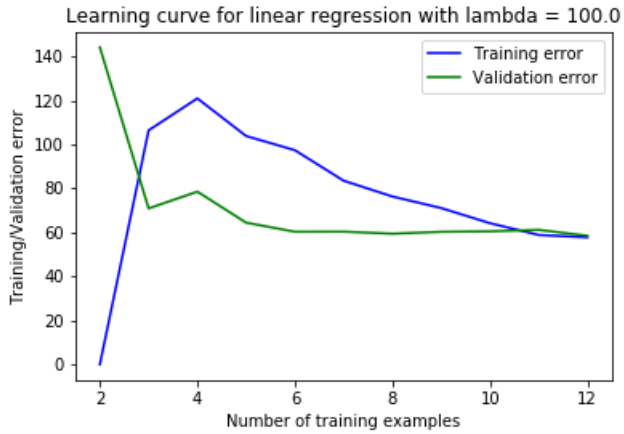


((b)) Learning curve for linear regression

Figure 12: $\lambda = 10.0$



((a)) Polynomial Regression fit



((b)) Learning curve for linear regression

Figure 13: $\lambda = 100.0$

- A5: Selecting λ using a validation set According to the figure above, the best λ is 3.

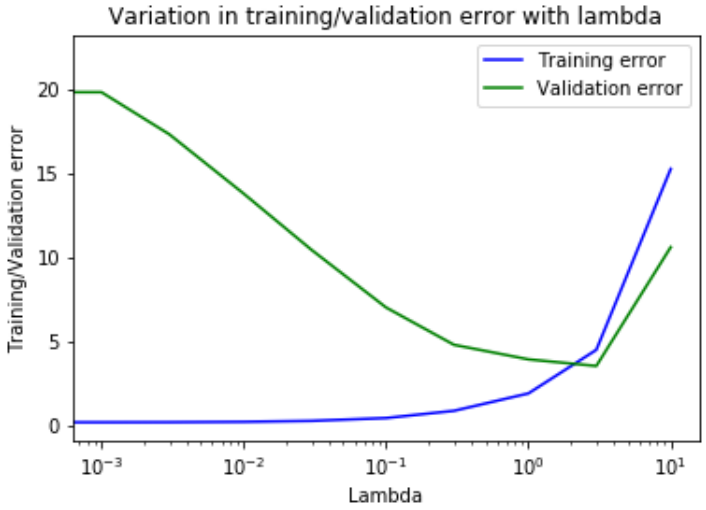


Figure 14: Fitting a linear model

- A6: Computing test set error
  Test error is 5.944297830865587.

- A7: Plotting learning curves with randomly selected examples

- A8: Comparing ridge regression and lasso regression models
  Lasso regression converges faster than Ridge regression.

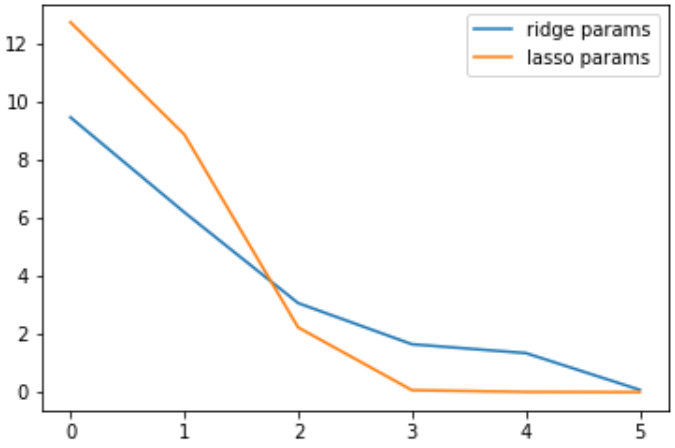


Figure 15: Fitting a linear model

4 Extra Credits

- The lowest achievable error on the test set with $\lambda = 0$
  Test error is 12.930601133114878 when $\lambda = 0$.

- Select the best value for λ and report the test set error with the best λ.
  The best $\lambda = 30.0$. Test error at $\lambda = 30.0$ is 13.256266000933937.

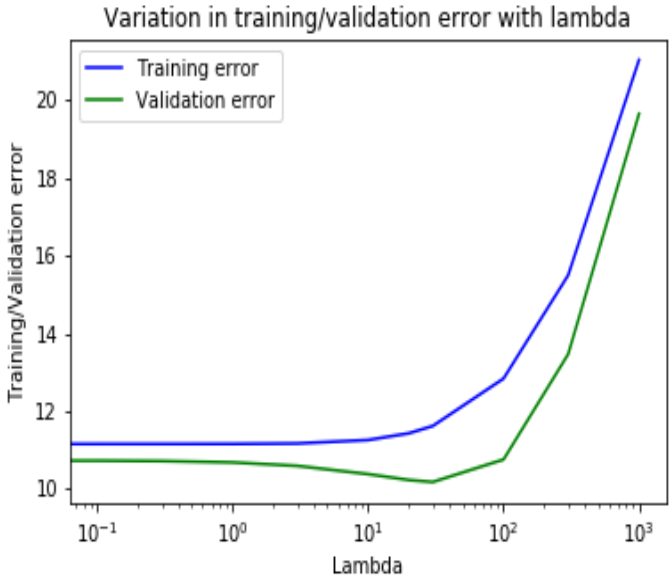


Figure 16: Fitting a linear model

- What is the test set error with quadratic features with the best λ chosen with the validation set?
  The best $\lambda = 30.0$. Test error at $\lambda = 30.0$ is 4.747514558496794.

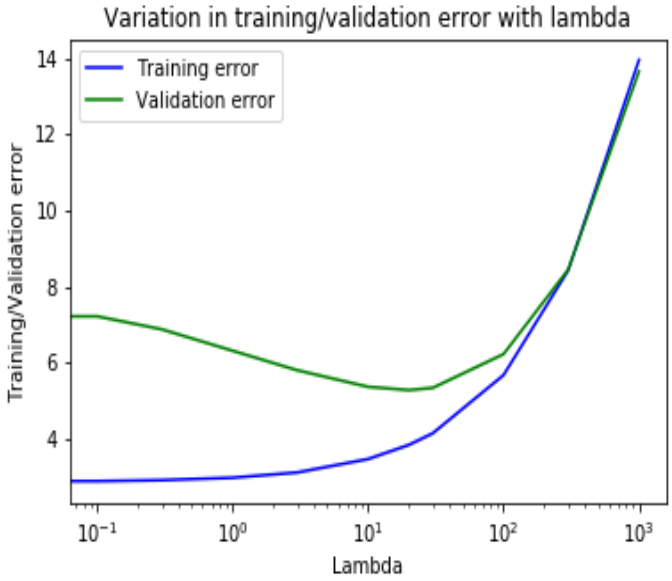


Figure 17: Fitting a linear model

- What is the test set error with cubic features with the best λ chosen with the validation set? The best $\lambda = 30.0$. Test error at $\lambda = 30.0$ is 4.767539493628556.

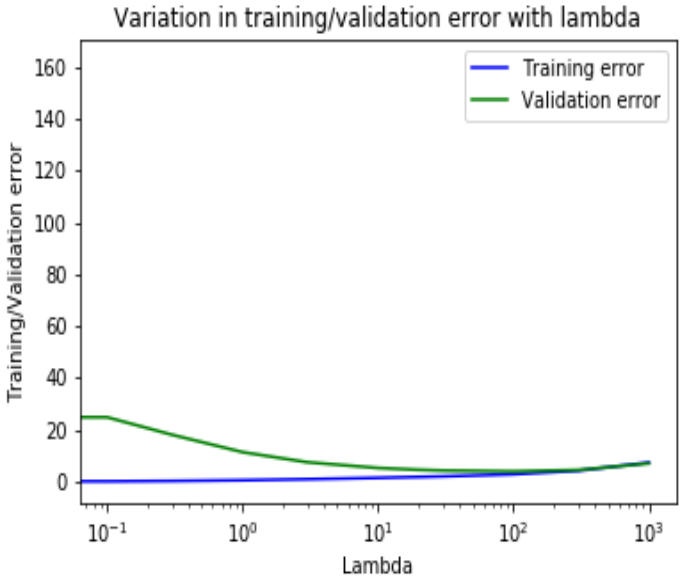


Figure 18: Fitting a linear model