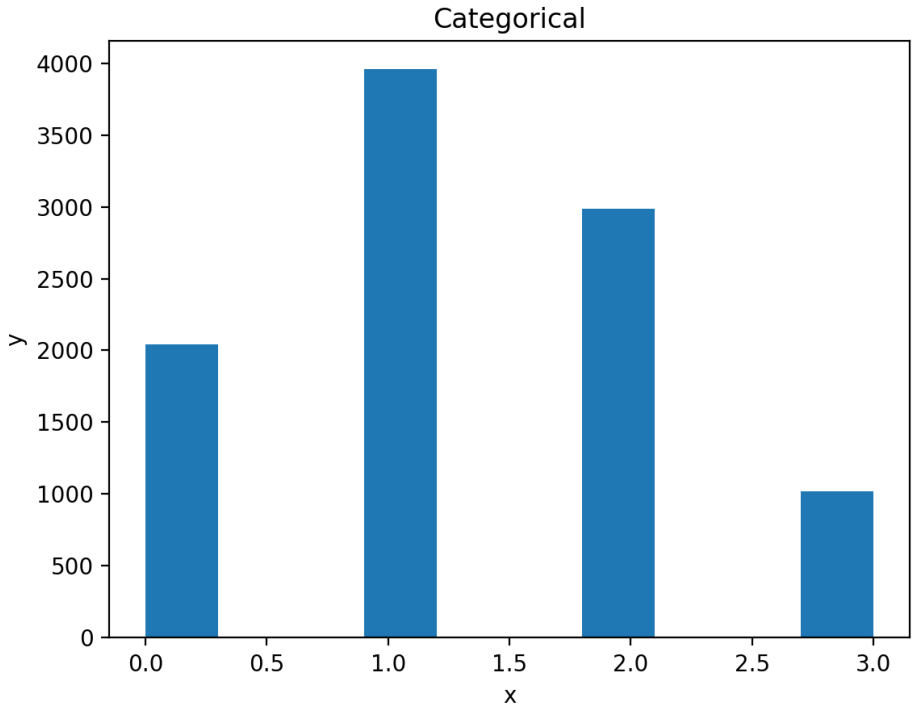
Assignment 0 （Writeup）

Created By:

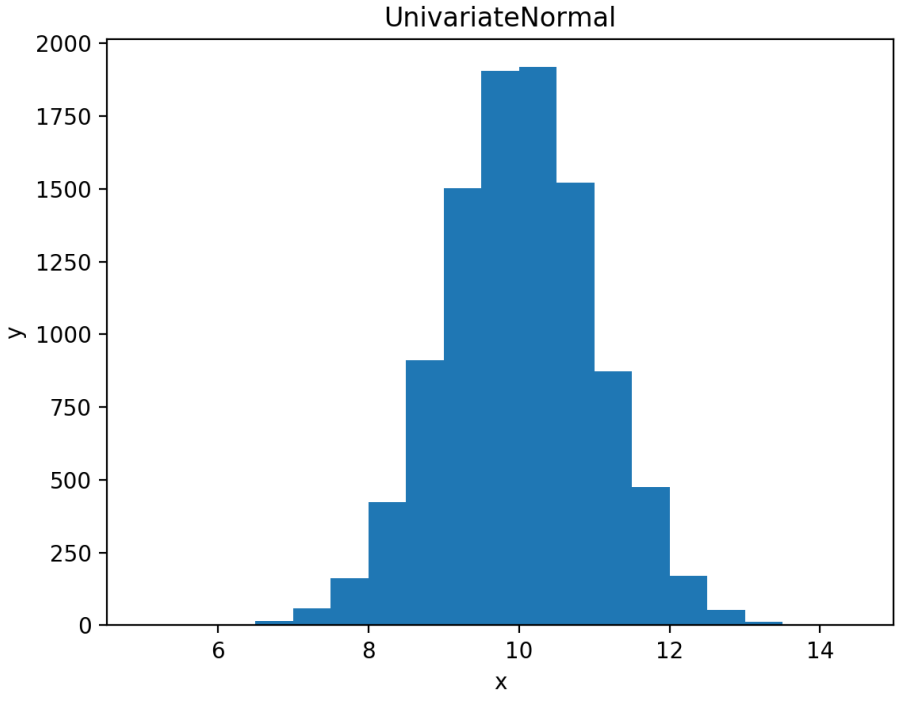
Group: Chengyin Liu(cl93), Ran Jin(Oliver)(rj23)

Jan 19, 2018
COMP 540 – Statistical Machine Learning
Rice University

Problem 0:

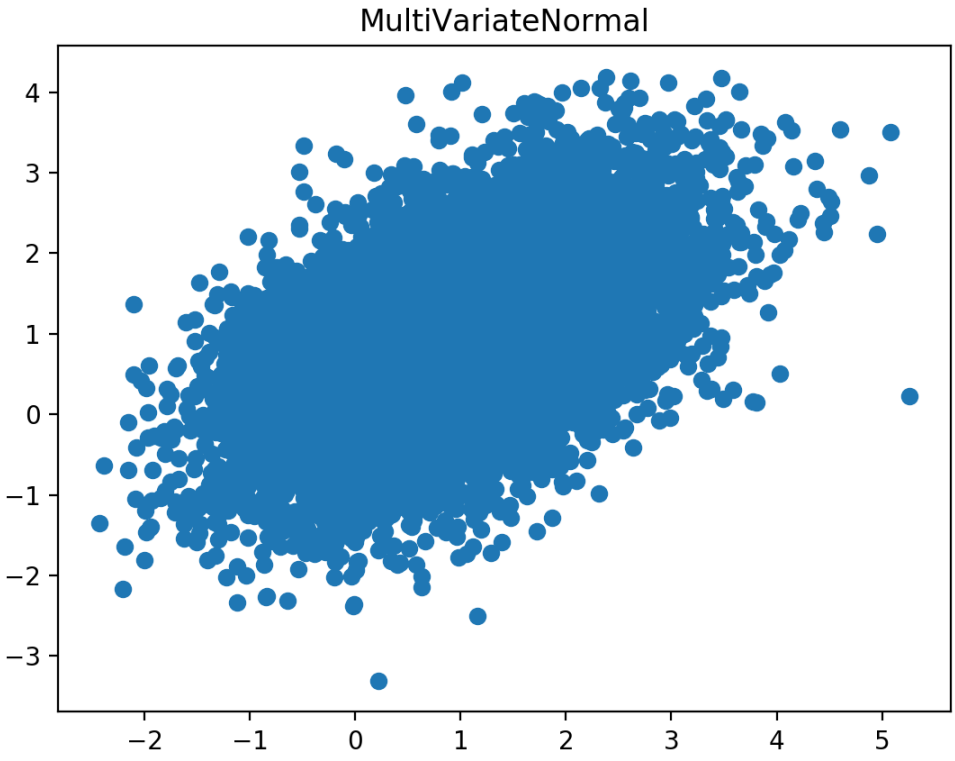

Categorical
4000
3500
3000
2500
2000
1500
1000
500
0
y
0.0
0.5
1.0
1.5
2.0
2.5
3.0
x

a.


UnivariateNormal
2000
1750
1500
1250
1000
750
500
250
0
y
6
8
10
12
14
x

b.

c.



COMP540

Problem 0:

1) Prove that the sum of two independent Poisson random variables is also a Poisson random variable

Solution: Let X & Y be 2 independent Poisson random variables w/ λ_1, λ_2 respectively. We WTS that for $Z = X+Y$ w/ $(\lambda_1 + \lambda_2)$

By def. of Probability generating functions of Poisson, We have

$$P(X=x) = \frac{e^{-\lambda_1} \cdot \lambda_1^x}{x!}$$

$$P(Y=y) = \frac{e^{-\lambda_2} \cdot \lambda_2^y}{y!}$$

Since X & Y are independent

$$\Rightarrow P(Z=z=X+y) = \sum_{x=0}^{z} P(X=x) P(Y=z-x)$$

$$= \sum_{x=0}^{z} \frac{e^{-\lambda_1} \cdot \lambda_1^x}{x!} \cdot \frac{e^{-\lambda_2} \cdot \lambda_2^{z-x}}{(z-x)!}$$

$$= e^{-(\lambda_1+\lambda_2)} \cdot \sum_{x=0}^{z} \frac{\lambda_1^x}{x!} \cdot \frac{\lambda_2^{z-x}}{(z-x)!}$$

By binomial formula, $(x+y)^z = \sum_{a=0}^{z} \binom{z}{a} x^a y^{z-a}$,

$$\Rightarrow \sum_{x=0}^{z} \frac{\lambda_1^x}{x!} \cdot \frac{\lambda_2^{z-x}}{(z-x)!} = \frac{(\lambda_1+\lambda_2)^z}{z!}$$

$\Rightarrow$ Hence the proof.

2) Let X_0, X_1 be continuous random variables, WTS

$P(X_0 = x_0) = \alpha_0 e^{-\frac{(x_0-\mu_0)^2}{2\sigma_0^2}}$

If $P(X_1 = x_1 \mid X_0 = x_0) = \alpha e^{-\frac{(x_1-x_0)^2}{2\sigma^2}}$

then $\exists\ \alpha_1, \mu_1, \sigma_1$ s.t. $P(X_1 = x_1) = \alpha_1 e^{-\frac{(x_1-\mu_1)^2}{2\sigma_1^2}}$

Solution:

$$P(X_1 = x_1) = \int P(X_1 = x_1 \mid X_0 = x_0) \cdot P(X_0 = x_0)\, dx_0$$

$$= \alpha_0 \cdot \alpha \cdot \int \exp\left(-\frac{\sigma^2(x_0-\mu_0)^2 + \sigma_0^2(x_1-x_0)^2}{2\sigma_0^2\sigma^2}\right) dx_0$$

$$= \alpha_0 \cdot \alpha \cdot \int \exp\left(-\frac{x_0^2(\sigma^2+\sigma_0^2) - 2(\sigma^2\mu_0 + \sigma_0^2 x_1)x_0 + \sigma^2\mu_0^2 + \sigma_0^2 x_1^2}{2\sigma_0^2\sigma^2}\right) dx_0$$

$$= \alpha_0 \cdot \alpha \cdot \int \exp\left(-\frac{x_0^2 - 2x_0\left(\frac{\sigma_0^2 x_1 + \sigma^2 x_0}{\sigma_0^2+\sigma^2}\right) + \left(\frac{\sigma_0^2 x_1 + \sigma^2\mu_0}{\sigma_0^2+\sigma^2}\right)^2}{2\sigma_0^2\sigma^2/(\sigma_0^2+\sigma^2)}\right)$$

$$\cdot \exp\left(-\frac{\sigma_0^2 x_1^2 + \sigma^2\mu_0^2 - \frac{(\sigma_0^2 x_1 + \sigma^2\mu_0)^2}{\sigma_0^2+\sigma^2}}{2\sigma_0^2\sigma^2}\right) dx_0.$$

Now that we have completed the square & put it in the Gaussian form, suppose $\exists$ a normalization constant α_2

Then $P(X_1 = x_1) = \frac{\alpha_0 \cdot \alpha}{\alpha_2} \cdot \exp\left(-\frac{(\sigma_0^2 x_1^2 + \sigma^2\mu_0^2)(\sigma_0^2+\sigma^2) - (\sigma_0^2 x_1 + \sigma^2 x_0)^2}{2\sigma_0^2\sigma^2(\sigma_0^2+\sigma^2)}\right)$

$$= \frac{\alpha_0\alpha}{\alpha_2} \cdot \exp\left(-\frac{[\sigma_0^2(\sigma_0^2+\sigma^2) - \sigma_0^4]x_1^2 + \sigma^2\mu_0^2(\sigma_0^2+\sigma^2) - 2\sigma_0^2\sigma^2\mu_0 x_1 - \sigma^4\mu_0^2}{2\sigma_0^2\sigma^2(\sigma_0^2+\sigma^2)}\right)$$

$$= \frac{\alpha_0\cdot\alpha}{\alpha_2} \cdot \exp\left(-\frac{x_1^2 - 2\mu_0 x_1 + \mu_0^2}{2(\sigma_0^2+\sigma^2)}\right) = \frac{\alpha_0\cdot\alpha}{\alpha_2} \cdot \exp\left(\frac{(x_1-\mu_0)^2}{2(\sigma_0^2+\sigma^2)}\right)$$

where $\mu_1 = \mu_0$, $\sigma_1 = \sqrt{\sigma_0^2+\sigma^2}$, $\alpha_1 = \frac{1}{\sqrt{2\pi(\sigma_0^2+\sigma^2)}}$

3) Find the eigenvalues & eigenvectors of $A = \begin{pmatrix} 13 & 5 \\ 2 & 4 \end{pmatrix}$

Solution: For eigenvalues.

$$\begin{pmatrix} 13-\lambda & 5 \\ 2 & 4-\lambda \end{pmatrix} = \lambda^2 - 17\lambda + 42 = (\lambda-3)(\lambda-14).$$

$\Rightarrow \lambda_1 = 3, \lambda_2 = 14.$

$\Rightarrow \lambda_1 = 3$

$$\text{eigenvector} = \begin{bmatrix} -\sqrt{\frac{1}{5}} \\ \sqrt{\frac{4}{5}} \end{bmatrix}$$

$\Rightarrow \lambda_2 = 14$

$$\text{eigenvector} = \begin{bmatrix} -\sqrt{\frac{25}{26}} \\ \sqrt{\frac{1}{26}} \end{bmatrix}$$

4) Provide one example each for

① $(A+B)^2 \neq A^2+2AB+B^2$

② $AB=0$, $A\neq 0$, $B\neq 0$ where A, B are 2×2.

Solution: For ①, Let $A=\begin{pmatrix}1&0\\0&0\end{pmatrix}$, $B=\begin{pmatrix}1&1\\0&0\end{pmatrix}$

$(A+B)^2=\begin{pmatrix}2&1\\0&0\end{pmatrix}^2=\begin{pmatrix}4&2\\0&0\end{pmatrix}$

$A^2+2AB+B^2=\begin{pmatrix}1&0\\0&0\end{pmatrix}+2\begin{pmatrix}1&1\\0&0\end{pmatrix}+\begin{pmatrix}1&1\\0&0\end{pmatrix}=\begin{pmatrix}4&3\\0&0\end{pmatrix}$

$\Rightarrow$ Not equal.

For ②, Let $A=\begin{pmatrix}1&1\\0&0\end{pmatrix}$, $B=\begin{pmatrix}1&0\\-1&0\end{pmatrix}$

Then $AB=\begin{pmatrix}0&0\\0&0\end{pmatrix}$ while $A\neq 0$, $B\neq 0$.

5) WTS $A=\underline{I}-2uu^T$ is orthogonal i.e. $A^TA=\underline{I}$

Solution: $A^TA=(\underline{I}-2uu^T)^T(\underline{I}-2uu^T)$

$=(\underline{I}-2uu^T)(\underline{I}-2uu^T)$

$=\underline{I}-4uu^T+4(uu^T)(uu^T)$

$=\underline{I}-4uu^T+4u(u^Tu)u^T$

$=\underline{I}$.

6) ① WTS $f(x)=x^3$ is convex for $x\geq 0$.

Solution: $f''(x)=(3x^2)'=6x$ where $x\geq 0$. $\Rightarrow$ It is convex.

② WTS $f(x_1,x_2)=\max(x_1,x_2)$ is convex on $\mathbb{R}^2$.

Solution: $\lambda\in[0,1]$, let $\hat{\lambda}=1-\lambda$

$\Rightarrow$ We have $f(\lambda x_1+\hat{\lambda}x_2)=\max(f(\lambda x_1+\hat{\lambda}x_2)$

$=\max(f(\lambda x_1))+\max(f(\hat{\lambda}x_2))$ $\Rightarrow \geq 0$.

$\Rightarrow$ Convex.

③ WTS if univariate functions f & g are convex on S, then $f+g$ is convex on S.

Solution: Let $\lambda \in [0,1]$, $\hat{\lambda} = 1-\lambda$

$$\Rightarrow (f+g)(\lambda x + \hat{\lambda} y) = f(\lambda x + \hat{\lambda} y) + g(\lambda x + \hat{\lambda} y)$$
$$\leq \lambda f(x) + \hat{\lambda} f(y) + \lambda g(x) + \hat{\lambda} g(y)$$
$$= \lambda (f+g)(x) + (1-\lambda)(f+g)(y)$$

$\Rightarrow (f+g)$ is convex.

④ WTS if univariate functions f & g are convex & non-negative on S, and have their minimum within S at the same point, then fg is convex on S.

Solution: WTS fg is convex on S.

$\Rightarrow$ WTS $(fg)''(x) \geq 0$.

$(fg)''(x) = (f'g + fg')(x) = (f''g + 2f'g' + fg'')(x)$ (*)

f'' & f, g'' & g are convex & non-neg.

$\Rightarrow f''g$ & fg'' are non-neg. on the set.

Suppose $\exists$ a minimum X_1 for f & g,

when $x \leq X_1$, then f & g are non-neg. $\Rightarrow f'g' \geq 0$.

when $x > X_1$, then f' & g' are non-neg. $\Rightarrow f'g' \geq 0$.

$\Rightarrow$ We have (*) is always non-neg. $\Rightarrow$ Convex.

7) Find the categorical distribution that has the highest entropy

Solution: $\sum_{i=1}^{k} p_i - 1 = 0$ which is the normalized constraint

Using Lagrange multipliers,

$$\alpha = -\sum_{i=1}^{k} p_i \log(p_i) + \lambda\left(\sum_{i=1}^{k} p_i - 1\right)$$

$$\Rightarrow \frac{\partial \alpha}{\partial p_i} = -\log(p_i) - 1 + \lambda = 0$$

$$\rightarrow \log(p_i) = \lambda - 1$$

$\Rightarrow$ All p_i's have to be $\frac{1}{k}$ in order to satisfy normalized constraint.

Since the stationary point p^* is unique

& $H(p^*) = \frac{1}{k} \geq 0$. w/ $H([1, 0, 0, \cdots, 0]) = 0$

$\Rightarrow p^*$ has to be a maximum.

Problem 1):

a) Show $J(\theta) = (X\theta - y)^T W (X\theta - y)$ & state W.

Solution: let $\frac{1}{2} w^{(i)} = w^{(ii)}$

We know that $x^T A x = \sum_{ij}^{m} A_{ij} x_i x_j$.

By working it one backwards:

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

$$= \sum_{ij}^{m} w^{(ii)} (X\theta - y)_i (X\theta - y)_j$$

$$= \sum_{ij}^{m} (\theta^T x^{(i)} - y^{(i)}) \cdot w^{(ii)} \cdot (\theta^T x^{(j)} - y^{(j)})$$

By setting $i = j$, then $J(\theta) = \sum_{i=1}^{m} w^{(ii)} (\theta^T x^{(i)} - y^{(i)})^2$.

Substituting $\frac{1}{2} w^{(i)} = w^{(ii)}$ back,

we get $J(\theta) = \frac{1}{2} \sum_{i=1}^{m} w^{(i)} (\theta^T x^{(i)} - y^{(i)})^2$

where W is a $m \times m$ diagonal matrix. $\Rightarrow$ (Q.E.D)

b) Solution:

We know that $\nabla_\theta J(\theta) = 2(W(X\theta - \vec{y}))^T X$ (*)

$\Rightarrow$ By setting (*) $= 0$

$$\Rightarrow (*) = X^T (W(X\theta - \vec{y}))$$

$$= X^T W X \theta - X^T W \vec{y} = 0.$$

$$\Rightarrow X^T W X \theta = X^T W \vec{y}$$

$$\Rightarrow \theta = \frac{X^T W \vec{y}}{X^T W X} = (X^T W X)^{-1} \cdot (X^T W \vec{y})$$

c) From b), we know that

$$\theta = (X^T W X)^{-1} \cdot (X^T W \vec{y}) \quad (*)$$

=> ① Define a function using x_0, X, Y, tao as parameters.

② Adding the bias terms to X & x_0.

③ Fitting the normal equations with kernel of parameters x_0, X, tao.

④ Setting $\theta = (*)$.

⑤ Return x_0 at $(*)$ as

⑥ define another function using x_0, X, and tao as parameters.

⑦ ~~Ha~~ Return value using the x_0 from above & the formula $w^{(i)} = \exp\left(-\frac{(x - x^{(i)})^T (x - x^{(i)})}{2\tau^2}\right)$

And it is non-parametric.

Problem 2):

a) Show $E[\theta] = \theta^*$ for the least squares estimator.

Solution:

Taking $\theta^T = \frac{\sum (x^{(i)} - \bar{x}) y^{(i)}}{\sum (x^{(i)})^2 - n\bar{x}^2}$

$$E[\theta^T | x^{(i)}] = \frac{\sum (x^{(i)} - \bar{x}) E[y^{(i)} | x^{(i)}]}{\sum (x^{(i)})^2 - n\bar{x}^2} \quad (*)$$

Also, we know that $E[y^{(i)} | x^{(i)}] = \theta^T x^{(i)}$ since $E[\epsilon^{(i)}] = 0$ by assumption. $\Rightarrow E[\epsilon^{(i)} | x^{(i)}] = 0$.

$$\Rightarrow (*) = \frac{\sum (x^{(i)} - \bar{x}) \cdot \theta^T x^{(i)}}{\sum (x^{(i)})^2 - n\bar{x}^2}$$

$$= \frac{\theta^T \sum (x^{(i)})^2 - \theta^T \sum \bar{x} x^{(i)}}{\sum (x^{(i)})^2 - n\bar{x}^2} = \frac{\theta^T \sum (x^{(i)})^2 - \theta^T n\bar{x}^2}{\sum (x^{(i)})^2 - n\bar{x}^2}$$

$$= \frac{\theta^T \left(\sum (x^{(i)})^2 - n\bar{x}^2 \right)}{\sum (x^{(i)})^2 - n\bar{x}^2} = \theta^T$$

$\Rightarrow E[\theta^T | x^{(i)}] = \theta^T$

$\Rightarrow E[\theta] = \theta^*$ which is the true value of the parameter which is unbiased.

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

Solution: let $i = 1, 2, \ldots$

$$\Rightarrow Var(\theta^{(i)}) = Var\left(\frac{\Sigma (y^{(i)} - \theta^{(0)}) x^{(i)}}{\Sigma (x^{(i)})^2}\right)$$

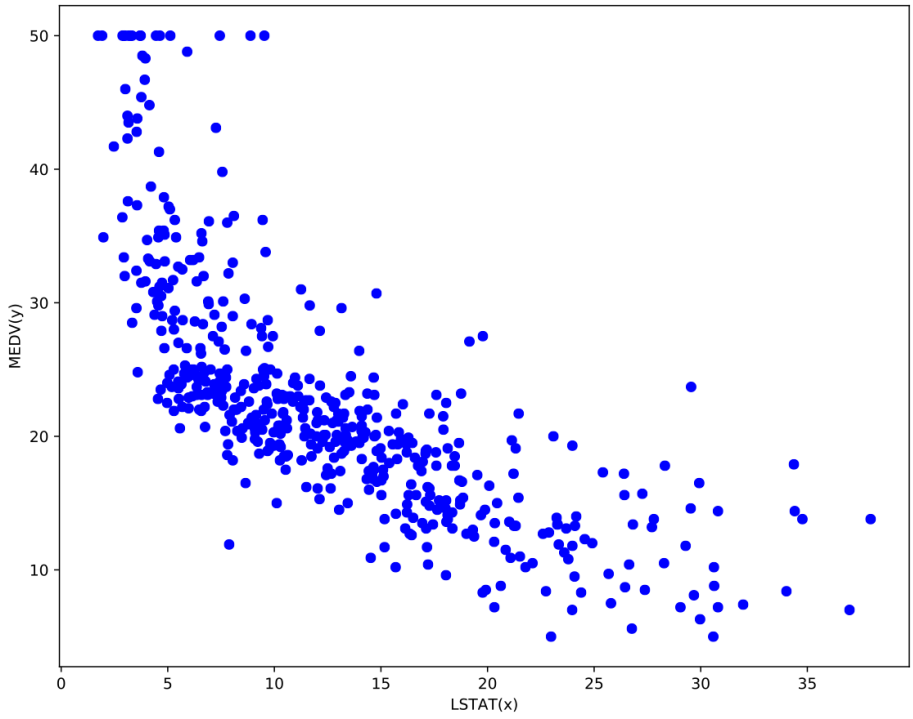
$$= \frac{1}{\left(\Sigma (x^{(i)})^2\right)^2} \cdot Var\left(\Sigma y^{(i)} x^{(i)}\right)$$

$$= \frac{1}{\left(\Sigma (x^{(i)})^2\right)^2} \cdot \Sigma (x^{(i)})^2$$

$$= \frac{\sigma^2}{\Sigma (x^{(i)})^2} = \frac{\sigma^2}{X^T X} = (X^T X)^{-1} \sigma^2$$

Problem 3:
3.1A

Plotting the data

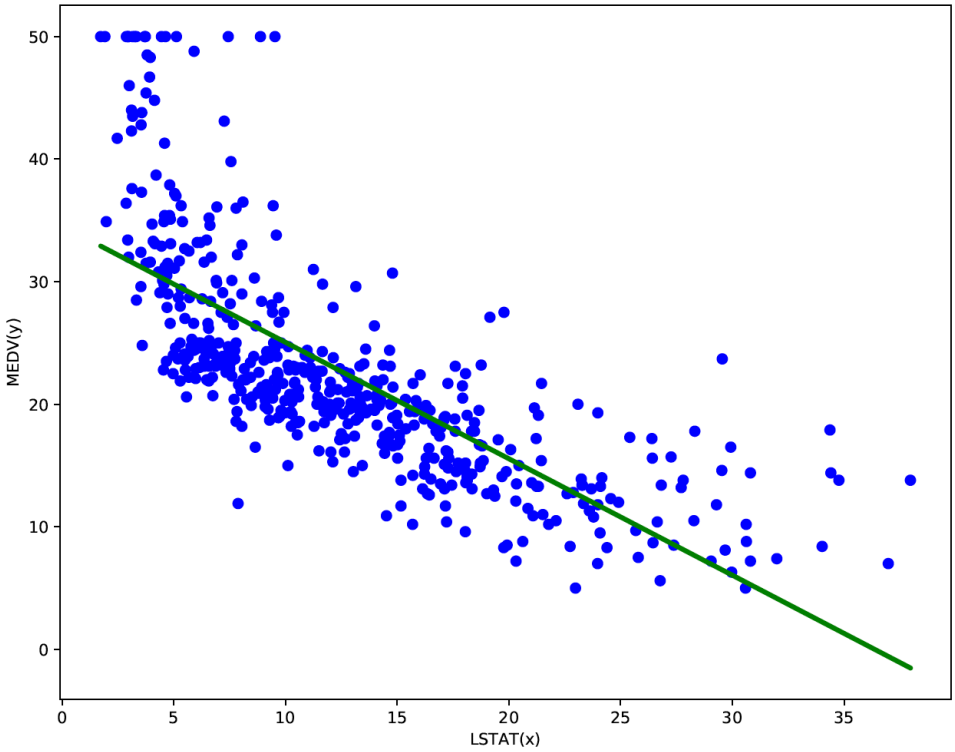


A1: Computing the cost function J(θ)
A2: Implementing gradient descent

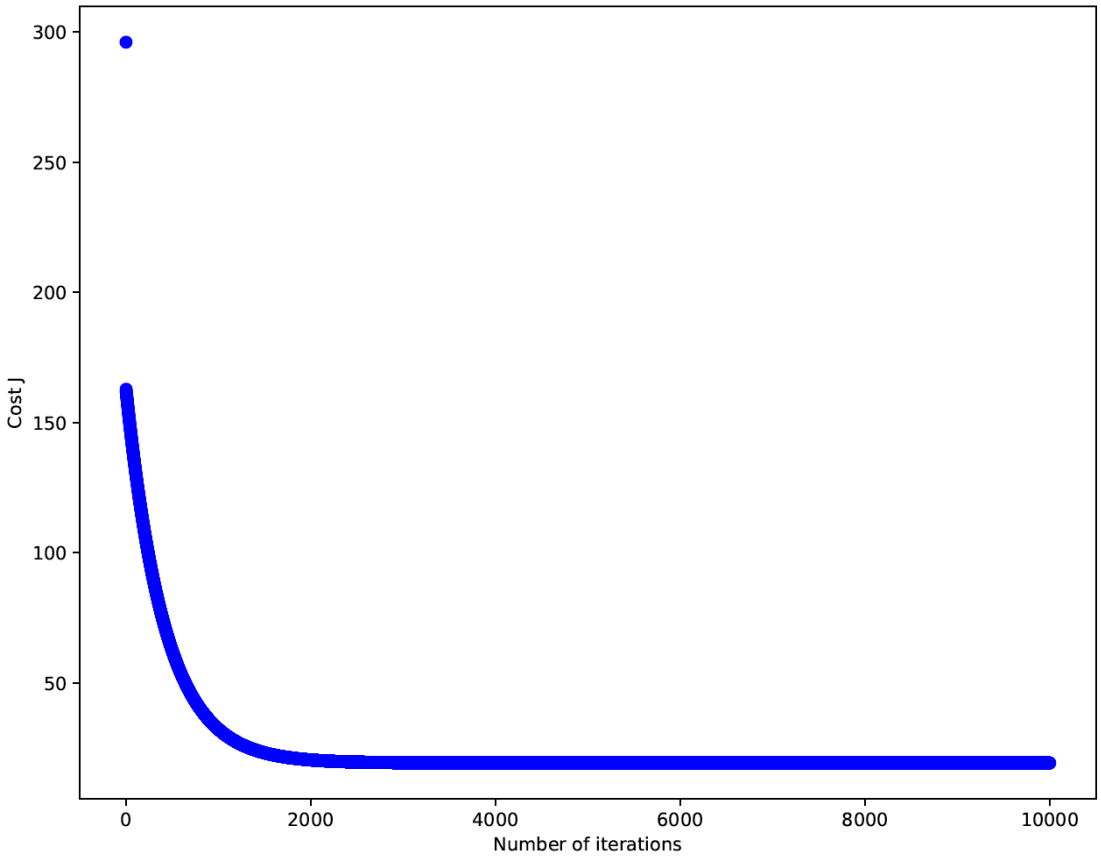
iteration 0 / 10000: loss 296.073458
iteration 1000 / 10000: loss 32.190429
iteration 2000 / 10000: loss 20.410446
iteration 3000 / 10000: loss 19.347011
iteration 4000 / 10000: loss 19.251010
iteration 5000 / 10000: loss 19.242344
iteration 6000 / 10000: loss 19.241561
iteration 7000 / 10000: loss 19.241491
iteration 8000 / 10000: loss 19.241484
iteration 9000 / 10000: loss 19.241484

Theta found by gradient_descent: [34.55363411 -0.95003694]

Using final parameters to plot the linear fit



Plotting the J (θ) values during gradient descent



- Qualitative analysis of the linear fit

➢ What can you say about the quality of the linear fit for this data?

There are some outliers especially at the lower and higher ends as we can see that are not close to the fitted line. Thus, we say that it is not a good fit.

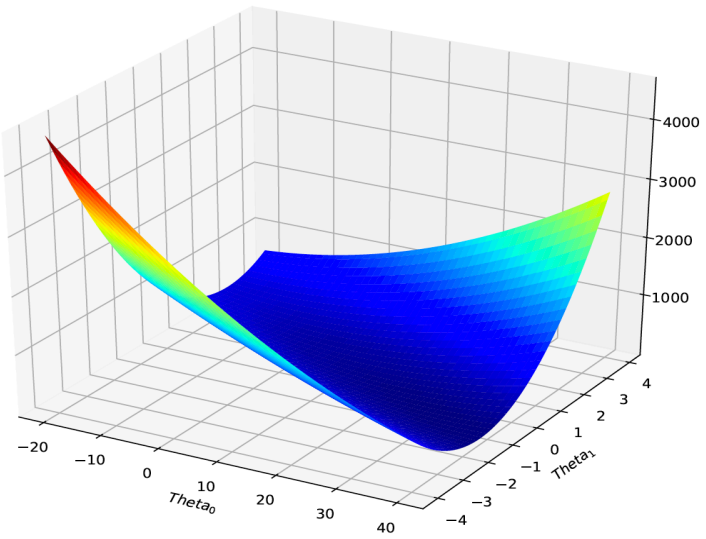
➢ In your assignment writeup.pdf, explain how you expect the model to perform at the low and high ends of values for LSTAT?

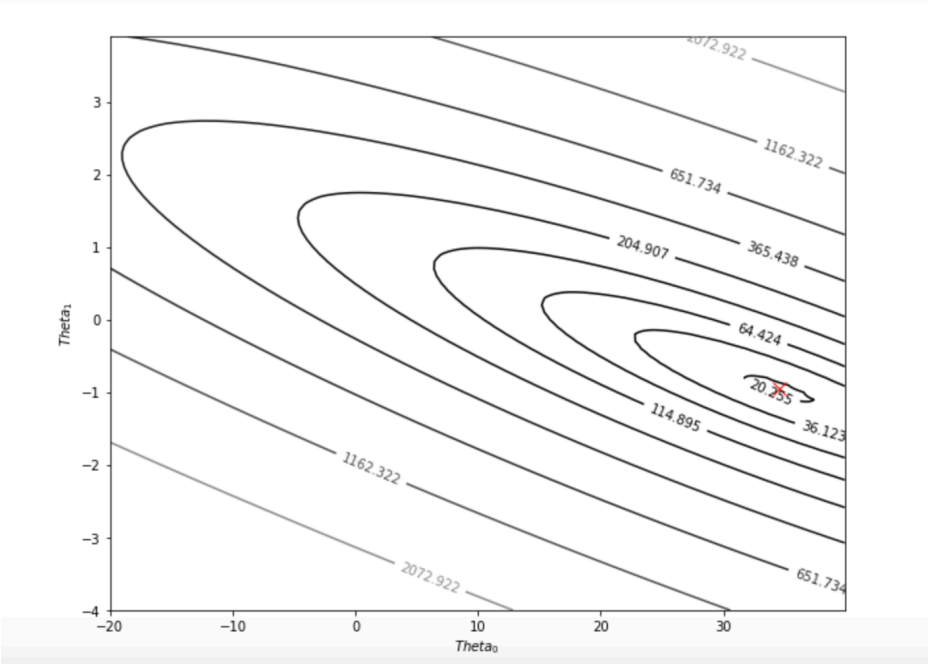
I would expect that as percentage of population with lower economic status decreases, the meadian home value in $10000s would be higher.

➢ How could we improve the quality of the fit?

We could add more regressors (Increasing R^2 value) rather than doing a simple linear regression.

Visualizing J (θ)





A3: Predicting on unseen data

For lower status percentage = 5, we predict a median home value of 298034.494122

For lower status percentage = 50, we predict a median home value of -129482.128898

The coefficients computed by sklearn:

34.5538408794 and -0.950049353758

```
------------- Estimating model on training data -------------
iteration 0 / 10000: loss 296.073458
iteration 1000 / 10000: loss 32.190429
iteration 2000 / 10000: loss 20.410446
iteration 3000 / 10000: loss 19.347011
iteration 4000 / 10000: loss 19.251010
iteration 5000 / 10000: loss 19.242344
iteration 6000 / 10000: loss 19.241561
iteration 7000 / 10000: loss 19.241491
iteration 8000 / 10000: loss 19.241484
iteration 9000 / 10000: loss 19.241484
Theta found by gradient_descent on training data:  [ 34.55363411  -0.9
5003694]
------------- Evaluating model on test data -------------------
Residual mean squared error:  43.0269872405
Variance explained by model:  0.568986637831
--------- Estimating/Evaluating model by crossvalidation ------
iteration 0 / 10000: loss 303.118502
iteration 1000 / 10000: loss 35.238515
iteration 2000 / 10000: loss 22.628279
iteration 3000 / 10000: loss 21.479423
iteration 4000 / 10000: loss 21.374756
iteration 5000 / 10000: loss 21.365220
iteration 6000 / 10000: loss 21.364352
iteration 7000 / 10000: loss 21.364272
iteration 8000 / 10000: loss 21.364265
iteration 9000 / 10000: loss 21.364265
```

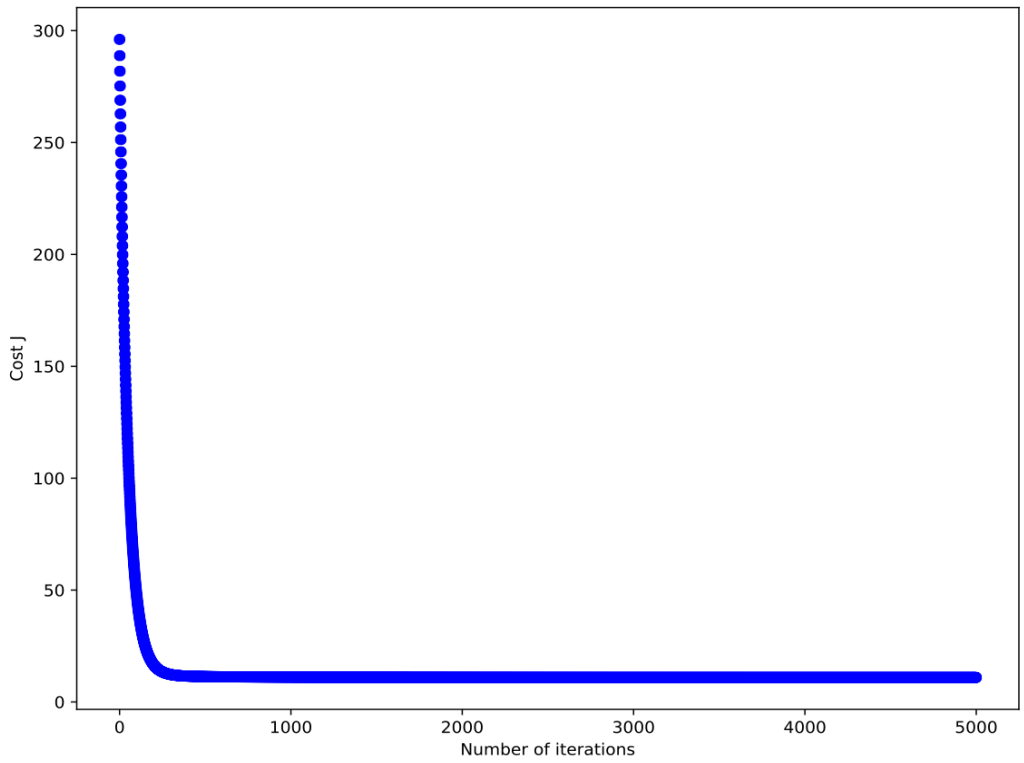
```
Residual mean squared error:  23.557634795
Variance explained by model:  0.31786588392
iteration 0 / 10000: loss 284.188716
iteration 1000 / 10000: loss 31.234345
iteration 2000 / 10000: loss 20.076450
iteration 3000 / 10000: loss 19.082816
iteration 4000 / 10000: loss 18.994331
iteration 5000 / 10000: loss 18.986451
iteration 6000 / 10000: loss 18.985750
iteration 7000 / 10000: loss 18.985687
iteration 8000 / 10000: loss 18.985682
iteration 9000 / 10000: loss 18.985681
Residual mean squared error:  41.821945299
Variance explained by model:  0.540603725401
iteration 0 / 10000: loss 249.897210
iteration 1000 / 10000: loss 28.596381
iteration 2000 / 10000: loss 17.098133
iteration 3000 / 10000: loss 15.796102
iteration 4000 / 10000: loss 15.648664
iteration 5000 / 10000: loss 15.631968
iteration 6000 / 10000: loss 15.630078
iteration 7000 / 10000: loss 15.629864
iteration 8000 / 10000: loss 15.629840
iteration 9000 / 10000: loss 15.629837
Residual mean squared error:  73.9968055907
Variance explained by model:  0.0760470759396
iteration 0 / 10000: loss 308.907778
iteration 1000 / 10000: loss 32.281782
iteration 2000 / 10000: loss 19.316662
iteration 3000 / 10000: loss 18.033820
iteration 4000 / 10000: loss 17.906888
iteration 5000 / 10000: loss 17.894329
iteration 6000 / 10000: loss 17.893086
iteration 7000 / 10000: loss 17.892964
iteration 8000 / 10000: loss 17.892951
iteration 9000 / 10000: loss 17.892950
Residual mean squared error:  50.5004502309
Variance explained by model:  0.424245991773
iteration 0 / 10000: loss 334.272481
iteration 1000 / 10000: loss 32.784978
iteration 2000 / 10000: loss 22.063887
```

```
iteration 3000 / 10000: loss 21.304138
iteration 4000 / 10000: loss 21.250299
iteration 5000 / 10000: loss 21.246484
iteration 6000 / 10000: loss 21.246213
iteration 7000 / 10000: loss 21.246194
iteration 8000 / 10000: loss 21.246193
iteration 9000 / 10000: loss 21.246193
Residual mean squared error:  23.2176807345
Variance explained by model:  0.126771322851
5  fold cross_validation MSE =  42.61890333
5  fold cross_validation r_squared =  0.297106799977
```

3.1B

B1: Feature normalization

B2: Loss function and gradient descent



Theta computed by gradient descent:

[2.25328063e+01 -9.13925619e-01 1.06949712e+00

1.07531669e-01 6.87258582e-01 -2.05340341e+00

2.67719690e+00 1.55788957e-02 -3.10668099e+00

2.56946272e+00 -1.97453430e+00 -2.05873147e+00

8.55982884e-01 -3.74517559e+00]

B3: Making predictions on unseen data

for average home in Boston suburbs, we predict a median home value of 225328.063241

B4: Normal equations

Theta computed by direct solution is:

[3.64911033e+01 -1.07170557e-01 4.63952195e-02

2.08602395e-02 2.68856140e+00 -1.77957587e+01

3.80475246e+00 7.51061703e-04 -1.47575880e+00

3.05655038e-01 -1.23293463e-02 -9.53463555e-01

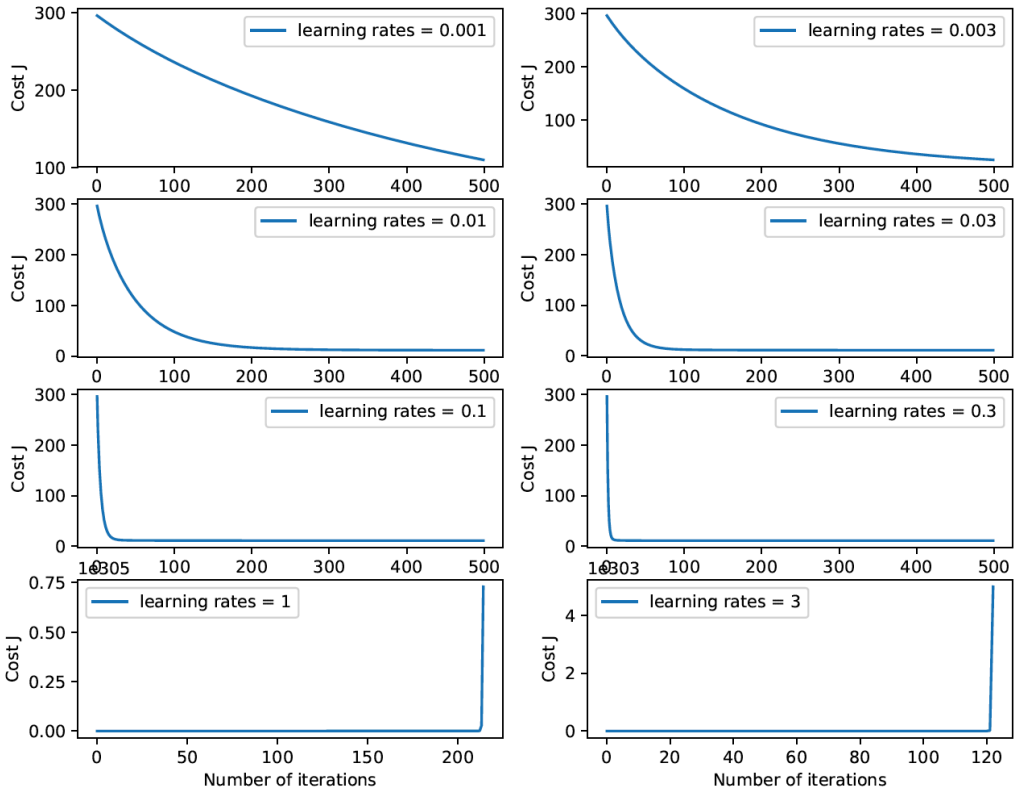
9.39251272e-03 -5.25466633e-01]

for average home in Boston suburbs, we predict a median home value of 225328.063241

- Do the predictions match up?

Yes.

B5: Exploring convergence of gradient descent



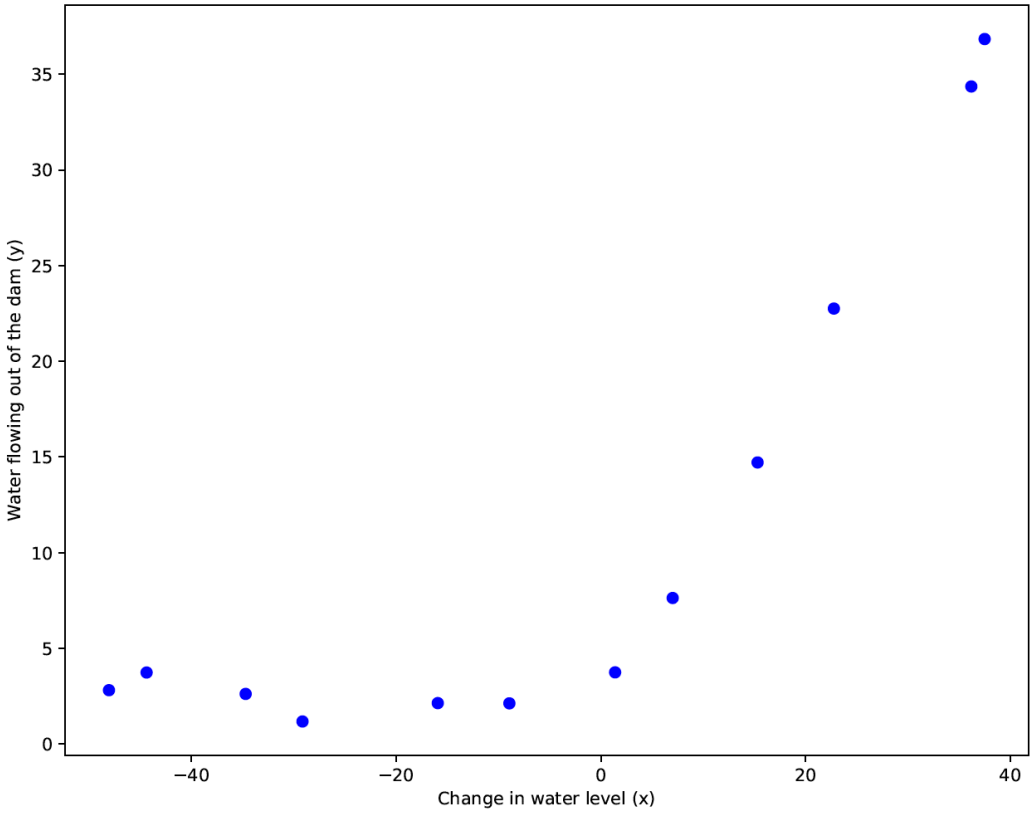
- What are good learning rates and number of iterations for this problem?

The learning rate of 0.03 and 0.01 is good. When learning rates = 0.001 or 0.003, the curves converge too slowly. As learning rates = 0.1 or 0.3, they

converge too quickly. And as for rates = 1 or 3, they are not descending. Thus, the ideal iterations for rate = 0.01 would be around 300 and for rate = 0.03 is around 100.

3.2A

Plotting the training data

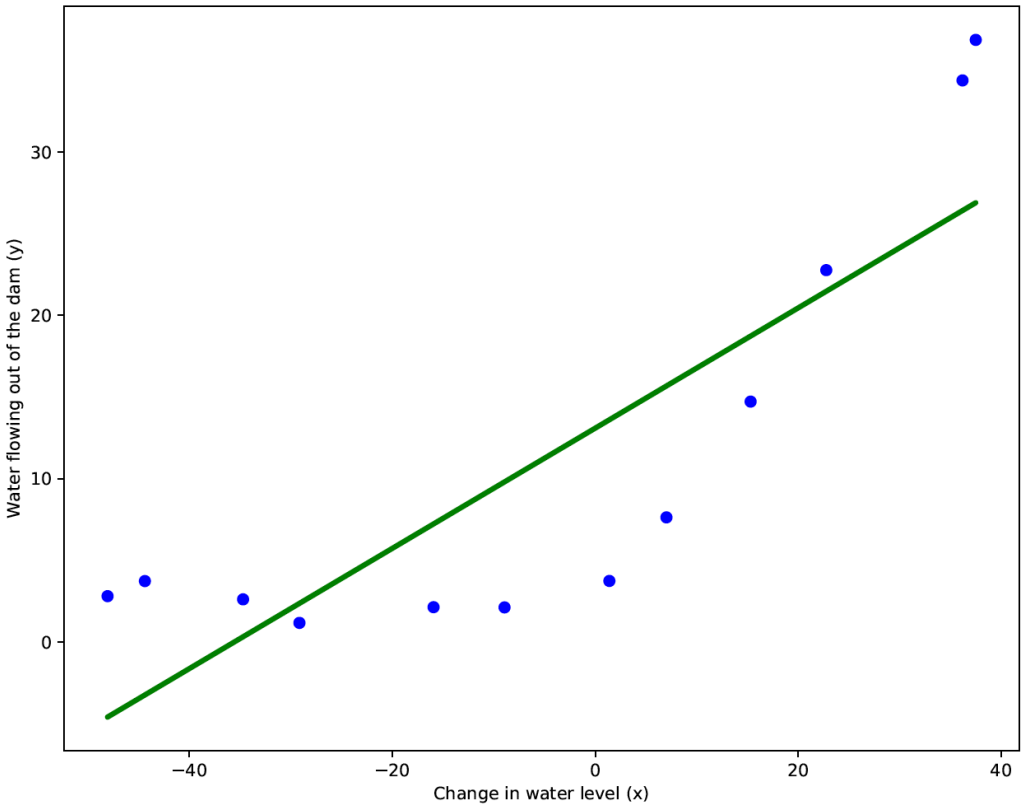


A1: Regularized linear regression cost function

A2: Gradient of the Regularized linear

regression cost function

Optimization terminated successfully.
Current function value: 22.373906
Iterations: 5
Function evaluations: 6
Gradient evaluations: 6
Theta at lambda = 0 is [13.08790353 0.36777923]

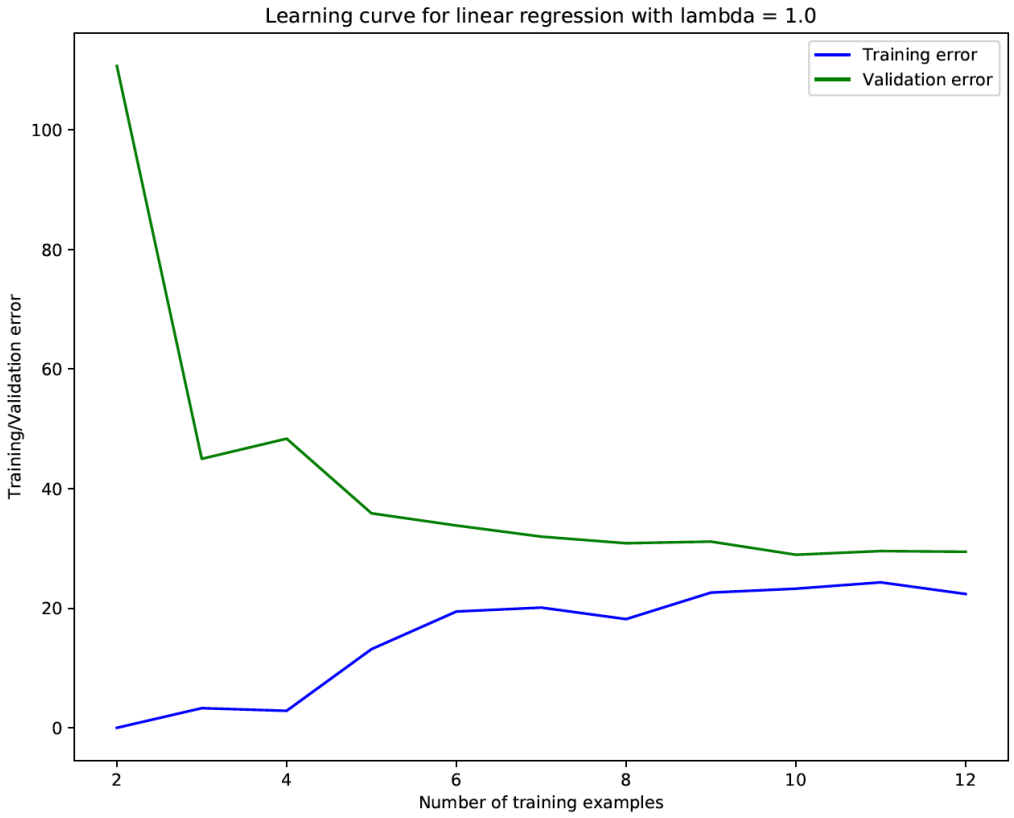


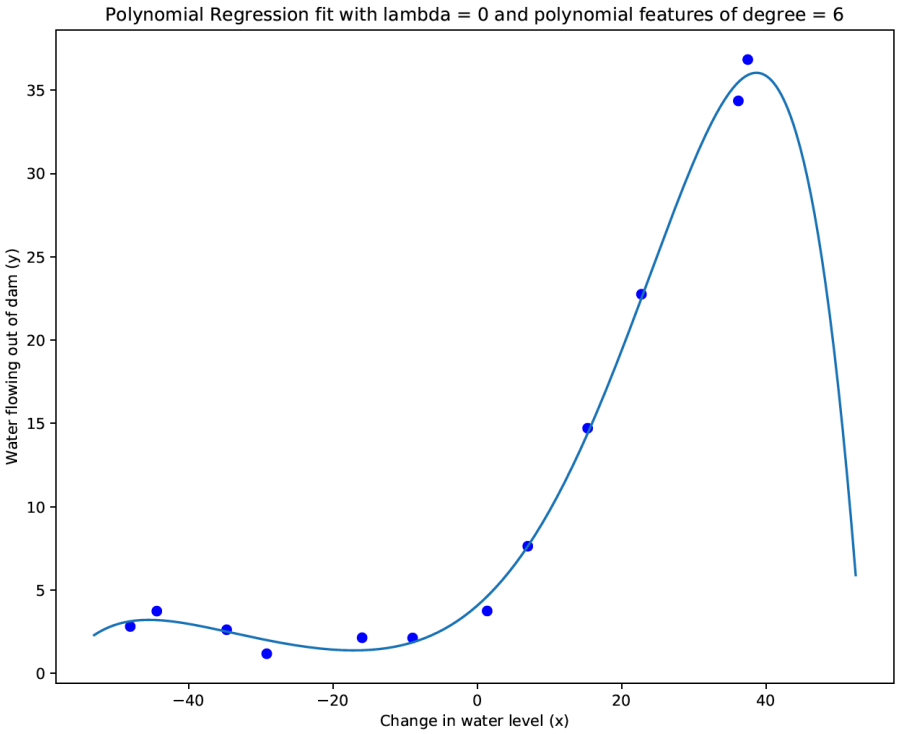
A3: Learning curves

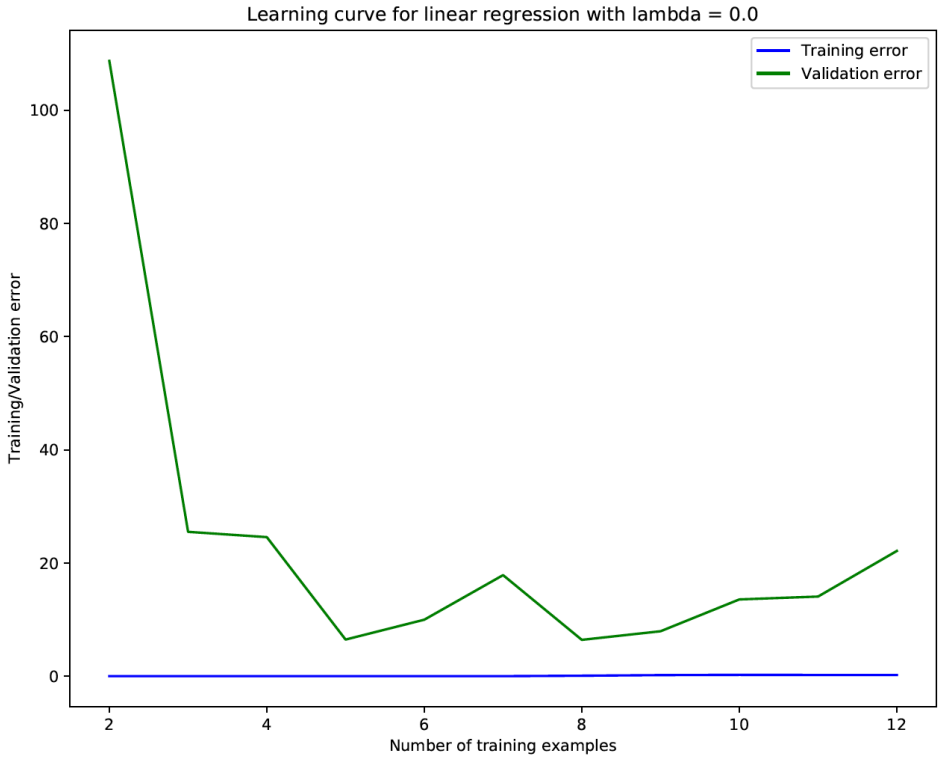
Optimization terminated successfully.
Current function value: 0.000000
Iterations: 5
Function evaluations: 9
Gradient evaluations: 9
Optimization terminated successfully.

Current function value: 0.001307
Iterations: 10
Function evaluations: 11
Gradient evaluations: 11
Optimization terminated successfully.
Current function value: 3.335017
Iterations: 3
Function evaluations: 5
Gradient evaluations: 5
Optimization terminated successfully.
Current function value: 2.881847
Iterations: 3
Function evaluations: 5
Gradient evaluations: 5
Optimization terminated successfully.
Current function value: 13.174273
Iterations: 5
Function evaluations: 6
Gradient evaluations: 6
Optimization terminated successfully.
Current function value: 19.461396
Iterations: 5
Function evaluations: 6
Gradient evaluations: 6
Optimization terminated successfully.
Current function value: 20.112149
Iterations: 4
Function evaluations: 5
Gradient evaluations: 5
Optimization terminated successfully.
Current function value: 18.184047
Iterations: 5
Function evaluations: 6
Gradient evaluations: 6
Optimization terminated successfully.
Current function value: 22.618880
Iterations: 5
Function evaluations: 6
Gradient evaluations: 6
Optimization terminated successfully.
Current function value: 23.268598

Iterations: 6
Function evaluations: 7
Gradient evaluations: 7
Optimization terminated successfully.
Current function value: 24.323253
Iterations: 5
Function evaluations: 6
Gradient evaluations: 6
Optimization terminated successfully.
Current function value: 22.379542
Iterations: 5
Function evaluations: 6
Gradient evaluations: 6

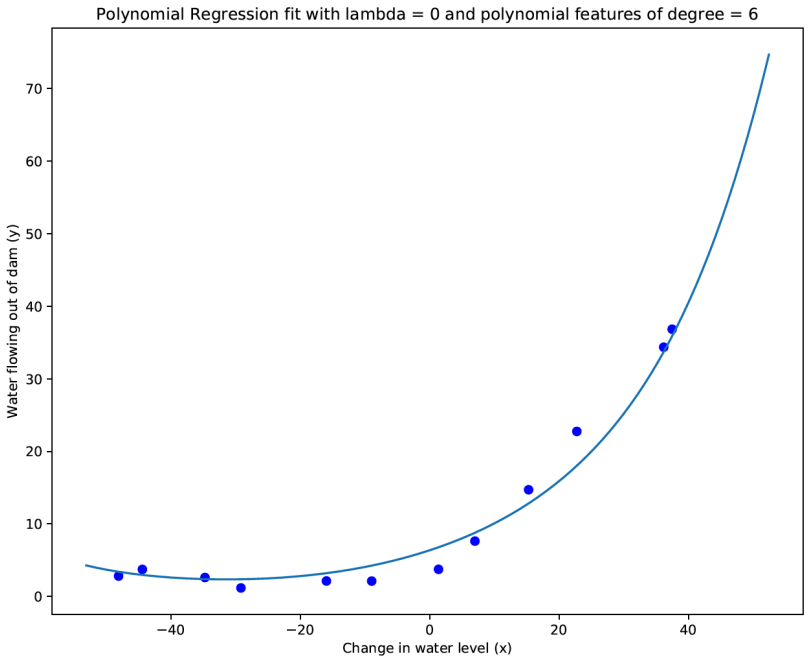


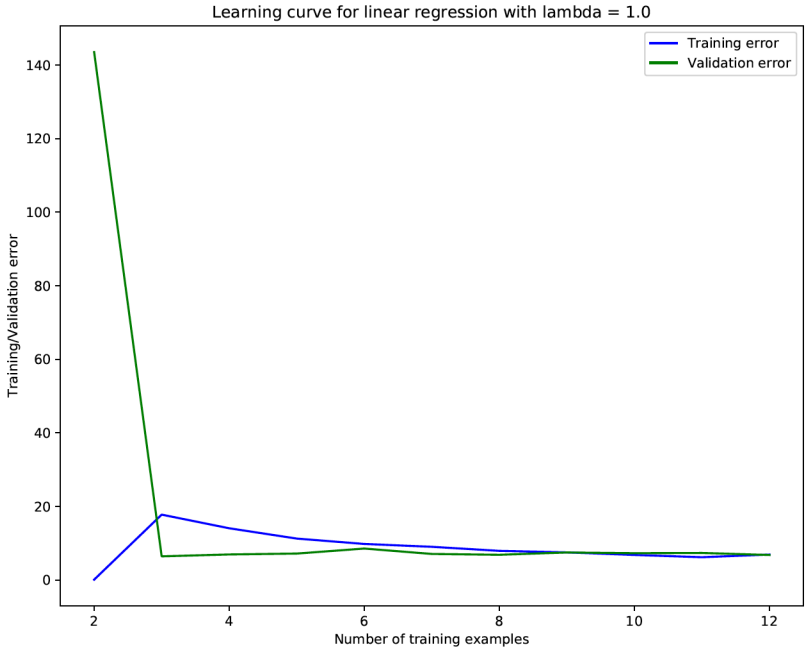

Polynomial Regression fit with lambda = 0 and polynomial features of degree = 6
Water flowing out of dam (y)
Change in water level (x)
35
30
25
20
15
10
5
0
−40
−20
0
20
40


Learning curve for linear regression with lambda = 0.0
Training error
Validation error
Training/Validation error
Number of training examples
100
80
60
40
20
0
2
4
6
8
10
12

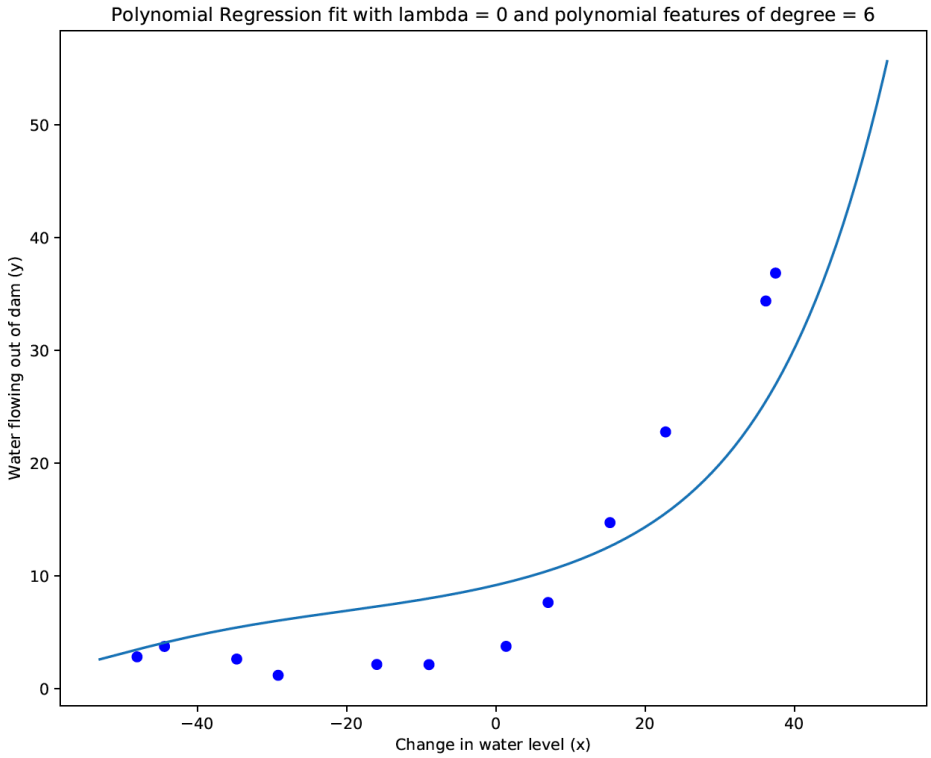
A4: Adjusting the regularization parameter

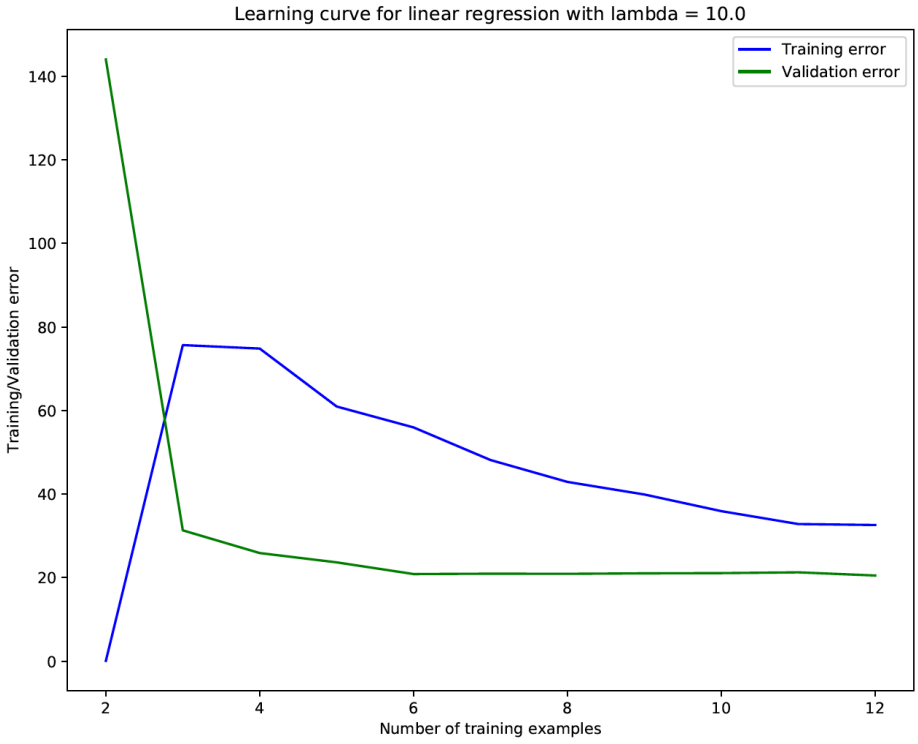
When $\lambda = 1$,



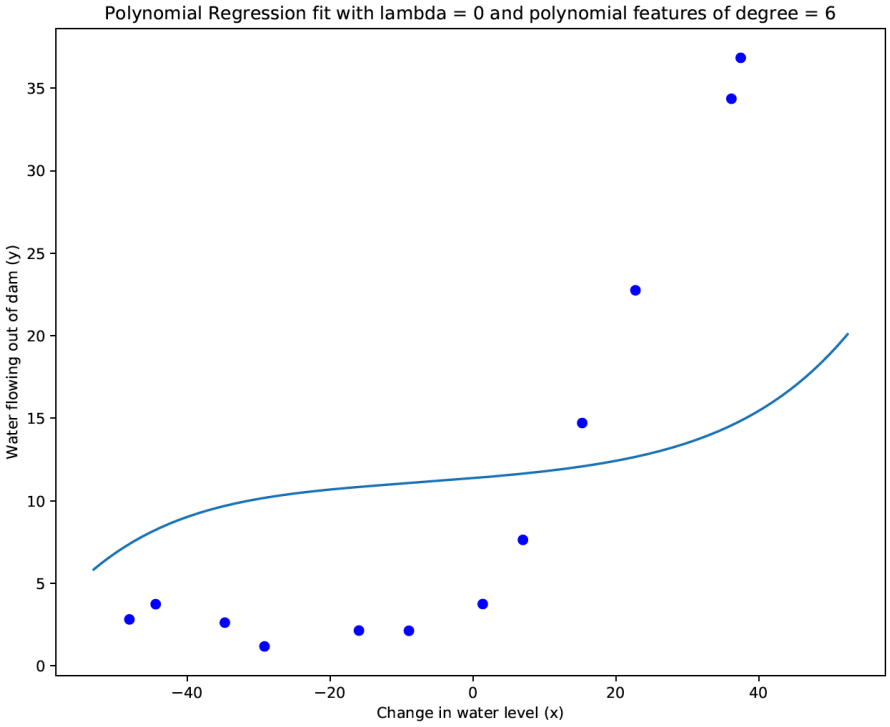


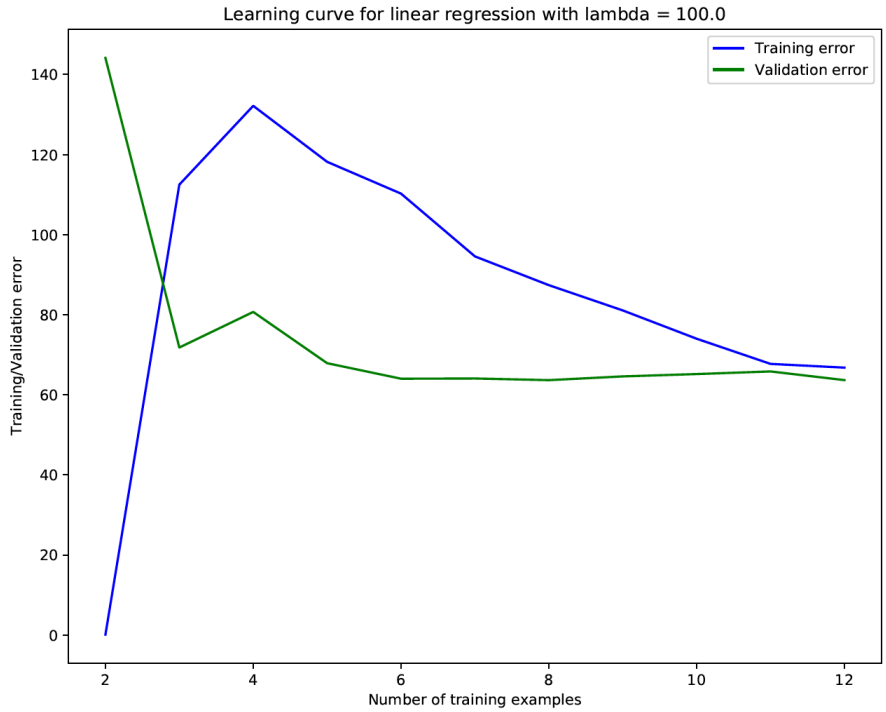
When λ =10,





When λ =100,

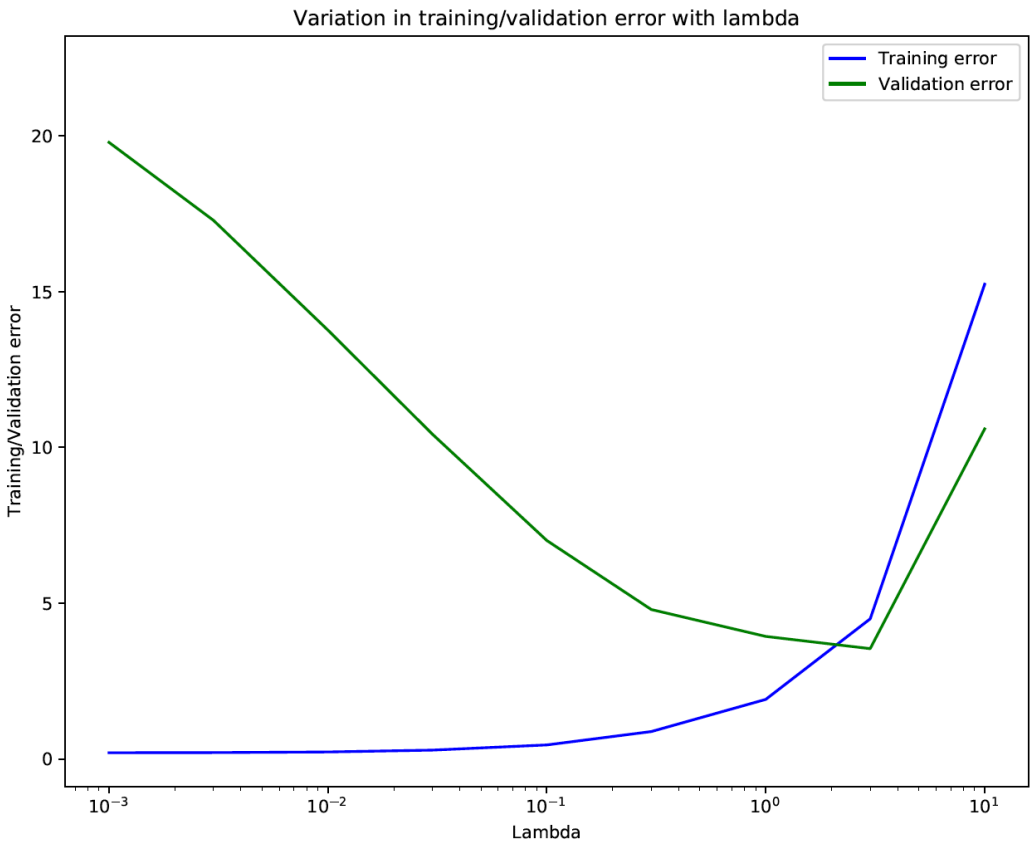




- Comment on the impact of the choice of lambda on the quality of the learned model.

As λ increases, the model becomes under-fit. Thus, the λ here has inc reased too far that it passed the best - fitted part.

A5: Selecting λ using a validation set



We chose λ= 3 since the validation error is the lowest amongst all others.

A6: Computing test set error

We choose λ= 3 and 4.39762337668 is our computed test set error.

A7: Plotting learning curves with randomly selected examples