CSCI567 Fall 2015

Homework #3

Vaibhav Behl

3123054410

vbehl@usc.edu

1. Logistic Regression and Regularization

a) We are given n training examples as- $(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$.

Now, for logistic regression, we can write the conditional probability expression as:

$$P(y_n|x_n; b; w) = \sigma(b + w^T x_n)^{y_n}[1 - \sigma(b + w^T x_n)]^{1-y_n}$$

Taking the negative log-likelihood of the above expression we get:

$$L(w) = -\log(\prod_{i=1}^{n} P(y_i|x_i))$$

$$L(w) = -\sum_{i=1}^{n}\{y_n \log \sigma(b + w^T x_n) + (1 - y_n)\log[1 - \sigma(b + w^T x_n)]\}$$

b) To prove whether this classifier is convex or not, we will use the property that The Hessian matrix of a convex function is positive semi-definite.

$$H = \frac{\delta^2 L(w)}{\delta w^T w} = \frac{\delta}{\delta w^T}(\frac{\delta L(w)}{\delta w})$$

Now using the value of derivative of log terms from below:

$$d\frac{\log \sigma(a)}{da} = 1 - \sigma(a)$$

Substituting this value back into derivative equation we get:

$$\frac{\delta L(w)}{\delta w} = -\sum\{y_n[1 - \sigma(w^T x_n)]x_n - (1 - y_n)\sigma(w^T x_n)x_n\}$$
$$= \sum\{\sigma(w^T x_n) - y_n\}x_n$$

Now taking the second derivative of this result and removing the terms not dependent on w.

$$\frac{\delta}{\delta w^T}\left(\frac{\delta L(w)}{\delta w}\right) = \sum \frac{\delta}{\delta w^T}(\sigma(w^T x_n)x_n)$$

Using the below formula:

$$\frac{\delta \sigma(a)}{\delta a} = \sigma(a)(1 - \sigma(a))$$

And substituting its value back into the equation:

$$\frac{\delta}{\delta w^T}\left(\frac{\delta L(w)}{\delta w}\right) = \sum x_n \sigma(w^T x_n)\big(1-\sigma(w^T x_n)\big).\frac{\delta w^T x_n}{\delta w^T}$$
$$= \sum x_n((\sigma(w^T x_n)\big(1-\sigma(w^T x_n)\big)x_n^T$$

We can represent this above equation in matrix form as follows:

$$= XAX^T$$

Now this A matrix is of diagonal elements consisting of all positive elements. Thus we can split this matrix into $B^T B$. Substituting this value back, we get:

$$= XB^T BX^T$$
$$= XB^T(XB^T)^T \geq 0$$

Thus we have proved that,

$$H = \frac{\delta^2 L(w)}{\delta w^T w} \geq 0$$

Hence the function is convex.

c) When datasets are linearly separable, then the conditional probability can only have either value 1 or 0. Given that our sigmoid function is as below:

$$\sigma(w^T X) = \frac{1}{1-e^{-w^T X}}$$

With this function to get a value of perfect 1 or 0, the exponent term will have to be infinite. Thus,

$$w^T X \to \infty => w \to \infty$$

Ideally this type of separation is a perfect candidate for step function.

d) We need the first gradient of the below function:

$$L(w) = -\log\left(\prod_{i=1}^{n} P(y_i|x_i)\right) + \lambda\big||w|\big|_2^2$$

Taking its first derivative, for the first term we can pick value directly from part (b) of this question.

$$\frac{\delta L(w)}{\delta w_i} = -\{\sigma(w_i^T x_i) - y_i\}x_i + 2\lambda w$$

e) To show unique solution, we can show that the Hessian matrix is Positive Definite. Taking the double derivative form from part (b) and using the derivative of regularization term from part (d).

$$H = \sum x_n((\sigma(w^T x_n)(1 - \sigma(w^T x_n))x_n^T + \frac{\delta}{\delta w}(2\lambda w)$$

$$= \sum x_n((\sigma(w^T x_n)(1 - \sigma(w^T x_n))x_n^T + 2\lambda$$

Now the first part of this equation is (≥ 0), and the last part of the equation is (> 0). Hence the Hessian is strictly greater than zero and is positive definite, so there is a unique solution.

2. Sparsity via Regularization

a) Here we need to provide an example to show that l_0 norm is non-convex. Let's consider the example of finding the norm of vector $\binom{0}{e}$. By the definition of l_0 norm we have $||w||_0 = \#\{i: w_i \neq 0\}$. So for this vector the l_0 norm will be:

$$\lim_{e \to 0} ||w||_0 = 1$$

But, at $e = 0$, we have this norm becoming zero. Thus it in not differentiable at point $\binom{0}{e}$. We can also find infinite number of other such points where it is not differentiable. Now convex functions can only be non-diffrentiable at countable number of points, but here we can find infinite such vectors where it is non-diffrentiable.

b) l_1 norm given as: $||w||_1 = \sum_i |w_i|$, and we need to prove it as convex. Using the property of convex function:

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y)$$

Using the function as $p_1$norm we simplify what we have to prove as:

$$||tX + (1-t)Y||_1 \leq t||X||_1 + (1-t)||Y||_1$$

Now looking at the Left hand side of this equation, we can see that the values in vector X, Y could be positive or negative. This implies that $||X + Y||_1 \leq ||X||_1 + ||Y||_1$, since on the right hand side we take the norm first and then add the values. Also, the positive scalar term- t will not affect this equality (we can multiply the vectors with that scalar for both left and right hand side). Thus as the given equality holds, we can prove that $l_1$norm is convex.

c) Our given objective function to minimize is -

$$J(w) = \min_w \sum_n (y_i - w^T x_i)^2 + \lambda \left| |w| \right|_1$$

Vectorizing this equation we get,

$$= Y^T Y - 2Y^T X w + \ w^T X^T X w + \lambda \left| |w| \right|_1$$

Removing the term $Y^T Y$ since it does not contain w, we get:

$$J(w) = \min_w -2Y^T X w + w^T X^T X w + \lambda \left| |w| \right|_1 \text{ ------------ } (eq.1)$$

Now, we want to transform this equation into a QP of form:

$$\min_u \frac{1}{2} u^T Q u + c^T u \ ; subject\ to\ Au \leq b$$

To start, we introduce D additional variables $t_1, \ldots, t_D$, where $|w_d| \leq t_d$, that is our w is bound by the vector t. Now the norm of w is: $\left| |w| \right|_1 = \sum |w_d|$, as each w_d is bound by a t_d, so $\min_w \left| |w| \right|_1 = \min_w \sum |w_d| = \min_t \sum t_d$

Replacing the above result in $(eq.1)$ we get the final equation that we have to arrive at

$$J(w) = \min_w -2Y^T X w + w^T X^T X w + \lambda \sum_1^D t_d$$

Simplifying this equation we get the final equation that we have to arrive at after substituting the matrix values in the QP form.

$$J(w) = \min_w -2Y^T X w + w^T X^T X w + \lambda \mathbf{1}^T t \text{ ------------ } (eq.2)$$

Again, the form we need is as follows:

$$\min_u \frac{1}{2} u^T Q u + c^T u \ ; subject\ to\ Au \leq b \text{ ----------- (eq.3)}$$

To get this form we consider our u vector to be in the form:

$$u = \begin{pmatrix} w \\ t \end{pmatrix} \in R^{2D}$$

Now, looking at the QP equation form, we can write Q and C so that we get the form of $J(w)$ from $(eq.2)$.

$$Q = \begin{pmatrix} 2X^T X & 0 \\ 0 & 0 \end{pmatrix} \in R^{2Dx2D}$$

$$c = \begin{pmatrix} -2X^T y \\ \lambda \mathbf{1} \end{pmatrix} \in R^{2Dx2D}$$

Also, to satisfy the constraint of: $|w_d| \le t_d$, we can write A and b as follows:

$$A = \begin{pmatrix} I_D & 0 \\ 0 & -I_D \end{pmatrix} \in R^{2Dx2D}$$

$$b = \begin{pmatrix} t \\ w \end{pmatrix} \in R^{2D}$$

Now, substituting all these values back into the (eq.3), we get the following values:

$$\min_{w,t} \frac{1}{2} (w^T \quad t^T) \begin{pmatrix} 2X^T X & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} w \\ t \end{pmatrix} + (-2Y^T X \quad \lambda \mathbf{1}^T) \begin{pmatrix} w \\ t \end{pmatrix}; st. \begin{pmatrix} I_D & 0 \\ 0 & -I_D \end{pmatrix} \begin{pmatrix} w \\ t \end{pmatrix} \le \begin{pmatrix} t \\ w \end{pmatrix}$$

Solving this equation we get out original equation (eq.2):

$$= \min_w -2Y^T Xw + w^T X^T Xw + \lambda \mathbf{1}^T t \ ; st. |w| \le t$$

3. Kernel Ridge Regression

We are given a set of training data $(x_1, y_1), \dots, (x_n, y_n)$, where x_i is in a D dimensional space. Using Linear ridge regression we learn about weights $\boldsymbol{w}$ by optimizing –

$$\min_w \sum_n (y_i - w^T x_i)^2 + \lambda \left||w|\right|^2$$

a) Here we want to write the above optimizing expression's solution in matrix form(analytical solution using inverse).

To start with, we can write the above expression as:

$$J(w) = \frac{1}{2} \left||y - Xw|\right|^2 + \frac{\lambda}{2} \left||w|\right|^2$$

$$= \frac{1}{2} (y - Xw)^T (y - Xw) \quad + \frac{\lambda}{2} w^T w$$

$$= \frac{1}{2} (y^T . y - y^T Xw - w^T X^T y + w^T X^T Xw) + \frac{\lambda}{2} w^T w$$

Now to optimize this function, we take its derivative w.r.t w and set it to zero to find the optimal value for w .

$$J' = \frac{1}{2} (-X^T y - X^T y + 2X^T Xw) + \lambda w = 0$$

$$-X^{Ty} + X^T Xw + \lambda w = 0$$

or we can write this as:

$$-X^T y + (X^T X + \lambda I)w = 0$$

This is simplified as:

$$\underline{w^* = (X^T X + \lambda w)^{-1} X^T y}$$

b) Now we are applying non-linear feature mapping to input samples x_i, given by $\phi(x_i)$, which is now defined in T dimensions(where T >> D).

So, the hypothesis becomes - $w^t \phi(x)$

Writing the new cost function with regularization –

$$J(w) = \frac{1}{2}\sum_n \left(y_n - w^T\phi(x_n)\right)^2 + \frac{\lambda}{2}||w||_2^2$$

Taking derivative of this equation and solving for w, we get –

$$w^{MAP} = \sum_n \frac{1}{\lambda}\left(y_n - w^T\phi(x_n)\right)\phi(x_n)$$
$$= \boldsymbol{\phi}^T \alpha$$

Here we define the first term in the summation as α_n and $\boldsymbol{\phi}$ is the design matrix of transformed features.

Now, writing the function $J(w)$ as:

$$J(w) = \frac{1}{2}||y - \phi w||^2 \quad + \frac{\lambda}{2}\,||w||^2$$

Substituting the value of w^{MAP} in this equation we get:

$$= \frac{1}{2}||y - \phi\phi^T\alpha||^2 + \frac{\lambda}{2}||\phi^T\alpha||^2$$

Taking $K = \phi\phi^T$

$$\frac{1}{2}||y - K\alpha||^2 + \frac{\lambda}{2}||\phi^T\alpha||^2$$

Expanding the terms and using the property that K is symmetric, also removing terms with only y as they will eventually disappear when we take derivative w.r.t α. Writing this as a function of α

$$J(\alpha) = \frac{1}{2}\alpha^T K^2 \alpha - (Ky)^T\alpha + \frac{\lambda}{2}\alpha^T K \alpha$$

Taking derivative of this with α and equating to zero, we get value of α as : (assuming K is invertible)

$$\alpha = (K + \lambda I)^{-1} y$$

Substituting this value back in the w^{MAP} value, we get the optimal w^* as follows:

$$w^* = \phi^T (K + \lambda I)^{-1} y$$

Substituting the value of K back into this equation we get:

$$\underline{w^* = \phi^T (\phi\phi^T + \lambda I)^{-1} y}$$

c) Now given a new testing sample in the transformed space as $\phi(x)$, we need to find the predicted value $\hat{y}$. Using the hypothesis for non-linear mapping, this can be written as:

$$\hat{y} = {w^*}^T \phi(x)$$

$$= (\phi^T (K + \lambda I)^{-1} y)^T \phi(x)$$

$$= y^T (K + \lambda I)^{-1} \boldsymbol{\phi} \phi(x)$$

Above we used the property that K is symmetric. Also we can represent $\boldsymbol{\phi}\phi(x)\ as\ k(x)$ column vector, where $k(x)_n = \boldsymbol{\phi}(\boldsymbol{x_n})^T \phi(x)$. Doing this we get :

$$\underline{\hat{y} = y^T (K + \lambda I)^{-1} k(x)}$$

The above equation also shows that for predicted value we only need the dot product of kernel values.

d) Take: M= Number of features, N= Number of examples
Firstly for Linear Ridge Regression-
The most computationally expensive step is to compute the w parameters using the inverse. Here we find this inverse of $(X^T X + \lambda w)^{-1}$. The final matrix in this expression will have a dimension of (MXM). So the computational complexity will be –

$$O(M^3)$$

Now, for Kernel Ridge Regression-
The most computationally expensive step is to invert the K(gram) matrix. Its dimensions are (NXN). So the computational complexity is –

$$O(N^3)$$

In most cases we have $N \gg M$, so Linear Ridge Regression will fair better than Kernel Ridge Regression.

4. Kernel Construction

In this question we are given two kernel functions- k_1 and k_2. We are also given other functions ($k_3, k_{4,} k_5, k_6, k_7$), which we have to prove are also kernel functions. We will use the property that if the Kernel matrix(K_x) for a function(k_x) is <u>symmetric</u> and <u>positive semi-definite</u> than it is a kernel function.

a) Proof of symmetry

$$k_3(x, x') = a_1 k_1(x, x') + a_2 k_2(x, x')$$

Now, since k_1 and $k_2$are symmetric, so we can write as:

$$= a_1 k_1(x', x) + a_2 k_2(x', x)$$
$$= k_3(x', x)$$

This shows that $k_3$is symmetric.

Proof of PSD: let there be a vector U of real numbers.

Now,

$$U^T K_3 U = \ U^T(a_1 K_1 + a_2 K_2)U$$

This can be written as:

$$= a_1(U^T K_1 U) + a_2(U^T K_2 U)$$

Now, since both K_1 and K_2 are given kernel matrices, so they are PSD and their respective terms are (≥ 0). Thus their sum is also (≥ 0).

Thus,

$$U^T K_3 U \geq 0$$

b) Proof of symmetry

$$k_4(x, x') = f(x)f(x')$$

We can change the order since the result would remain same:

$$= f(x')f(x)$$
$$= k_4(x', x)$$

Proof of PSD: let there be a vector U of real numbers, and also let us write ($f(x) = y, f(x') = y'$)

Now kernel function becomes $k_4(x, x') = yy'$ and the corresponding kernel matrix can be written as $K_4 = YY^T$.

Multiplying U^T and U to K_4 we get:

$$U^T K_4 U = U^T Y Y^T U$$
$$= (U^T y)^2 \geq 0 \quad \text{(Since } Y^T U \text{ is scalar)}$$

d) Proof of symmetry

$$k_6(x, x') = k_1(x, x')k_2(x, x')$$
$$= k_1(x', x)k_2(x', x) \quad \text{(since both are kernels)}$$
$$= k_6(x', x)$$

Proof of PSD: let there be a vector U of real numbers. Also suppose a notation $k(i, j) = K_{ij}$

$$U^T K_6 U = \sum_{ij} u_i u_j K_{6_{ij}} = \sum_{ij} u_i u_j K_{1_{ij}} K_{2_{ij}}$$

Now we know that any symmetric PSD matrix can be written in the following way: $K = A^T A$, and its entries can be written as- $K_{ij} = A_i^T A_j = \sum_m a_{im} a_{jm}$

Using this property and factoring $K_1 = B^T B \ and \ K_2 = C^T C$, and substituting the corresponding summation values into above equation, we get following equation:

$$U^T K_6 U = \sum_{ij} u_i u_j \, (\sum_k b_{ik} b_{jk})(\sum_l c_{il} c_{jl})$$

Rearranging the summations:

$$= \sum_{kl} (\sum_i u_i b_{ik} c_{il})(\sum_j u_j b_{jk} c\_jl)$$

The two terms in the above equation are equal and thus when multiplied we will get a square term. ($Q = \sum_i u_i b_i c_i = \sum_j u_j b_j c_j$)

Thus,

$$U^T K_6 U = \sum_{kl} Q^2 \geq 0$$

c) Proof of symmetry

$$\begin{aligned} k_5(x,x') &= g(k_1(x,x')) \\ &= g(k_1(x',x)) \text{ (since } k_1 \text{is symmetric)} \\ &= k_5(x',x) \end{aligned}$$

Proof of PSD: let there be a vector U of real numbers. We are also given that $g()$ is a polynomial function with positive coefficients. Thus we can write $g(y)$ as:

$$k_5(x,x') = g(y) = w_0 + w_1 y + w_2 y^2 + \cdots \text{, where } y = k_1(x,x')$$
$$k_5(x,x') = w_0 + w_1 k_1(x,x') + w_2 k_1(x,x') k_1(x,x') + \cdots$$

Multiplying U^T and U to K_5 we get :

$$U^T K_5 U = \sum_{ij} u_i u_j K_{5_{ij}}$$
$$= w_0 \sum_{ij} u_i u_j + w_1 \sum_{ij} u_i u_j K_{1_{ij}} + w_2 \sum_{ij} u_i u_j K_{1_{ij}} K_{1_{ij}} + \cdots$$

Now, given all coefficients are positive, the first two terms are greater than or equal to zero, and using the result **from (d) part**, we know that third term is also (≥ 0). Hence we can write:

$$U^T K_5 U \geq 0$$

e) Proof of symmetry

$$\begin{aligned} k_7(x,x') &= \exp(k_1(x,x')) \\ &= \exp\big(k_1(x',x)\big) \text{, Since } k_1 \, is\ a\ kernel\ function \\ &= k_7(x',x) \end{aligned}$$

Proof of PSD: let there be a vector U of real numbers. We can expand exp function as a polynomial with positive coefficients as follows :

$$k_7 = \exp\big(k_1(x,x')\big) = 1 + k_1(x,x') + \frac{k_1\ (x,x')}{2!} + \cdots$$

Now, **from (c) part** we know that any polynomial function of a kernel with positive coefficients gives a kernel function. Hence k_7 is also a kernel function.

5. Programming – Bias/Variance Tradeoff (Matlab Version used: R2013b)

a) Results for Samples Size = 10

For g_1(x): Bias = 0.469464, Variance = 0.000000

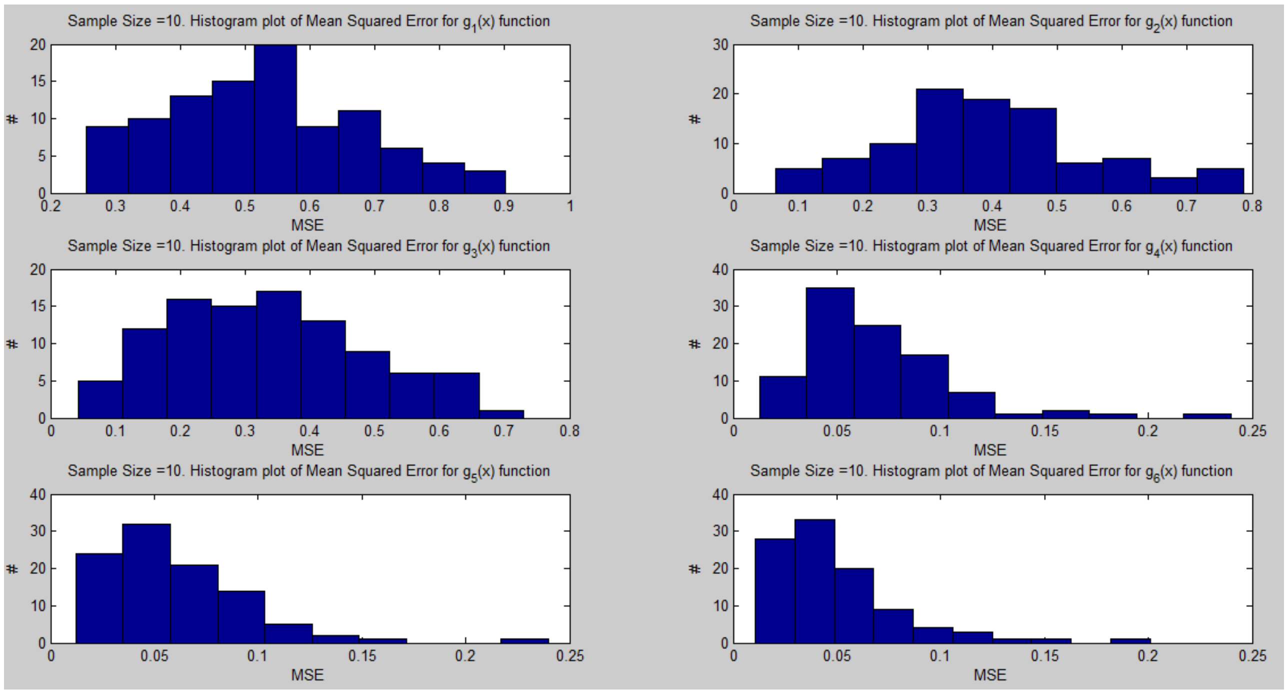
For g_2(x): Bias = 0.333116, Variance = 0.000000

For g_3(x): Bias = 0.295053, Variance = 0.057994

For g_4(x): Bias = 0.027976, Variance = 0.359481

For g_5(x): Bias = 0.042855, Variance = 0.376013

For g_6(x): Bias = 0.053914, Variance = 0.388301



b) Results for Samples Size = 100

For g_1(x): Bias = 0.464225, Variance = 0.000000

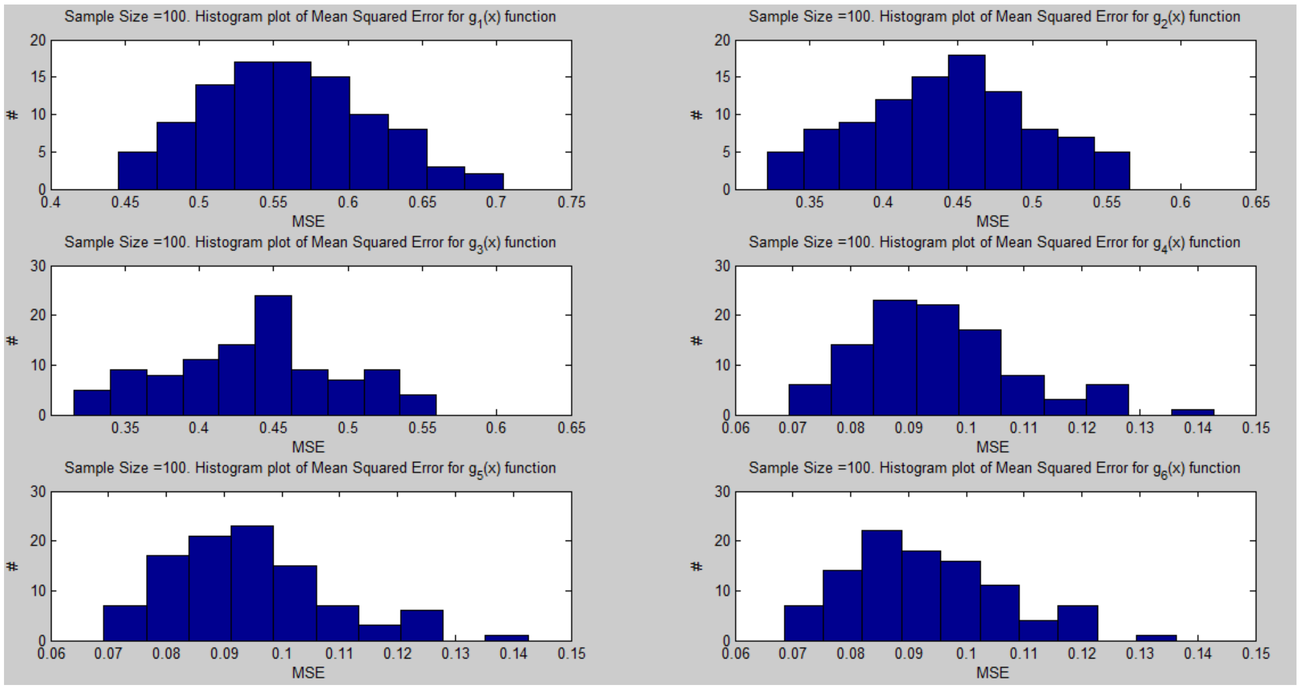
For g_2(x): Bias = 0.354198, Variance = 0.000000

For g_3(x): Bias = 0.348809, Variance = 0.006831

For g_4(x): Bias = 0.003353, Variance = 0.364266

For g_5(x): Bias = 0.004179, Variance = 0.365099

For g_6(x): Bias = 0.005086, Variance = 0.366016



c) <u>Impact of Model complexity and Sample size on Bias and variance</u>

Model Complexity:

Bias- For both sample sizes, as model complexity increases the bias decreases and tends to zero. Although sometimes due to random nature of selected datasets we don’t have a monotonic relationship.

Variance- For both sample sizes, as model complexity increases the variance increases. Although sometimes due to random nature of selected datasets we don’t have a monotonic relationship.

Sample Size:

Bias – Increasing sample size does have some minute effect on Bias, but the impact is random.

Variance – Increasing sample size does decrease the variance.

d) h(x) with regularization

We can see a noticeable increase in Bias as we increase Lambda value. There is little change in variance because it already has a low value, which is because the h(x) here is a quadratic function and fits out true function perfectly.

For Lambda = 0.01 : Bias = 0.000056, Variance = 0.003005
For Lambda = 0.10 : Bias = 0.000029, Variance = 0.003063
For Lambda = 1.00 : Bias = 0.003520, Variance = 0.002564
For Lambda = 10.00 : Bias = 0.091832, Variance = 0.003774

6. Programming – Regression

Below are the various values required by this question:

```
-- Linear Ridge Regression --
For Random Split: 1 - Optimal Lambda value = 0.1000
For Random Split: 2 - Optimal Lambda value = 0.1000
For Random Split: 3 - Optimal Lambda value = 0.1000
For Random Split: 4 - Optimal Lambda value = 0.1000
For Random Split: 5 - Optimal Lambda value = 0.0100
For Random Split: 6 - Optimal Lambda value = 0.0000
For Random Split: 7 - Optimal Lambda value = 0.0000
For Random Split: 8 - Optimal Lambda value = 0.0000
For Random Split: 9 - Optimal Lambda value = 0.1000
For Random Split: 10 - Optimal Lambda value = 0.0000
Average Test Error over all 10 random splits = 0.015619
```

```
-- Kernel Ridge Regression --

a) Linear kernel
For Random Split: 1 - Optimal Lambda value = 0.1000
For Random Split: 2 - Optimal Lambda value = 0.0001
For Random Split: 3 - Optimal Lambda value = 0.0001
For Random Split: 4 - Optimal Lambda value = 0.1000
For Random Split: 5 - Optimal Lambda value = 0.1000
For Random Split: 6 - Optimal Lambda value = 0.1000
For Random Split: 7 - Optimal Lambda value = 0.0001
For Random Split: 8 - Optimal Lambda value = 0.0100
For Random Split: 9 - Optimal Lambda value = 0.0000
For Random Split: 10 - Optimal Lambda value = 0.1000
Average Test Error over all 10 random splits = 0.015550

b) Polynomial kernel
For Random Split: 1 - Optimal Lambda value = 1.0000, a value = 1.0, b value = 2
For Random Split: 2 - Optimal Lambda value = 1.0000, a value = 1.0, b value = 2
For Random Split: 3 - Optimal Lambda value = 1.0000, a value = 1.0, b value = 2
For Random Split: 4 - Optimal Lambda value = 1.0000, a value = 1.0, b value = 2
For Random Split: 5 - Optimal Lambda value = 0.1000, a value = 0.0, b value = 2
For Random Split: 6 - Optimal Lambda value = 1.0000, a value = 1.0, b value = 2
For Random Split: 7 - Optimal Lambda value = 0.0100, a value = 1.0, b value = 2
For Random Split: 8 - Optimal Lambda value = 1.0000, a value = 1.0, b value = 2
For Random Split: 9 - Optimal Lambda value = 1.0000, a value = 1.0, b value = 2
For Random Split: 10 - Optimal Lambda value = 1.0000, a value = 1.0, b value = 2
Average Test Error over all 10 random splits = 0.013638

c) Gaussian RBF kernel
For Random Split: 1 - Optimal Lambda value = 0.0100, Variance value = 8.000
For Random Split: 2 - Optimal Lambda value = 0.0010, Variance value = 8.000
For Random Split: 3 - Optimal Lambda value = 0.0010, Variance value = 8.000
For Random Split: 4 - Optimal Lambda value = 0.0100, Variance value = 8.000
For Random Split: 5 - Optimal Lambda value = 0.0100, Variance value = 4.000
For Random Split: 6 - Optimal Lambda value = 0.0100, Variance value = 8.000
For Random Split: 7 - Optimal Lambda value = 0.0100, Variance value = 8.000
For Random Split: 8 - Optimal Lambda value = 0.0100, Variance value = 8.000
For Random Split: 9 - Optimal Lambda value = 0.0010, Variance value = 8.000
For Random Split: 10 - Optimal Lambda value = 0.0010, Variance value = 8.000
Average Test Error over all 10 random splits = 0.010782
```

<u>Comparison</u> – The average test error for Linear Ridge Regression and Kernel Ridge Regression with Linear Kernel is almost the same. Among the three Kernels, the Gaussian RBF kernel is doing the best.