1 Linear Regression

1.1 Linear Regression Basic

a. Assumption
1) Weak exogeneity.
the predictor variables x can be treated as fixed values, rather than random variables.
2) Linearity.
The mean of the response variable is a linear combination of the parameters (regression coefficients) and the predictor variables.
3) Constant variance (a.k.a. homoscedasticity).
Different values of the response variable have the same variance in their errors, regardless of the values of the predictor variables.
4) Independence of errors.
This assumes that the errors of the response variables are uncorrelated with each other.
5) Lack of perfect multicollinearity in the predictors.
For standard least squares estimation methods, the design matrix X must have full column rank p; otherwise, we have a condition known as perfect multicollinearity in the predictor variables

b. Matrix representation
$Y = Hw + \epsilon$
where Y is $N \times 1$, H is $N \times D$, w is $D \times 1$, ϵ is $N \times 1$.

c. Cost Function

$$L(\mathbf{w}) = \sum_{i=1}^{N} (\mathbf{y} - \hat{\mathbf{y}})^2 = \sum_{i=1}^{N} (\mathbf{y} - \mathbf{Hw})^2 \tag{1}$$

d. Analytical Solution

$$grad L(\mathbf{w}) = -2\mathbf{X}^T(\mathbf{y} - \mathbf{Hw}) = 0 \tag{2}$$
$$\mathbf{w} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y} \tag{3}$$
$$\tag{4}$$

e. Analysis of Analytical Solution
1) To have $(H^T H)^{-1}$ invertible, the number of observations ¿ the number of features.
2) Requires matrix inverse which is $O(n^3)$, too computationally intensive.
3) Thats why we need to seek for numerical solution, like gradient descent.

f. Gradient descent algorithm

```
Init $w^1 = 0$
while $||\frac{\partial L(\hat{w})}{\partial \hat{w}}||_2 > \epsilon$
For i= 1 to D(loop of features)
    $\frac{\partial L(w_j)}{\partial w_j} = -2\sum_i^N H_{ij}(y_i - \hat{y}_i(w^t))$
```

$w_j^{j+1} = w_j^{j+1} - \eta * \frac{\partial L(w_j)}{\partial w_j}$;
t= t+1;
In order to write the gradient in matrix notation, note

$$\frac{\partial L(w_j)}{\partial w_j} = \qquad program@epstopdf$$

$$= -2\begin{pmatrix} H_{1j} & H_{2j} & H_{3j} & ... & H_{Nj} \end{pmatrix}\begin{pmatrix} y_1 - \hat{y}_1(w^t) \\ y_2 - \hat{y}_2(w^t) \\ y_3 - \hat{y}_3(w^t) \\ ... \\ y_N - \hat{y}_N(w^t) \end{pmatrix}$$

$$\begin{pmatrix} \frac{\partial L(w_1)}{\partial w_1} \\ \frac{\partial L(w_2)}{\partial w_2} \\ \frac{\partial L(w_3)}{\partial w_3} \\ ... \\ \frac{\partial L(w_5)}{\partial w_5} \end{pmatrix} = -2\begin{pmatrix} H_{11} & H_{21} & H_{31} & ... & H_{N1} \\ H_{22} & H_{22} & H_{32} & ... & H_{N2} \\ H_{13} & H_{23} & H_{33} & ... & H_{N3} \\ ... \\ H_{1D} & H_{2D} & H_{3D} & ... & H_{ND} \end{pmatrix}\begin{pmatrix} y_1 - \hat{y}_1(w^t) \\ y_2 - \hat{y}_2(w^t) \\ y_3 - \hat{y}_3(w^t) \\ ... \\ y_N - \hat{y}_N(w^t) \end{pmatrix}$$

$$\frac{\partial L(\mathbf{w})}{\partial \mathbf{w}} = -2H^T(\mathbf{y} - \hat{\mathbf{y}}(\mathbf{w}^t))$$

1.2 Performance Assessment/Model Selection

a. Training/validation/testing data split
1) Fit the model parameters using the training data
2) Select the model that minimize the error function on the validation data set
3) Use the error on the test set as a generalization assessment of the model

b. K fold Cross Validation
K fold cross validation applies in the situation where there is not so much data available so we use different portion of the data as validation set and we evaluate the model multiples times. The method has the following setups:

```
1) Shuffle the data
2) Divide the data into k set, called data[1] data[2]..data[k]
3) For(int i =0; i $\leq$k; i++)
{
    use data[i] as validation set,
    The rest data as training set,
    Fit the model, get $RSS_i$.
}
```

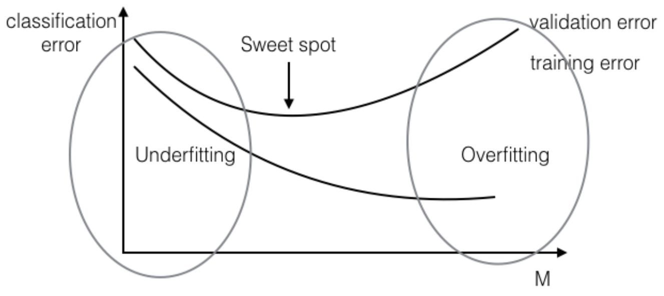
For example, we partition the data into 10 sets, called P_1 to P_{10}. First we use P_1 as validation, the rest as training. Second we use P_2 as validation, the rest as training. Third we use P_3 as validation, the rest as training.
4) Average RSS$_{Aver}(\lambda)$,
5) Repeat the same procedure 1-4 for models.
6) Pick the model that gives the least average RSS_{Aver}.
7) Use this model to train the entire data set.

c. Understanding Bias and Variance Tradeoff
Define $f_{\bar{w}}(x)$ as the fitted value average over all possible values of w, then
The mean square error

$$
\begin{aligned}
& MSE(f_{\hat{w}(train)}(x)) \\
& = E_{training}((f_{w(true)}(x) - f_{\hat{w}}(x))^2) \\
& = E_{training}((((f_{w(true)}(x) - f_{\bar{w}}(x)) + (f_{\bar{w}}(x) - f_{\hat{w}}(x))^2) \\
& = E((f - \bar{f})^2) + 2E((f - \bar{f})(\bar{f} - \hat{f})) + E(\bar{f} - \hat{f})^2 \\
& E((f - \bar{f})^2) = bias^2(f) \\
& E(\bar{f} - \hat{f})^2 = var(\hat{f}) \\
& 2E((f - \bar{f})(\bar{f} - \hat{f})) = 0 \\
& MSE(f_{\hat{w}(train)}(x)) = bias^2 f + var(\hat{f})
\end{aligned}
$$

1) Conclusion
High bias leads to under fitting
High variance leads to over fitting

2) A plot which shows how training error and validation error changes as the model goes more complex



http://www.cs.cornell.edu/courses/cs4780/2015fa/web/lecturenotes/lecturenote13.html

d. Debugging Learining Algorithm Tricks
1) Getting more training examples would be likely to fix high variance
2) Smaller sets of features would be likely to fix high variance
3) Getting additional feature would be likely to fix high bias
4) Decrease penalty parameter would be likely to fix high bias
5) Increase penalty parameter would be likely to fix high variance

1.3 Ridge and Lasso Regression

a. Definition
Ridge uses two norm as penalty and add it into the cost function, $\lambda \sum w_i^2$
Lasso uses one norm as penalty and add it into the cost function, $\lambda \sum (w_i)$

b. Method

1) Ridge regression: Gradient descent

$$\mathbf{Y} = \mathbf{H}\mathbf{w} + \epsilon$$

$$L(\mathbf{w}) = \sum_{i=1}^{N} (\mathbf{y} - \mathbf{H}\mathbf{w})^2 + \lambda \sum w_i^2$$

$$Loss = -2\mathbf{H}^T(\mathbf{y} - \mathbf{H}\mathbf{w}) + 2\lambda w$$

Step update: for $j \neq 0$:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta(-2\mathbf{H}^T(\mathbf{y} - \mathbf{H} * \mathbf{w}) - 2\lambda\mathbf{w})$$

if j =0, as we do not need to add penalty to the constant term:

$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta(-2\mathbf{H}^T(\mathbf{y} - \mathbf{H} * \mathbf{w}))$$

2) Ridge regression: Analytical

$$w = (H^T H + \lambda \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix})^{-1} X^T Y$$

3) Lasso regression: Coordinate descent

$$L(\vec{w}) = \sum_{i=1}^{N} (y - Hw)^2 + \lambda \sum w_i$$