Week 5: Multivariate Linear Regression

Data Science Certificate Program

Ryerson University

Bora Caglayan

12 Feb, 2015

Announcements

- **Reminder:** Homework Deadline March 1st.
- **Midterm:** Feb 26. @ ENG 101.
- *No class next week.*
- Homework announcements:
  - Tip: You can convert the data to cvs first to import easier.

Outline

- Multivariate Linear Regression
- Over-Under Fitting Problem
- Methods to Address Over Fitting
    - Regularized regression
    - Stepwise regression
    - Feature extraction
- Application with R
    - Multivariate regression
    - Design of experiments with CARET package
- Lab

Multivariate Linear Regression

- The purpose of multiple regression is to analyze the relationship between metric or dichotomous independent variables and a metric dependent variable.

- If there is a relationship, using the information in the independent variables will improve our accuracy in predicting values for the dependent variable.

Multivariate Linear Regression

Formally:

- Linear regression model: $$\sum_{i=1}^{N}(\beta_i x_i) + \beta_0 = y$$

Goal:

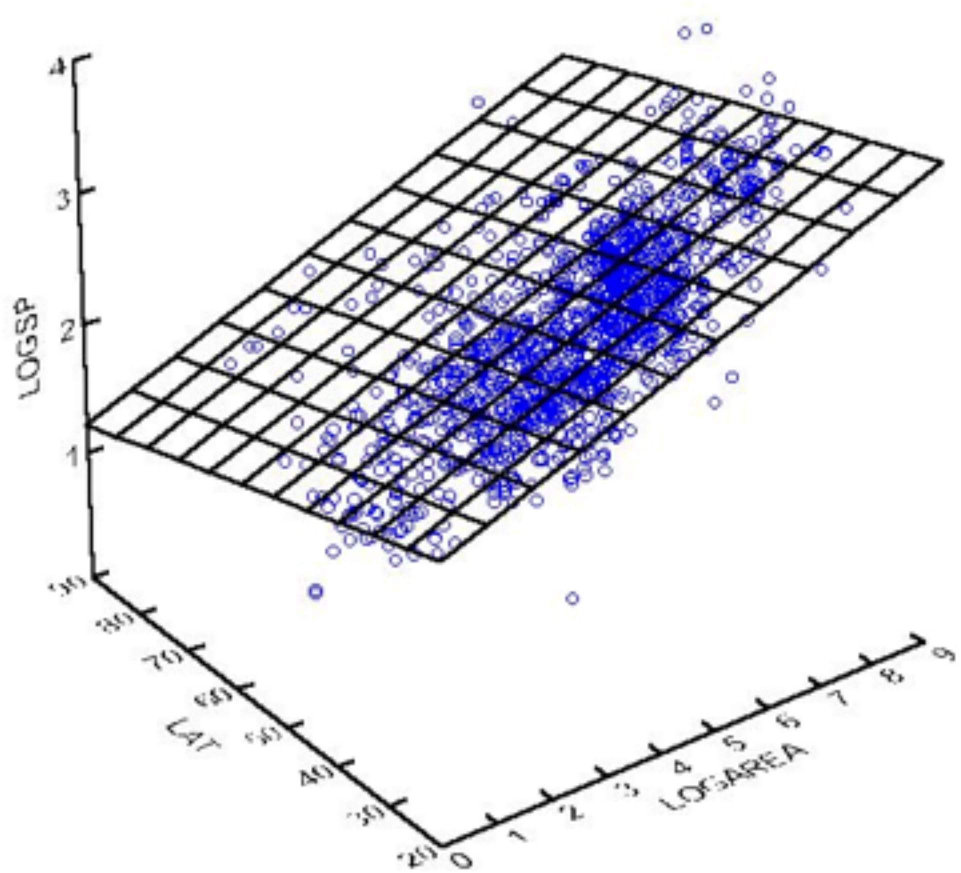
- Find the parameters of beta to minimize the cost function: $$J(\theta) = 1/2 \sum_{i=1}^{N}(h_\theta(x_i) - y_i)^2$$

Assumptions:

- Linear relationship
- No or little multicollinearity among input variables.

Multivariate Linear Regression

2 Variable Case: $\beta_1 x_1 + \beta_2 x_2 + \beta_0 = y$



Implementation

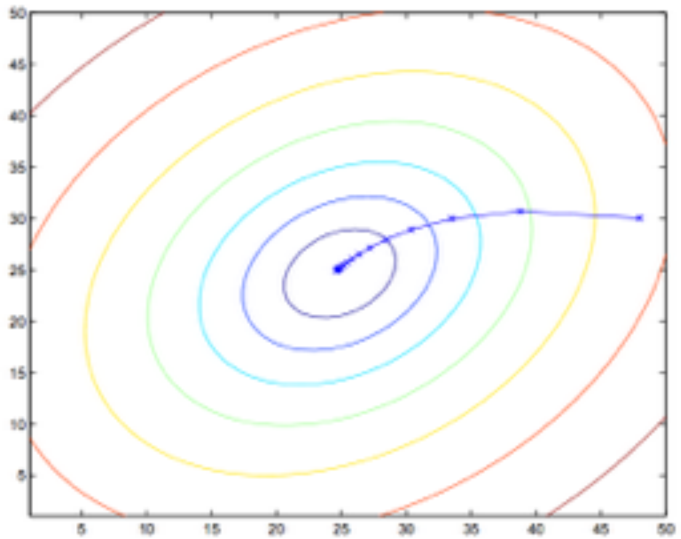
- Gradient descent

  Number of steps is hard to predict.

Repeat until convergence {

$$\theta_j := \theta_j + \alpha \sum_{i=1}^{m} \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)} \qquad \text{(for every } j\text{)}.$$

}



Implementation

- Closed Form Solution

  Closed form: We know the number of steps before computation.

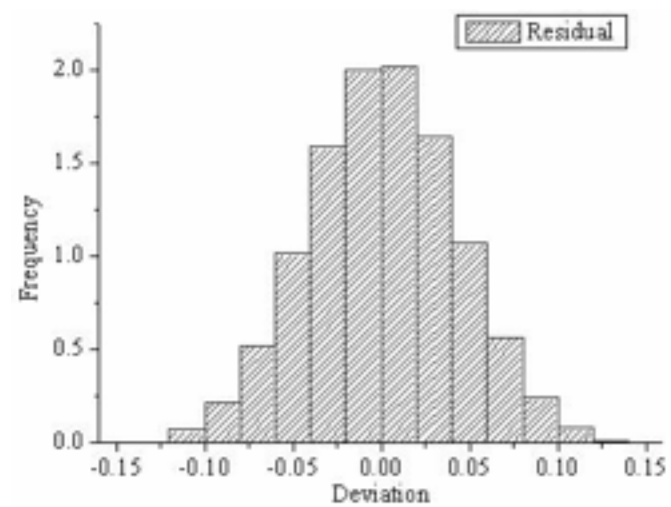
$$\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \quad \mathbf{X} = \begin{pmatrix} \mathbf{x}_1^{\mathrm{T}} \\ \mathbf{x}_2^{\mathrm{T}} \\ \vdots \\ \mathbf{x}_n^{\mathrm{T}} \end{pmatrix} = \begin{pmatrix} x_{11} & \cdots & x_{1p} \\ x_{21} & \cdots & x_{2p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} \end{pmatrix}, \quad \boldsymbol{\beta} = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}, \quad \boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_n \end{pmatrix}.$$

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

Remember the computational complexity discussion from last week…

Performance Measures

- Error distribution



- RMSE (root mean squared error)

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

- PRED(x): Percentage of predictions with at most X percent error rate.

Performance Measures

- We may introduce custom performance measures based on the project.

  - **Project related costs:** A single of inaccurate estimation might be extremely risky.

  - **Benefit:** What is the reward for using the model? Do we increase efficiency, make money etc…

  - **User satisfaction:** Can be estimated offline (using historic data), online (checking user behaviour real-time) or through surveys.

Bias vs Variance

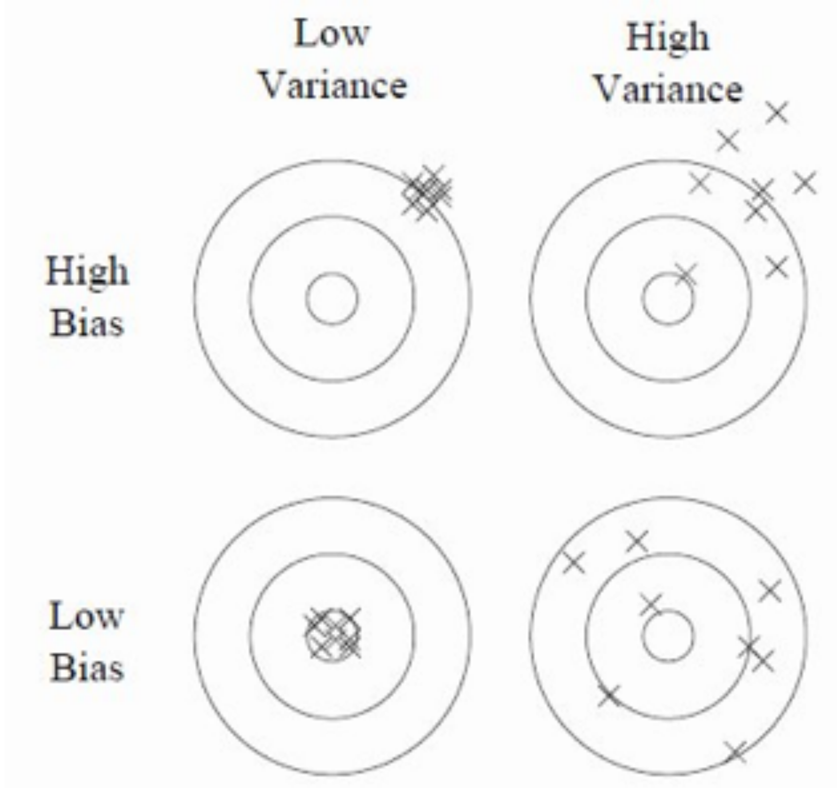
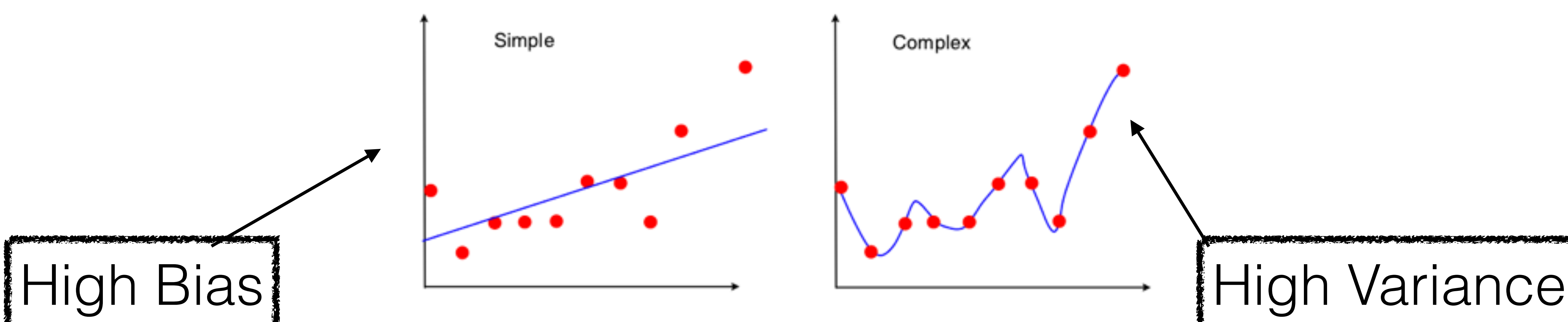


Figure 1: Bias and variance in dart-throwing.



Bias vs Variance

Two types of errors:

1. Irreducible error: Error due to noise in the data etc.

2. Reducible error: Bias + Variance

- Our goal in a predictive model: minimize reducible error

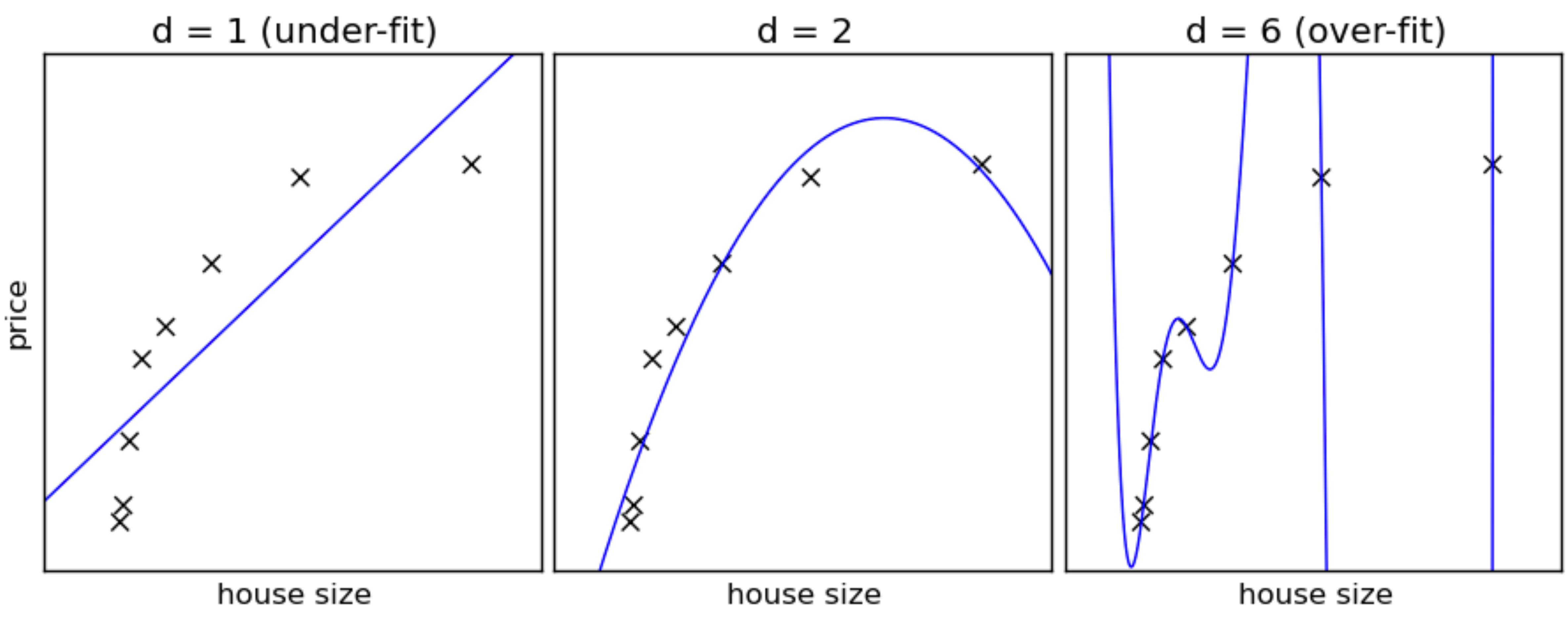
- **Bad news:** There is a tradeoff between bias and variance.

  - Low complexity models have lower variance.

  - High complexity models have higher bias.

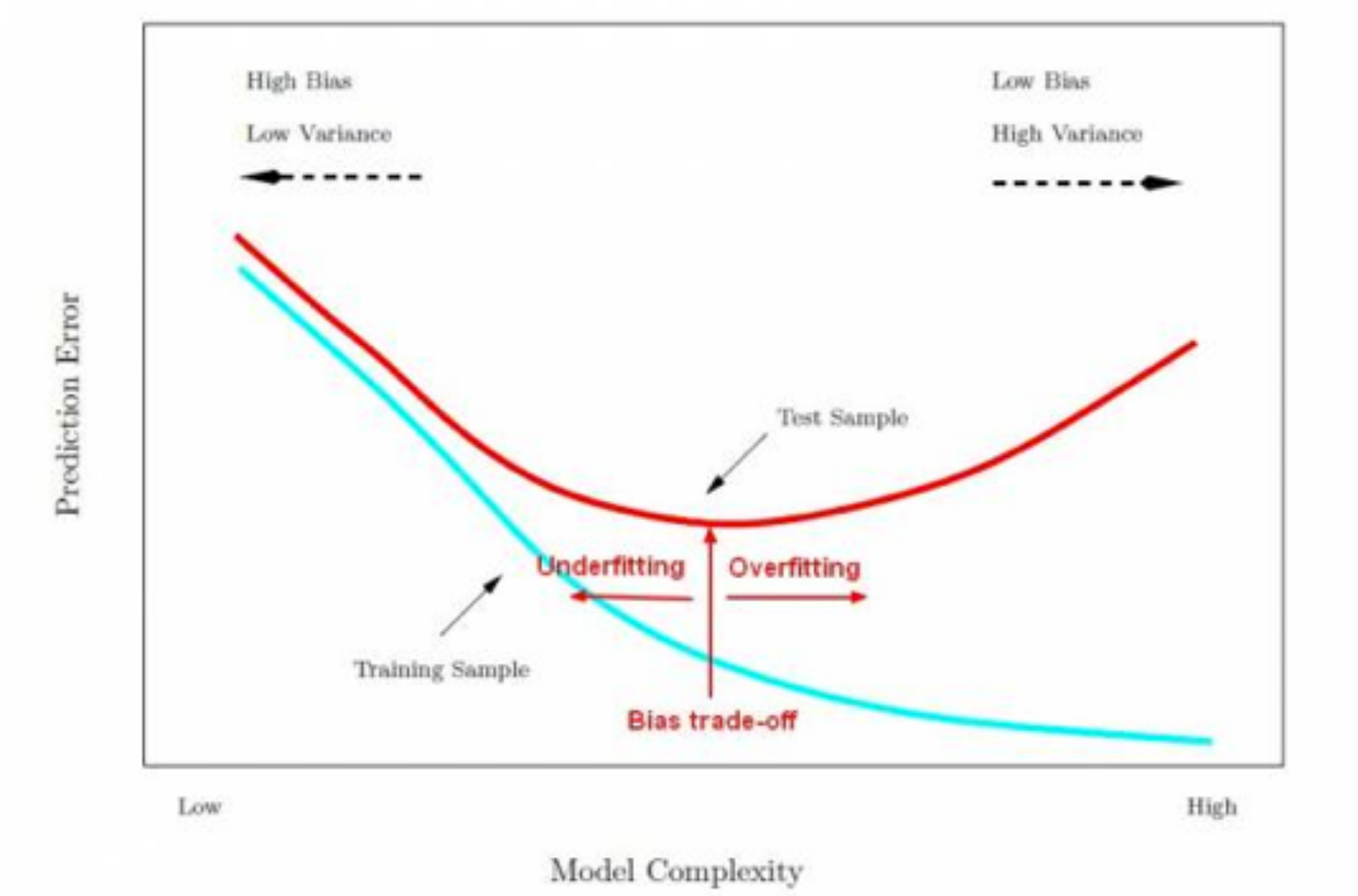
Over-Under Fitting

- Under-fitting: High bias, low variance.
- Over-fitting: Low bias, high variance



Over-Under Fitting

- **Overfitting:**
  - The learner learns the training data ***too*** well.
  - The results may not generalize to the test data.
- **Underfitting:**
  - The learner learns the training data **too** little.
  - The error in both training data and the test data is high.
  - Usually testing error >= training error



Methods to Address Over Fitting Problem

Solutions for Overfitting Problem

1. Regularization:
    - Keep all the features and keep the theta values low.
    - Good if every feature contributes a bit to predicting y.
2. Reduce number of features:
    - Manually choose which features to keep
    - Transform data and extract new features.
    - Model selection.

Regularized Linear Regression

The goal:

Regularization term

Keep coefficients small to reduce sudden changes.

We change the cost function to reduce the coefficient values.

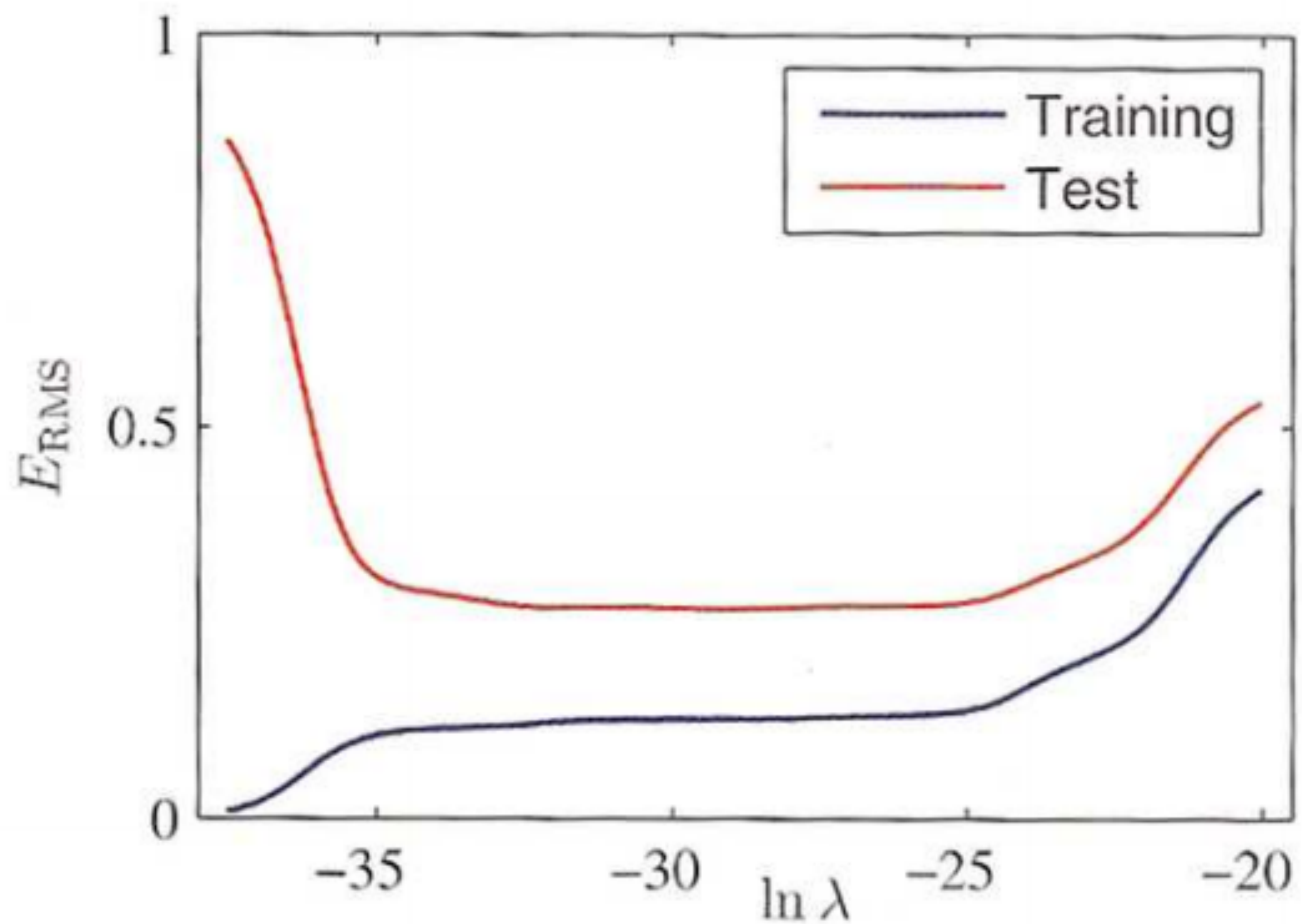
$$J(\theta) = 1/2 \sum_{i=1}^{N} (h_\theta(x_i) - y_i)^2 \longrightarrow J(\theta) = 1/2 \sum_{i=1}^{N} (h_\theta(x_i) - y_i)^2 + \lambda \sum_{j=1}^{M} \theta_j^2$$

Idea: The values of the coefficients are added to the cost. Very high parameters would increase the cost function value.

Remember: The goal of linear regression is minimizing the cost function.

Regularized Linear Regression

λ can be set to avoid under-over fitting.



Regularized Linear Regression

Question

What would happen if regularization coefficient λ is too large?

Curse of Dimensionality

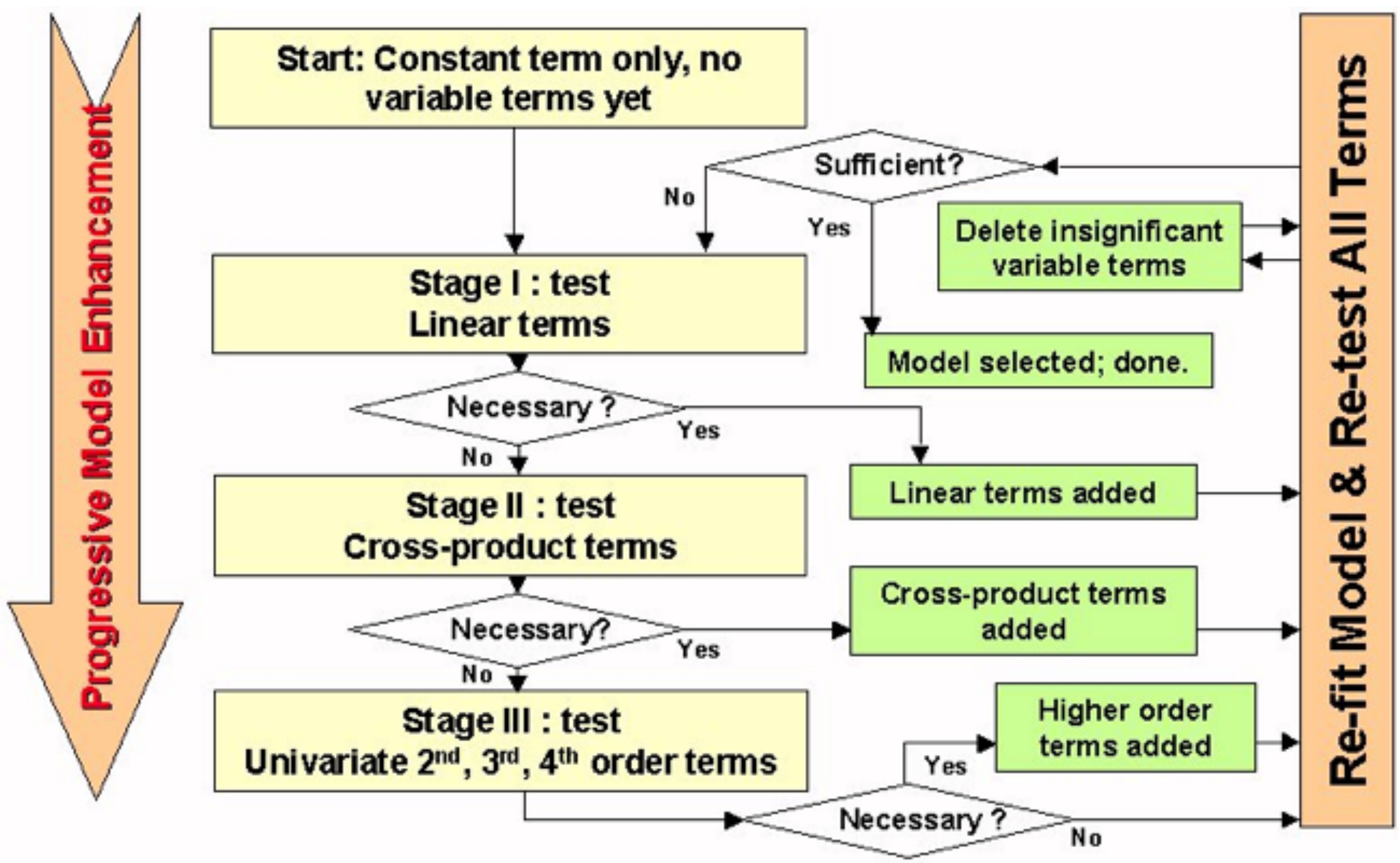
- Increasing the dimensionality of the feature space exponentially increases the data needs.
- **Note:** The dimensionality of the feature space = The number of features.
- What is the message of this?
  - Models should be small relative to the amount of available data.
- Dimensionality Reduction techniques – feature selection – can help.
- Kitchen sink approach (throwing every independent variable into the model) may not be optimal in machine learning models.

Stepwise Linear Regression

Problem: Select minimum number of attributes for the model to avoid curse of dimensionality.

Initiation: Select a performance criteria to optimize. R^2 (goodness of fit), BIC, AIC etc.

Optionally put an *upper* or *lower* limit to the number of variables.



Stepwise Linear Regression

Problem: Select minimum number of attributes for the model to avoid curse of dimensionality.

Strategies:

- **Forward selection (FS)**
  - Start with no variables in the model
    - Test the addition of each variable using a chosen model comparison criterion
    - Add the variable (if any) that improves the model the most.
    - Repeat this process until none improves the model.

- **Backward elemination (StS)**
  - Start with all candidate variables.
  - Test the deletion of each variable using a chosen model comparison criterion.
  - Delete the variable (if any) that improves the model the most by being deleted.
  - Repeat this process until no further improvement is possible.

Possible Problems: Depends on the model performance criteria, biased tests.

Step-wise Linear Regression

Testing every possible combination of variables in linear regression is also called *best-subset regression.*

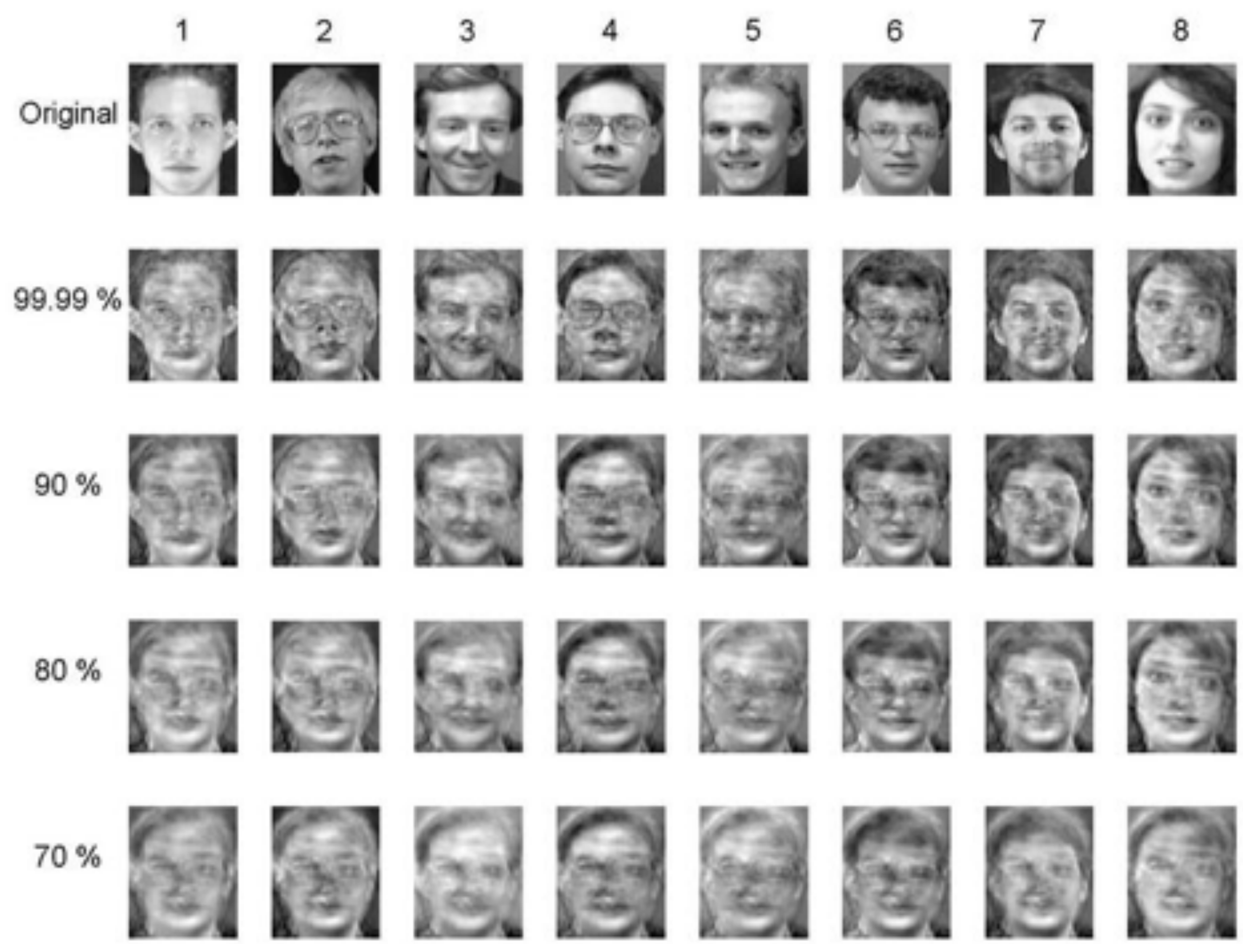
Question

How many combinations of variables can you use in linear regression?

$2^N - 1$

Feature Extraction

Goal: Reduce number of features & collinearity of the data



Feature Extraction

- A sample method for feature extraction: Principle component analysis (PCA)
  - Creates orthogonal features with no collinearity.
  - For most datasets feature size can be reduced dramatically without reducing the information content of the dataset.

Summary

Multi-variate linear regression

- **Definition:** Predict the value of a numeric variable based on more than one input variable.
- **Exploratory analysis:** Check correlations, scatter plot all the pairs.
- **Preprocessing:** Changing scales, *feature selection, feature extraction*
- **Algorithms:** Gradient descent, closed form solution with linear algebra, (optional: changed cost function for *regularized regression, stepwise regression)*.
- **Experiment:** k-fold cross validation, random split
- **Performance criteria:** RMSE, pred(x), error distribution
- **Advantages:** Simplicity, low computation cost
- **Disadvantage:** May not be a good fit for most data.

Non-linear Regression Models

Cross product terms

- Used to model the interaction between two variables.
- Cross product Terms : $x_i * x_j$
- There are $\binom{N}{2} = N * (N-1)/2$ possible cross products
- We can select a subset of cross products by step-wise or best subset regression.

Some Other Regression Models

- **Higher order polynomials:**

Second-order polynomial model in two vars.

$$\begin{aligned} y &= \beta_0 + \beta_1 x_1 + \beta_2 x_2 \\ &\quad + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \beta_{12} x_1 x_2 + \epsilon \end{aligned}$$

  - Linear effect parameters: β_1, β_2
  - Quadratic effect parameters: β_{11}, β_{22}
  - Interaction effect parameters: β_{12}

- **Other *basis* functions**
  - Logit (Week 7) used for classification: $F(x) = \dfrac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$
  - Sinusoidal
  - etc.

Midterm

Questions 1.5 hours, open book

- 1 open-ended question about the homework 1.
- 1 problem from probability tables.
- 1 open-ended question.
- 5 multiple choice questions about R data structures.
- 10 multiple choice questions from experimental design and linear regression.
- 2 multiple choice questions about big O notation.

References

- Andrew Gelman Jennifer Hill. "Data Analysis Using Regression and Multilevel/Hierarchical Models". Cambridge University Press, 2006 http://www.stat.columbia.edu/~gelman/arm/

- [Advanced] “Pattern Recognition and Machine Learning Bishop, C. M. (2006) Springer” http://research.microsoft.com/en-us/um/people/cmbishop/prml/

- http://en.wikipedia.org/wiki/Occam%27s_razor

Week 5 Application Part

February 12, 2015

Preparation

```
library(caret) # experiment design
library(MASS) # stepwise regression
library(leaps) # all subsets regression
library(glmnet) # for regularized linear regression
```

Multivariate Linear Regression

```
model_mlr <- lm(Temp~Month+Wind, data=airquality)
summary(model_mlr)
```

```
Call:
lm(formula = Temp ~ Month + Wind, data = airquality)

Residuals:
     Min       1Q   Median       3Q      Max
-18.2946  -5.2490   0.0679   5.7414  18.5270

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  72.0875     3.9827  18.100  < 2e-16 ***
Month         2.3416     0.4543   5.154 7.92e-07 ***
Wind         -1.0626     0.1827  -5.817 3.49e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
```

Multivariate Linear Regression: Test Example

```r
rn_train <- sample(nrow(airquality),
                   floor(nrow(airquality)*0.8))
train <- airquality[rn_train,]
test <- airquality[-rn_train,]
model_mlr <- lm(Temp~Month+Wind, data=train)
prediction <- predict(model_mlr, interval="prediction",
                      newdata=test)
```

Multivariate Linear Regression: RMSE

```r
sqrt(sum((prediction[,"fit"] - test$Temp)^2)/nrow(test))
```

```
[1] 6.803321
```

Linear Regression: PRED(10)

Find the percentage of cases with less than 10 percent error:

```
errors <- prediction[,"fit"] - test$Temp
rel_change <- 1 - ((test$Temp - abs(errors)) / test$Temp)
table(rel_change<0.10)["TRUE"] / nrow(test)
```

```
     TRUE
0.8387097
```

Stepwise Linear Regression - Forward

Start from a null formula. Go forward.

```
full <- lm(mpg~wt+hp+disp+gear,data=mtcars)
null <- lm(mpg~1,data=mtcars)
stepF <- stepAIC(null, scope=list(lower=null, upper=full),
                 direction= "forward", trace=FALSE)
summary(stepF)
```

```
Call:
lm(formula = mpg ~ wt + hp, data = mtcars)

Residuals:
   Min     1Q Median     3Q    Max
-3.941 -1.600 -0.182  1.050  5.854

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
```

Stepwise Linear Regression - Backward

Start from a full formula. Go back.

```r
full <- lm(mpg~wt+hp+disp+gear,data=mtcars)
stepB <- stepAIC(full, direction= "backward", trace=FALSE)
summary(stepB)
```

```

Call:
lm(formula = mpg ~ wt + hp, data = mtcars)

Residuals:
   Min     1Q Median     3Q    Max 
-3.941 -1.600 -0.182  1.050  5.854 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 37.22727    1.59879  23.285  < 2e-16 ***
wt          -3.87783    0.63273  -6.129 1.12e-06 ***
```

Best Subset

Select the best subset of the variables through exhaustive search.

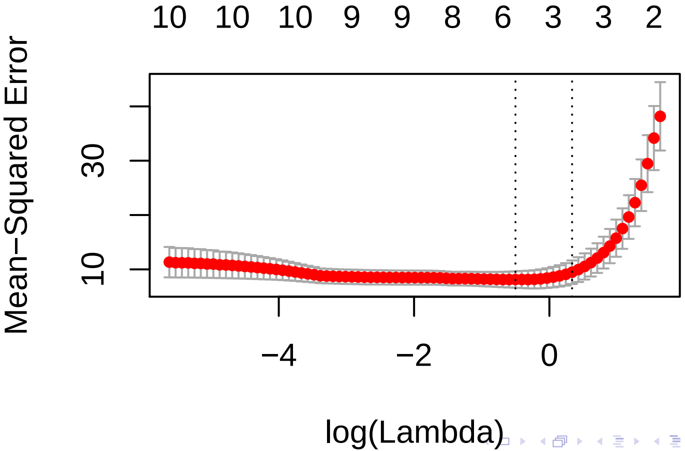
```
subsets<-regsubsets(mpg~wt+hp+disp+gear,data=mtcars,
                    nbest=2)
sub.sum <- summary(subsets)
as.data.frame(sub.sum$outmat)
```

```
         wt hp disp gear
1  ( 1 )  *
1  ( 2 )          *
2  ( 1 )  *  *
2  ( 2 )  *       *
3  ( 1 )  *  *         *
3  ( 2 )  *  *    *
4  ( 1 )  *  *    *    *
```

Regularized Linear Regression

See the effect of λ on test performance.

```
cv.fit <- cv.glmnet(as.matrix(mtcars[,-1]),
                    as.vector(mtcars[,1]), nlambda=100)
plot(cv.fit)
```



Lab

Preparation Data load

```
library(RCurl)
l<-"http://vincentarelbundock.github.io/\
Rdatasets/csv/Ecdat/Computers.csv"
u <- getURL(l)
c_prices <- read.csv(text = u)
```

Lab Questions

Use c_prices data

1- Build a multivariate linear regression model with any 5 attributes to predict price. Compare the performance with univariate regression model.

2- Do stepwise regression (backward/forward) for the model you created for question 1 to check if you can simplify the model.

3- Find the best combination of 5 attributes.

4- Plot regularized linear regression error rate as a function of lambda for your regression model 1.