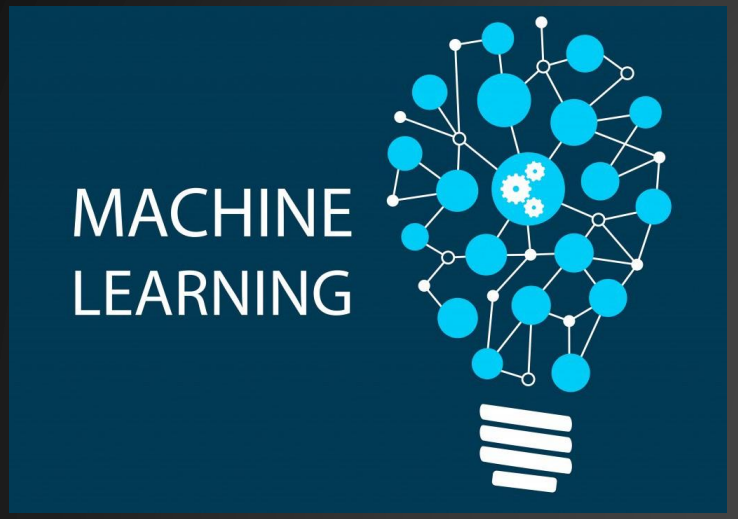
INTRODUCTION TO



FOR

USING

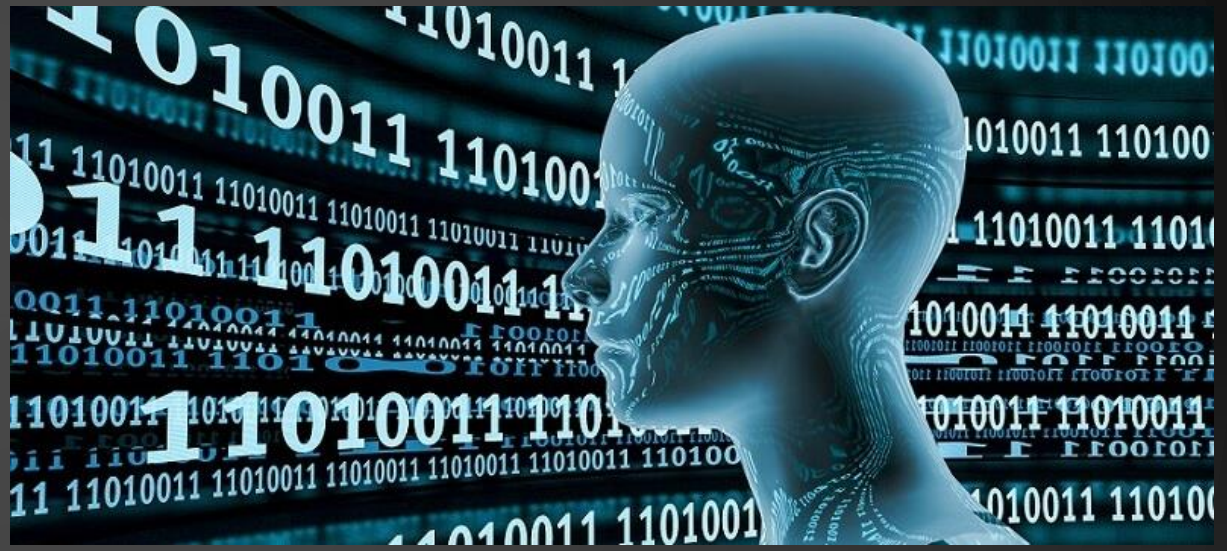
TUE VU

ADVANCED COMPUTING & DATA SCIENCE (ACDS)

CCIT\CITI

OUTLINES

1. Introduction to Machine Learning
2. Why R
3. Types of Machine Learning
4. Caret package
5. Supervised Learning
6. Unsupervised Learning



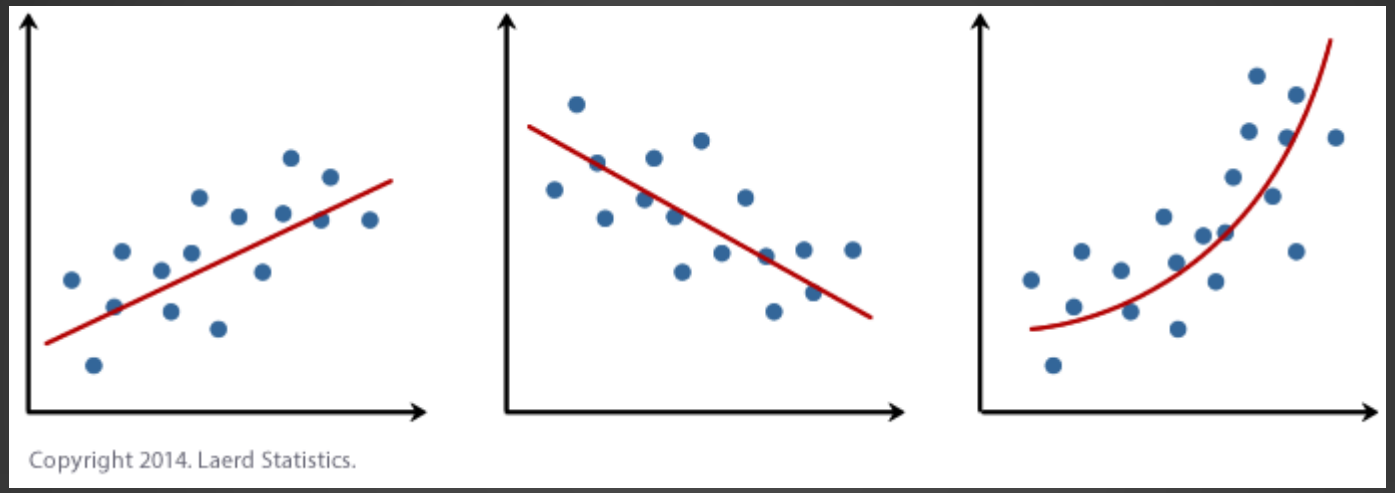
5. Supervised Learning

1. Regression
   - Linear Regression
   - Multi-Linear Regression (MLR)
   - Other typical Linear Regression Technique
   - Logistic Regression
2. Decision Tree
3. Ensemble Prediction
   - Random Forest
   - Bagging
   - Boosting
4. Model based Prediction
   - Naïve Bayes
   - Linear Discriminant Analysis
5. Regularization & Variable selection
   - Ridge Regression
   - LASSO
   - ELASTIC-NET
6. Dimension Reduction
   - Principal Component Analysis
7. Neural Network
8. Support Vector Machine
9. K-Nearest Neighbor

5. Supervised Learning

5.1. Regression based method

5.1.1. Linear Regression



$$y = ax + b$$

5. Supervised Learning

5.1. Regression based method

5.1.2. Multi-Linear Regression – Ordinary Least Square Regression

$$y = a_1x_1 + a_2x_2 + \ldots + a_nx_n + b$$

5. Supervised Learning

5.1. Regression based method

5.1.3. Other typical Linear Regression Technique

Stepwise Linear Regression:
It's a step by step Regression to determine which covariates set best match with the dependent variable. Using AIC as criteria:

method = 'lmStepAIC'

Principal Component Regression:
Linear Regression using the output of a Principal Component Analysis (PCA). PCR is skillful when data has lots of highly correlated predictors
method = 'pcr'

5. Supervised Learning

5.1. Regression based method

5.1.4. Logistic Regression

Logistic regression is another technique borrowed by machine learning from the field of statistics. It is the go-to method for binary classification problems (problems with two class values).

Typical binary classification: True/False, Yes/No, Pass/Fail, Spam/No Spam, Male/Female

5. Supervised Learning

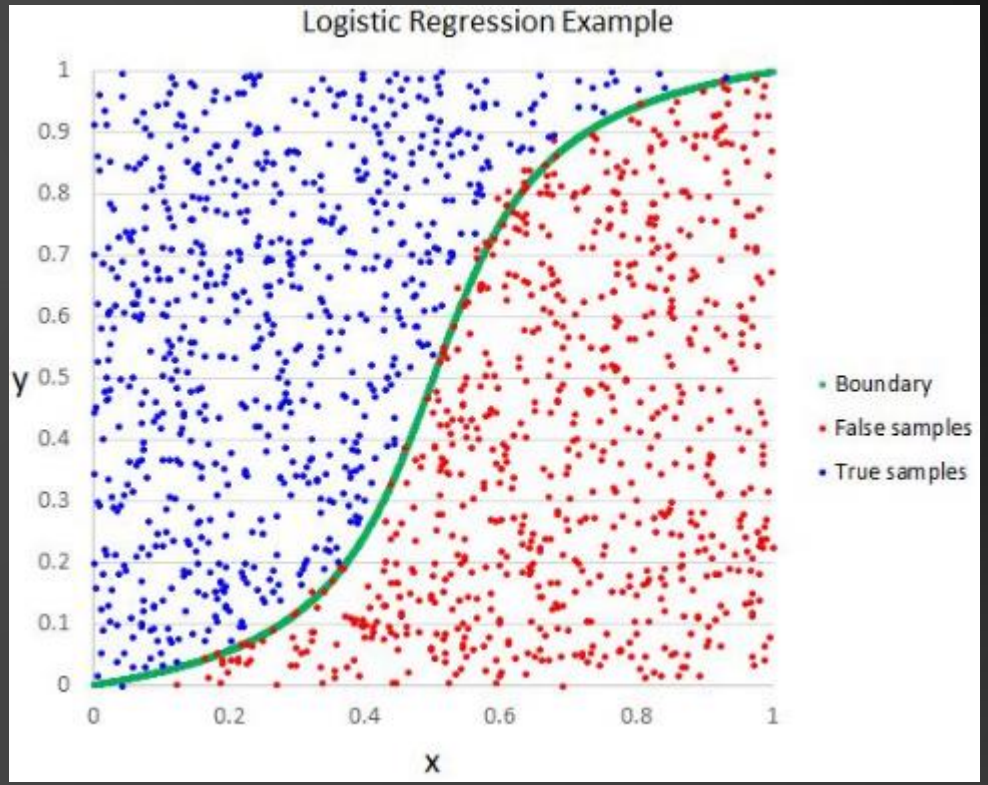
5.1. Regression based method

5.1.4. Logistic Regression

Unlike linear regression, the prediction for the output is transformed using a non-linear function called the logistic function.

The standard logistic function has formulation:

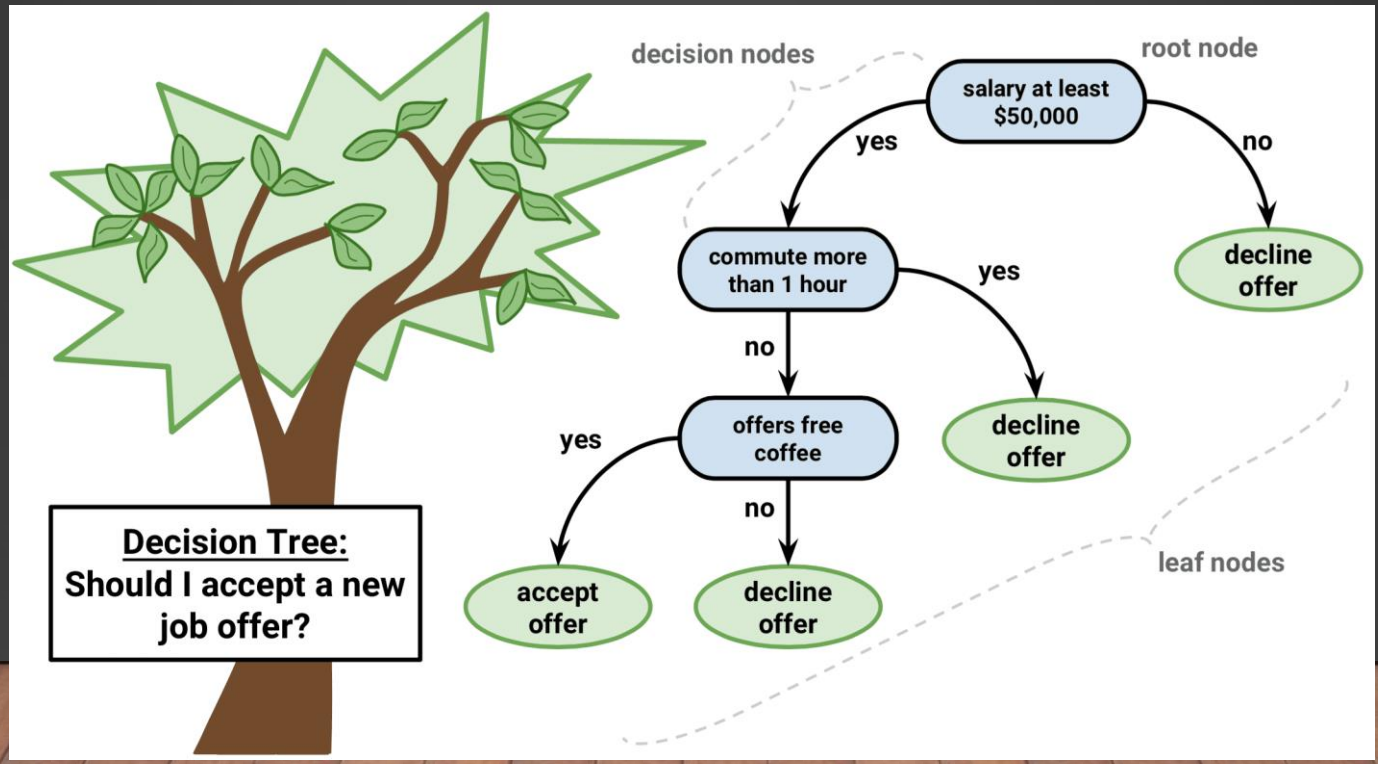
$$f(x) = \frac{1}{1 + e^{-x}}$$



5. Supervised Learning

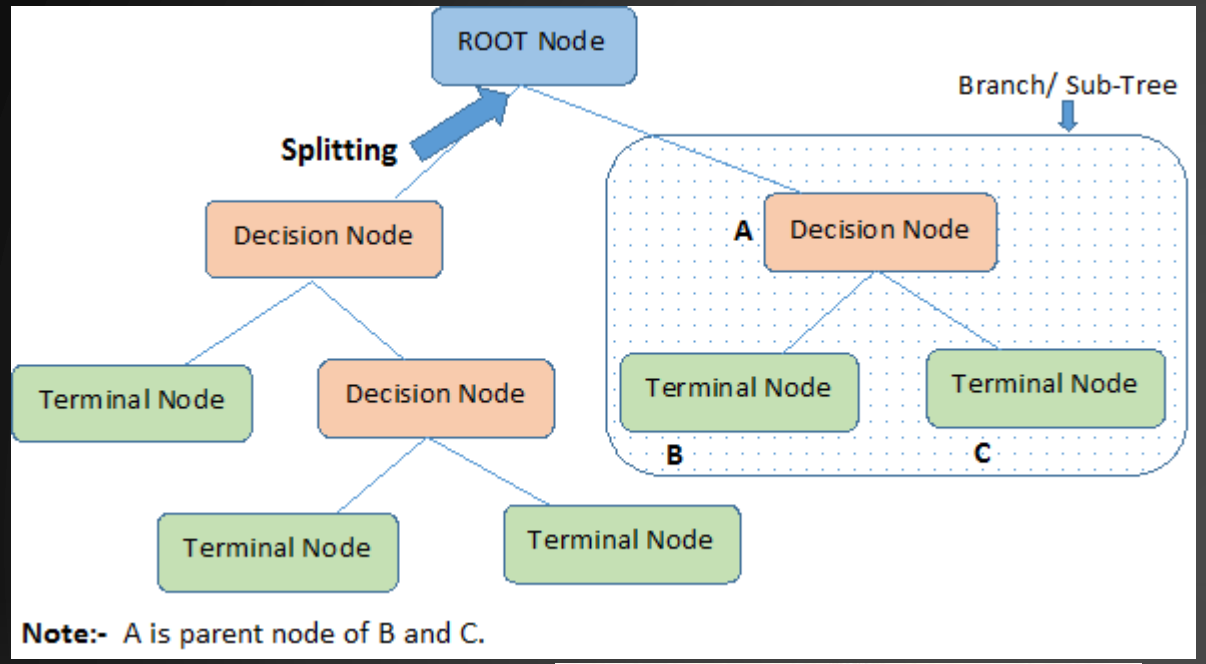
5.2. Decision Tree

- Tree based learning algorithms are considered to be one of the best and mostly used supervised learning methods.
- Tree based methods empower predictive models with high accuracy, stability and ease of interpretation
- Non-parametric and non-linear relationships
- Types: Categorical and Continuous



5. Supervised Learning

5.2. Decision Tree



Terminology

1. Root Node: It represents entire population or sample and this further gets divided into two or more homogeneous sets.
2. Splitting: It is a process of dividing a node into two or more sub-nodes.
3. Decision Node: When a sub-node splits into further sub-nodes, then it is called decision node.
4. Leaf/ Terminal Node: Nodes with no children (no further split) is called Leaf or Terminal node.
5. Pruning: When we reduce the size of decision trees by removing nodes (opposite of Splitting), the process is called pruning.
6. Branch / Sub-Tree: A sub section of decision tree is called branch or sub-tree.
7. Parent and Child Node: A node, which is divided into sub-nodes is called parent node of sub-nodes where as sub-nodes are the child of parent node

5. Supervised Learning

5.2. Decision Tree

Types

- Categorical Regression: "Yes/No", "TRUE/FALSE", "Male/Female", "Colors"
- Continuous Regression: when the dependent variable is continuous

5. Supervised Learning

5.2. Decision Tree

Splitting algorithm

- Gini Impurity: (*Categorical*)
- Chi-Square index (*Categorical*)
- Cross-Entropy & Information gain (*Categorical*)
- Reduction Variance (*Continuous*)

Different packages/methods use different splitting algorithm

- CART (Classification And Regression Tree) use Gini index & Entropy
- ID3 & C5.0 Tree use Entropy & Information gain
- CHAID use Chi-Square

5. Supervised Learning

5.2. Decision Tree

Gini impurity example: (*Categorical*)

$$Gini = 1 - \sum_i p(i)^2$$

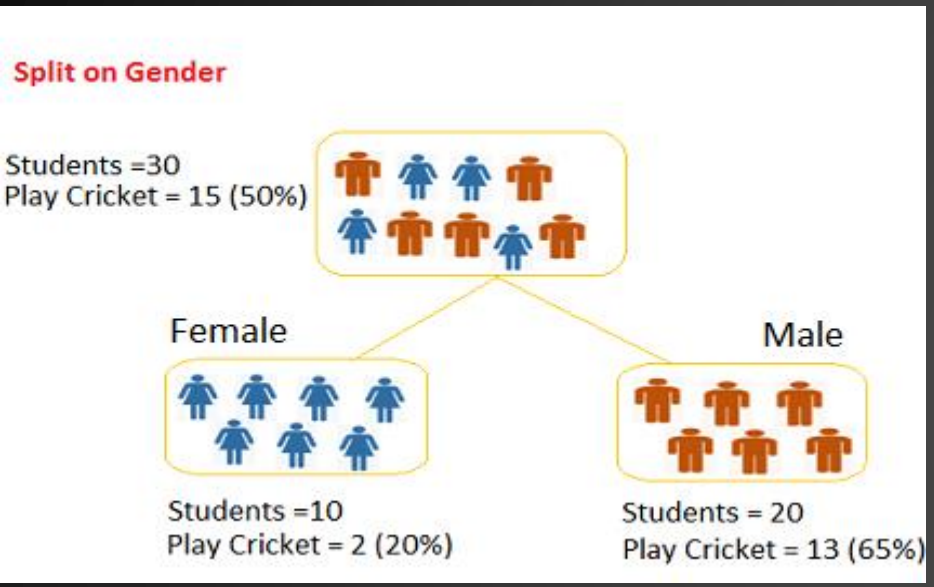
A sample of 30 students with three variables Gender (Boy/ Girl), Class(IX/ X) and Height (5 to 6 ft).
- 15 out of these 30 play cricket in leisure time.
- predict who will play cricket during leisure period?

In this problem, we need to segregate students who play cricket in their leisure time based on highly significant input variable among all three.

5. Supervised Learning

5.2. Decision Tree

Gini impurity example: (*Categorical*)



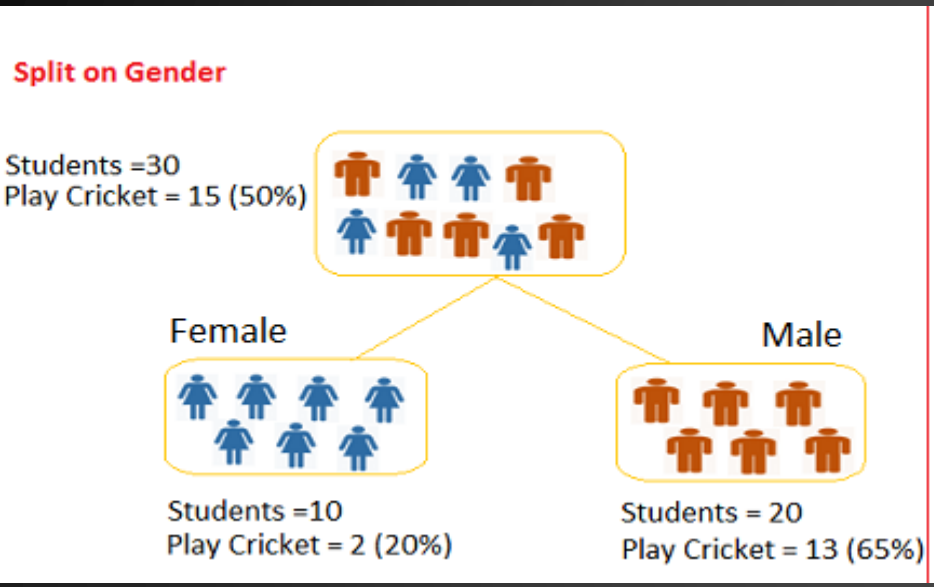
$G_1 = 1-0.2^2-(1-0.2)^2 = 0.32$ $G_2 = 1-0.65^2-(1-0.65)^2 = 0.45$

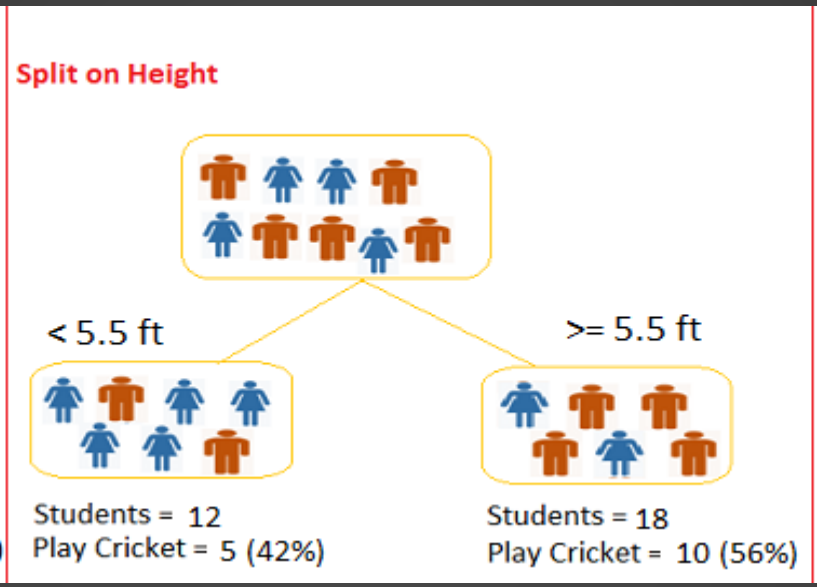
$GG = 10/30*.32+20/30*.45 = \mathbf{0.4}$

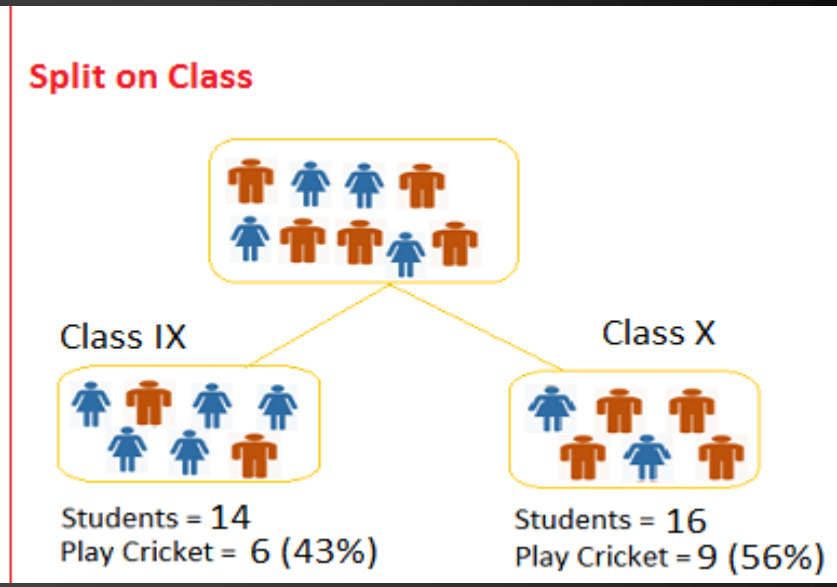
5. Supervised Learning

5.2. Decision Tree

Gini impurity example: (*Categorical*) – The smaller the Gini the better







G_1=1-0.2^2-(1-0.2)^2 = 0.32 G_2=1-0.65^2-(1-0.65)^2 = 0.45 G_3=1-0.42^2-(1-0.42)^2 = 0.5 G_4=1-0.56^2-(1-0.56)^2 = 0.5 G_5=1-0.43^2-(1-0.43)^2 = 0.49 G_4=1-0.56^2-(1-0.56)^2 = 0.50

GG=10/30*.32+20/30*.45=**0.4**

GH = 12/30*.5+18/30*.5=0.5

GC = 14/30*.49+16/30*. 5=0.49

Gini=0 when all observation belongs to one label.

5. Supervised Learning

5.2. Decision Tree

Chi-Square example: (*Categorical*)

$$\chi^2 = \sqrt{\frac{(Actual - Expected)^2}{Expected}}$$

Split by Gender

Node	Play Cricket	Not Play Cricket	Total	Expected Play Cricket	Expected Not Play Cricket	Chi-Square	
						Play Cricket	Not Play Cricket
Female	2	8	10	5	5	1.34	1.34
Male	13	7	20	10	10	0.95	0.95
					Total Chi-Square	4.58	

Gender is more significant than Class

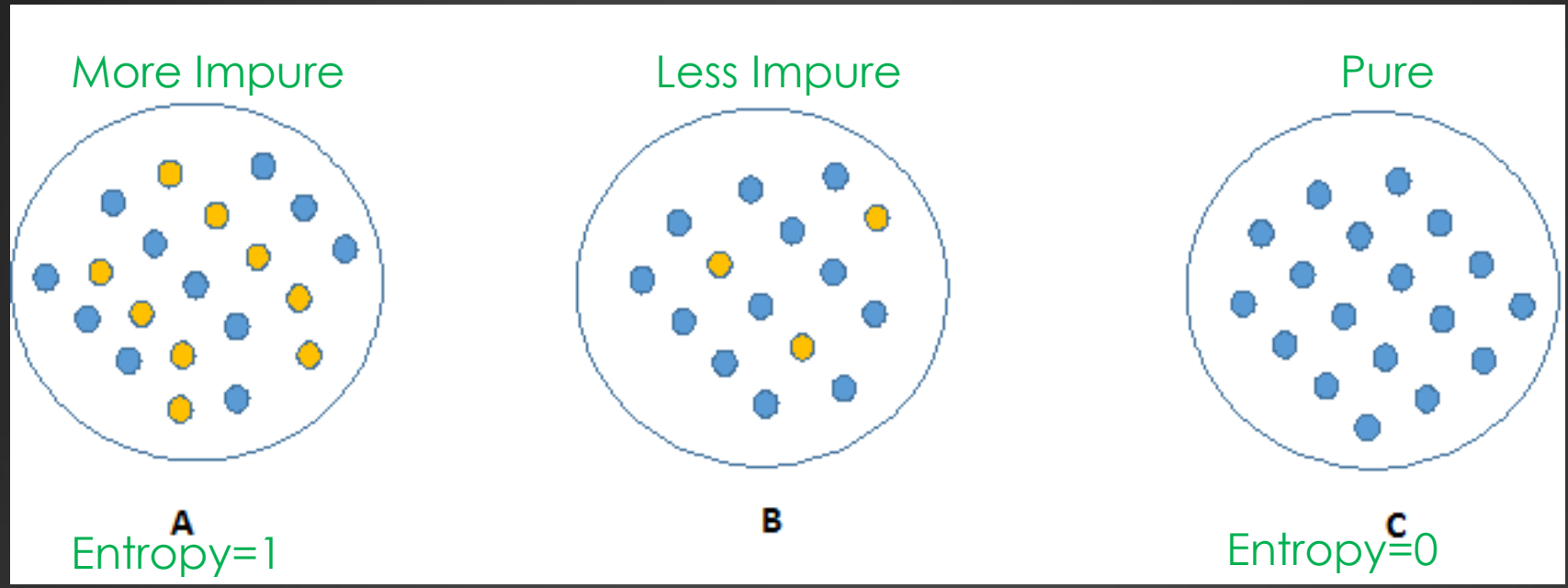
Split by Class

Node	Play Cricket	Not Play Cricket	Total	Expected Play Cricket	Expected Not Play Cricket	Chi-Square	
						Play Cricket	Not Play Cricket
IX	6	8	14	7	7	0.38	0.38
X	9	7	16	8	8	0.35	0.35
					Total Chi-Square	1.46	

5. Supervised Learning

5.2. Decision Tree

Information Gain: Specify by Entropy



$$Entropy = -p\log_2 p - q\log_2 q$$

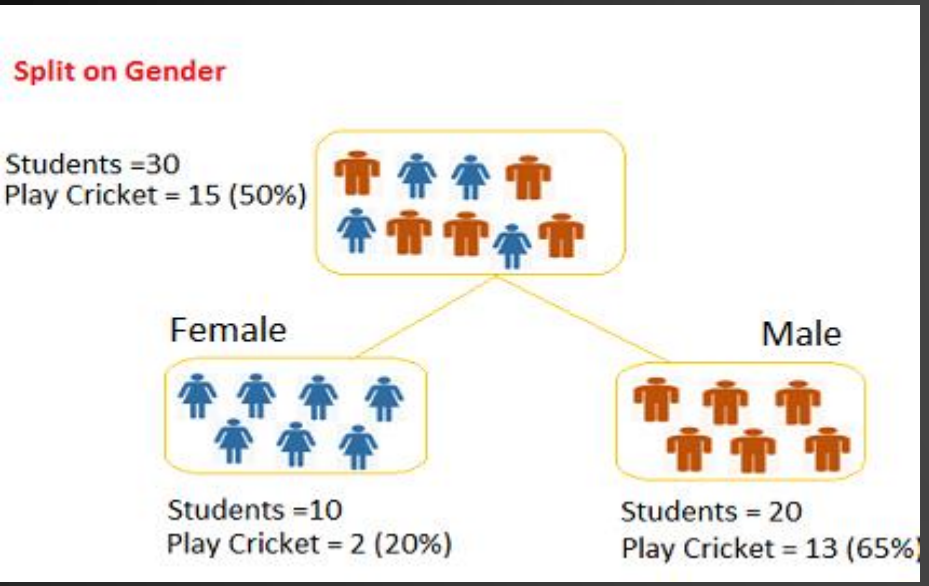
5. Supervised Learning

5.2. Decision Tree

Cross-Entropy example: (*Categorical*) – the smaller the Entropy the better

$$Entropy = -p\log_2 p - q\log_2 q$$

$$Information\ Gain = 1\ - Entropy$$



$E_1 = -0.2\log_2 0.2 - 0.8\log_2 0.8 = 0.72$

$E_2 = -0.65\log_2 0.65 - 0.35\log_2 0.35 = 0.93$

$E_G = \frac{10}{30} * 0.72 + \frac{20}{30} * 0.93 = 0.86$

5. Supervised Learning

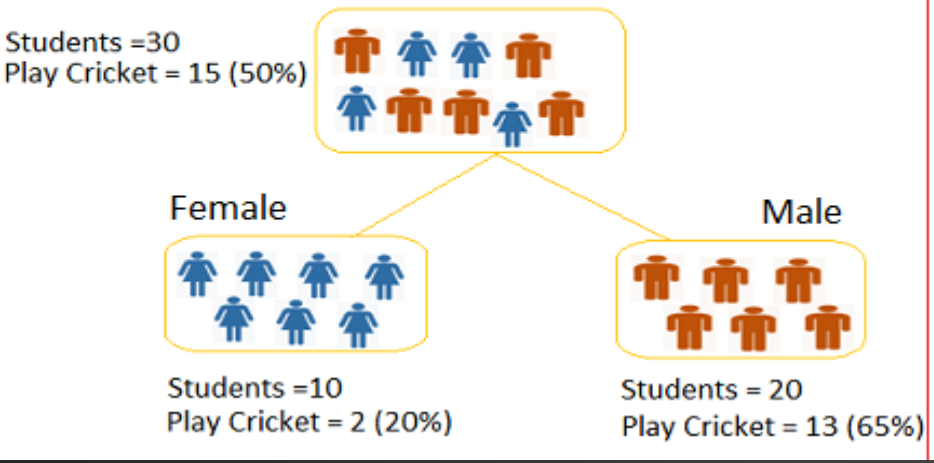
5.2. Decision Tree

Cross-Entropy example: (*Categorical*) – the smaller the Entropy the better

$$Entropy = -p\log_2 p - q\log_2 q$$

$$Information\ Gain = 1 - Entropy$$

Split on Gender

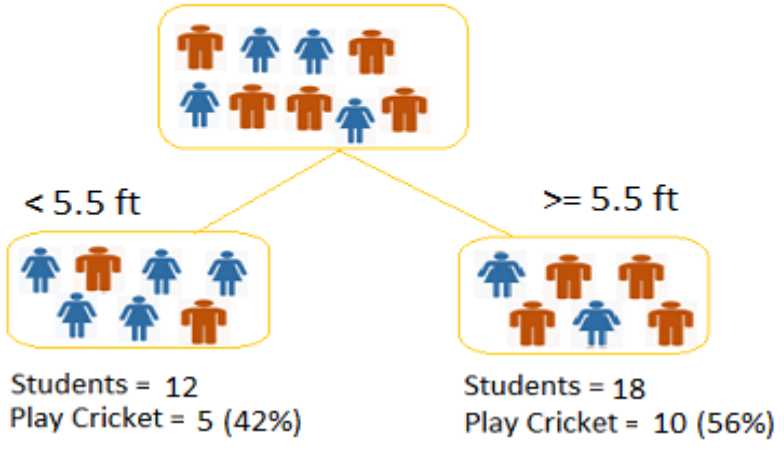


$E_1 = -0.2\log_2 0.2 - 0.8\log_2 0.8 = 0.72$

$E_2 = -0.65\log_2 0.65 - 0.35\log_2 0.35 = 0.93$

$E_G = \frac{10}{30} * 0.72 + \frac{20}{30} * 0.93 = 0.86$

Split on Height

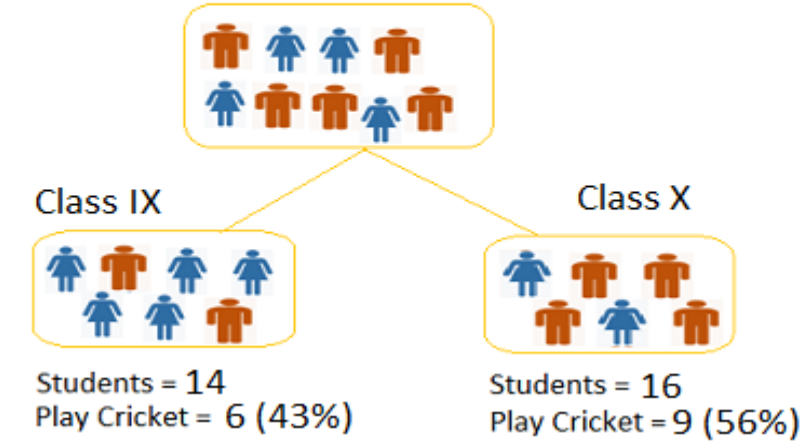


$E_3 = 0.98$

$E_4 = 0.99$

$E_C = 0.98$

Split on Class



$E_3 = 0.99$

$E_4 = 0.99$

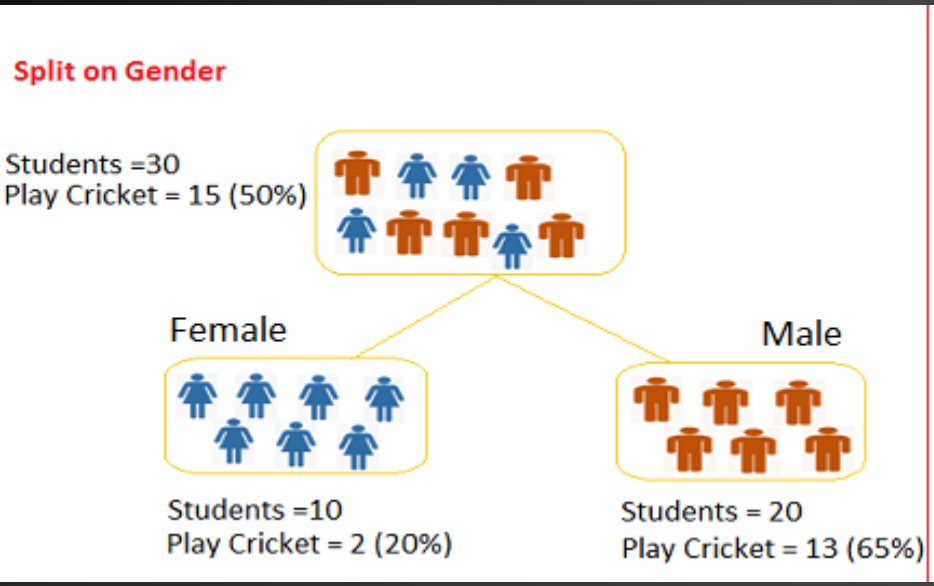
$E_C = 0.99$

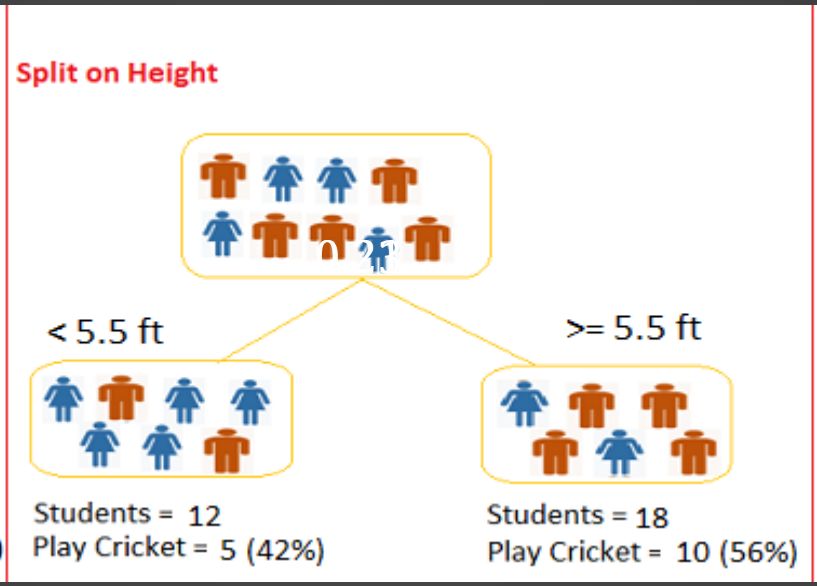
5. Supervised Learning

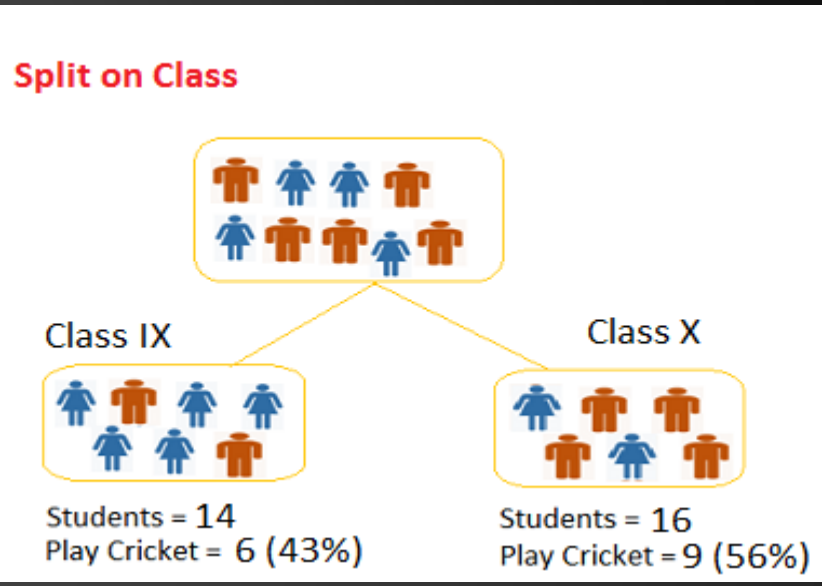
5.2. Decision Tree

Reduction-Variance example: (*Continuous*) – the smaller the Variance the better

$$Variance = \frac{\sum (X - \bar{X})^2}{n}$$







$$\bar{V_1} = \frac{(2 * 1 + 8 * 0)}{10} = 0.2$$

$$V_1 = \frac{2 * (1 - 0.2)^2 + 8 * (0 - 0.2)^2}{10} = 0.16 \qquad V_2 = 0.23$$

$$VG = \frac{10}{30} * 0.16 + \frac{20}{30} * 0.23 = 0.21$$

$$VC = \frac{14}{30} * 0.24 + \frac{16}{30} * 0.25 = 0.25$$

5. Supervised Learning

S. No.	Factors	Variables	Algorithms	Methods	Splitting Criteria	Type of Tree	Missing Value	Noisy Data	Analysis	Performance	Pruning Support
1	Entropy	Continuous Discrete Categorical	ID3 and C4.5	Exploratory Analysis (Impurity Measure)	Grouping of Classes that include up to 50% of the Data	---	---	---	---	Little slower to compute	---
2	Information Gain	Categorical , discrete or Nominal	ID3 and C4.5	Use Entropy for Analysis	Split dataset into large number of partitions	Consider Binary Split	Can't handle missing values	Can't handle noisy Data	Biased towards multivalued attributes	---	Cannot handle Over Fitting
3	Information Gain Ratio	Numerical Continuous Discrete and Nominal	C4.5	Use Information Gain and split information value or Intrinsic Information for Analysis	Prefer unbalanced splits in which one partition is much smaller than the other	Consider Multivalued Split	Can handle Missing Values	Can handle noisy Data	Reduce Biased towards multivalued attributes	---	Reduce Over Fitting by using Pruning Methods
4	Gini Index	Discrete Numerical continuous	CART	Use Weighted average of each branch index (Impurity Measure)	Grouping of largest class from other classes (Twoing Criteria)	Consider Binary Split	Can handle Missing Values	Can handle noisy Data	Biased towards multivalued attributes	Little Faster to compute	Reduce Over Fitting by using Pruning Methods

5. Supervised Learning

5.2. Decision Tree

Pros	Cons
- Work with categorical/continuous - Easy to implement & explain - Easy to display graphically - No need to clean data	- Low Predictive accuracy - High chance of overfitting

5. Supervised Learning

5.2. Decision Tree

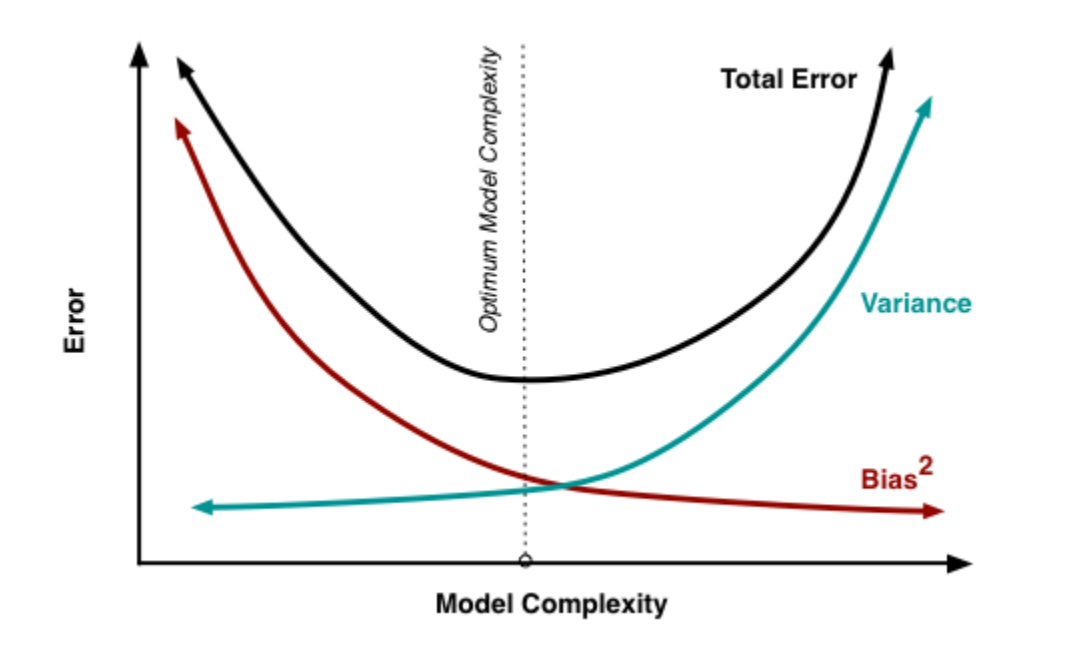
Pros	Cons
- Work with categorical/continuous - Easy to implement & explain - Easy to display graphically - No need to clean data	- Low Predictive accuracy - High chance of overfitting

By aggregating many Decision Trees using Ensemble methods, the Predictive Performance can be substantially improved

5. Supervised Learning

5.3. Ensemble

- Ensemble is to "group" many predictive model to achieve a better accuracy and model stability
- How to balance between Bias & Variance to get optimal Error?
- Trade-off management
- Solved by Ensemble method



5. Supervised Learning

5.3. Ensemble

5.3.1. Random Forest

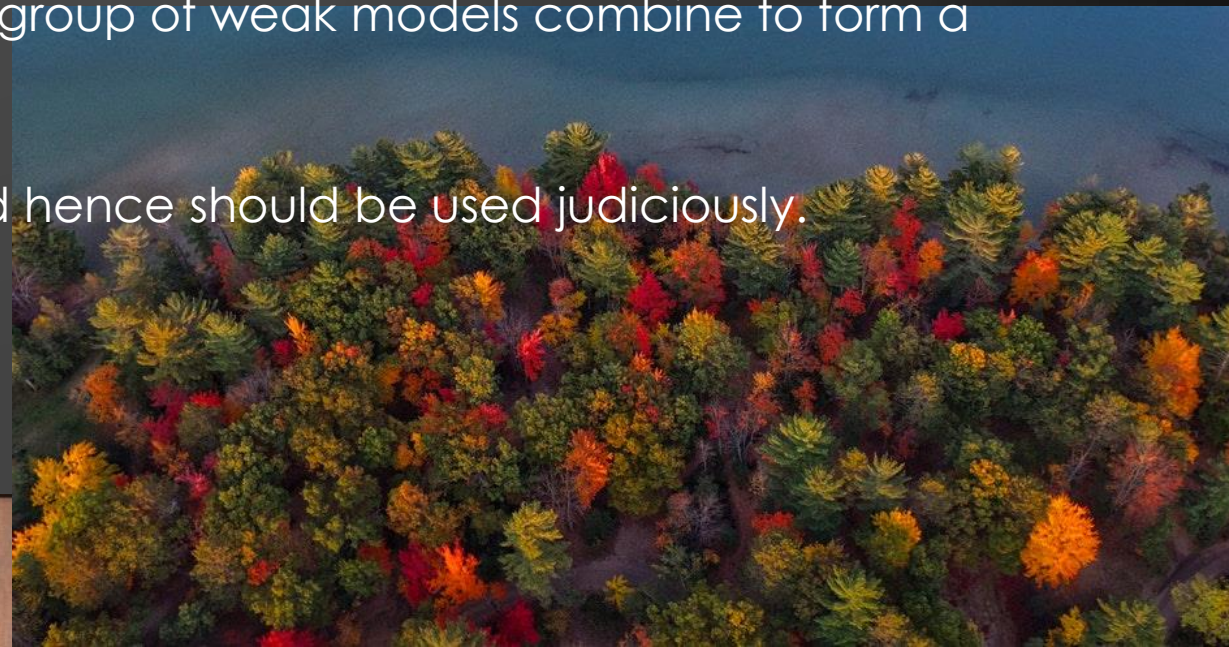
Random Forest is considered to be a **panacea** of all data science problems. On a funny note, when you can't think of any algorithm (irrespective of situation), use random forest!

Opposite to Decision Tree, Random Forest use bootstrapping technique to grow multiple tree

Random Forest is a versatile machine learning method capable of performing both regression and classification tasks.

It is a type of ensemble learning method, where a group of weak models combine to form a powerful model.

The end output of the model is like a black box and hence should be used judiciously.

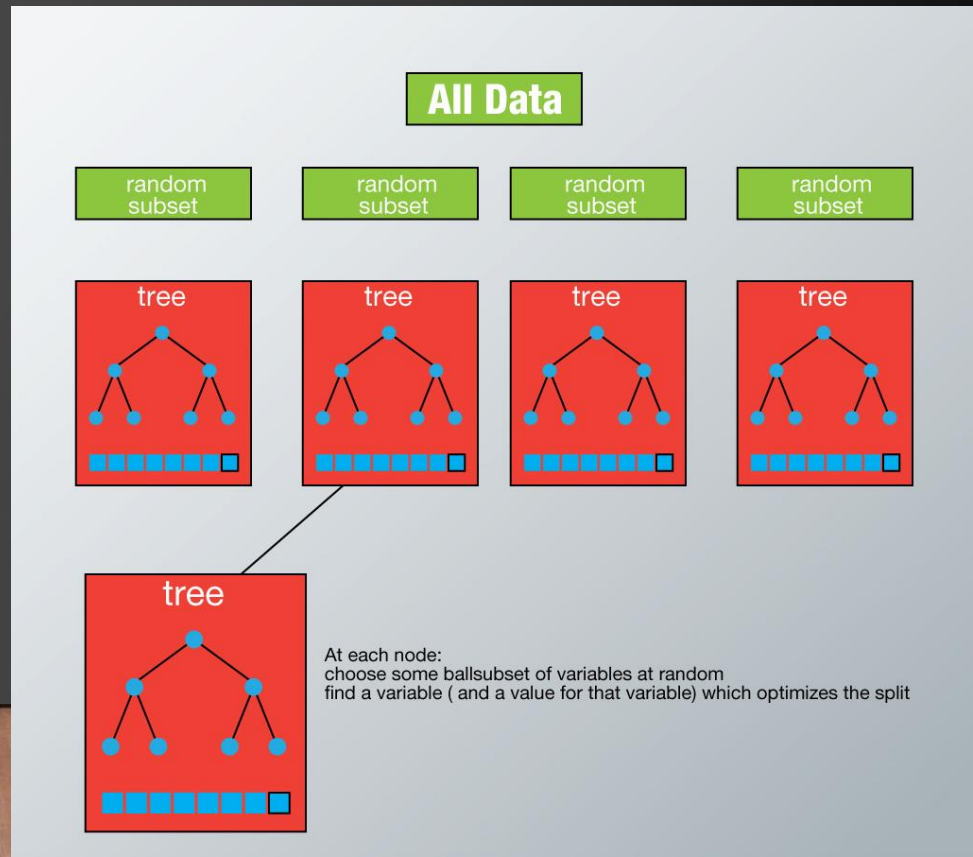


5. Supervised Learning

5.3. Ensemble

5.3.1. Random Forest

1. If there are M input variables, a number m<M is specified such that at each node, m variables are selected at random out of the M. The best split on these m is used to split the node. The value of m is held constant while we grow the forest.
2. Each tree is grown to the largest extent possible and there is no pruning.
3. Predict new data by aggregating the predictions of the ntree trees (i.e., majority votes for classification, average for regression).



5. Supervised Learning

5.3. Ensemble

5.3.1. Random Forest

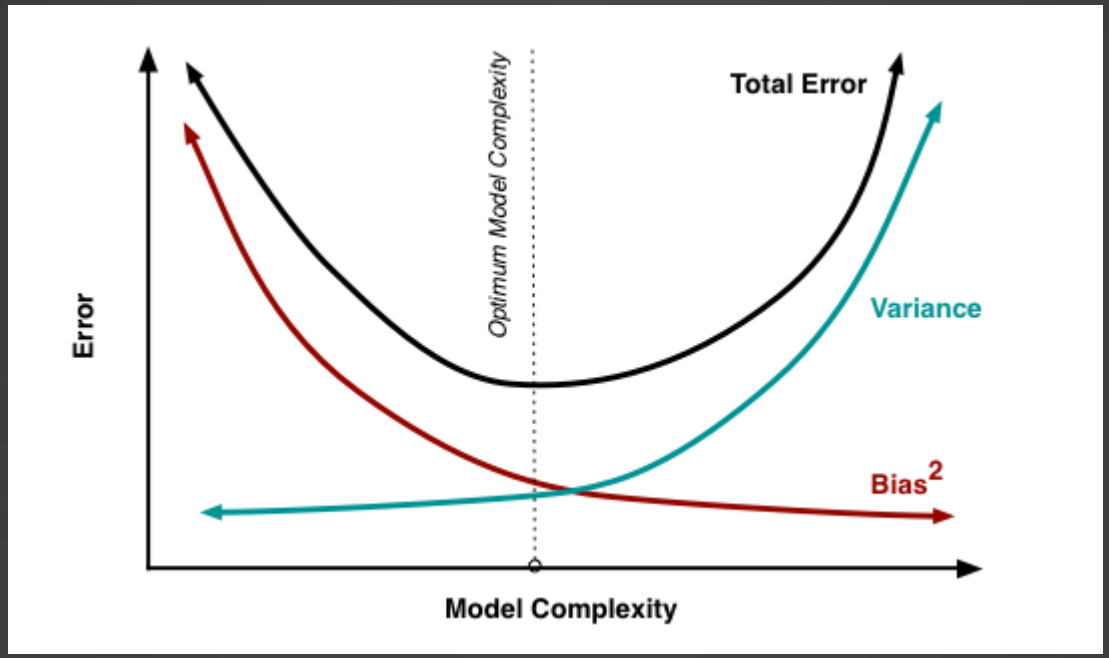
Pros	Cons
- Solve classification & regression - Power to handle large data with high dimension - Handy at missing data	- Not very good for regression (limited in predicting) - Like blackbox model and hard to control parameter - Slow processing speed

5. Supervised Learning

5.3. Ensemble

5.3.2. Bagging (Bootstrap Aggregating)

Reduce variance & Avoid Overfitting by combining results of multiple classifiers on different sub-samples



Classifier: Decision Trees, Regression models, etc.

5. Supervised Learning

5.3. Ensemble

5.3.2. Bagging (Bootstrap Aggregating)

What is Bootstrap?

The bootstrap method is a resampling technique used to estimate statistics on a population by sampling a dataset with replacement.

5. Supervised Learning

5.3. Ensemble

5.3.2. Bagging (Bootstrap Aggregating)

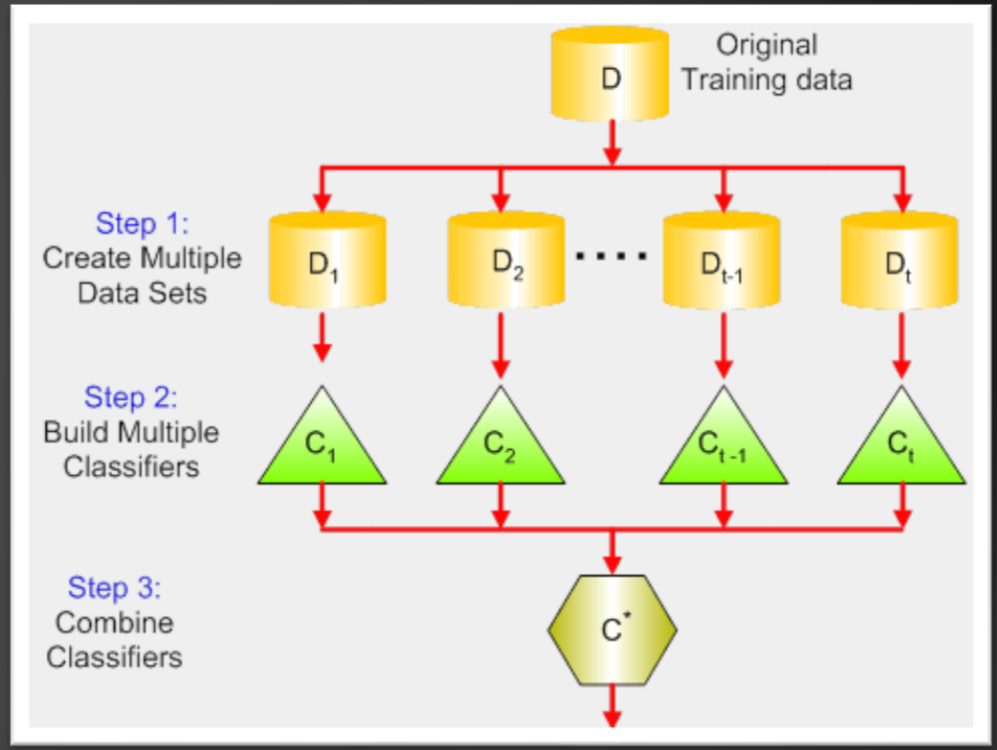
3 steps in Baggings:

Step 1: Here you replace the original data with new sub-sample data using bootstrapping.

Step 2: Train each sub-sample data using ML algorithm

Step 3: Lastly, you use an average value to combine the predictions of all the classifiers, depending on the problem. Generally, these combined values are more robust than a single model.

Bagging has been demonstrated to give impressive improvements in accuracy by combining together hundreds or even thousands of models into a single procedure.



5. Supervised Learning

5.3. Ensemble

5.3.2. Bagging (Bootstrap Aggregating)

Bagging in R can be used in many different model:

- ctreebag: used for Decision Tree
- bagFDA: used for Flexible Discriminant Analysis
- ldaBag: Bagging for Linear Discriminant Analysis
- plsBag: Bagging for Principal Linear Regression
- etc

5. Supervised Learning

5.3. Ensemble

5.3.3. Boosting

Approach to convert weak predictors to get stronger predictors.

5. Supervised Learning

5.3. Ensemble

5.3.3. Boosting

Step 1: A set of classifiers: h_1, …. h_k

Classifier h: Decision Trees, Regression models, etc.

5. Supervised Learning

5.3. Ensemble

5.3.3. Boosting

Step 2: Create a classifier that combine all classification functions $f(x) = sign\left(\sum_{i=1}^{n} \alpha_i h_i(x)\right)$

- Minimize error "ε"
- Iterative, each classifier h at a step
- (Re)calculate weight "w" based on error
- Upweight misclassification
- Select next classifier h for the next step

Weight $w_{new} = \begin{cases} \frac{1}{2}\frac{1}{1-\varepsilon} w_{old} & \text{If right} \\ \frac{1}{2}\frac{1}{\varepsilon} w_{old} & \text{If wrong} \end{cases}$

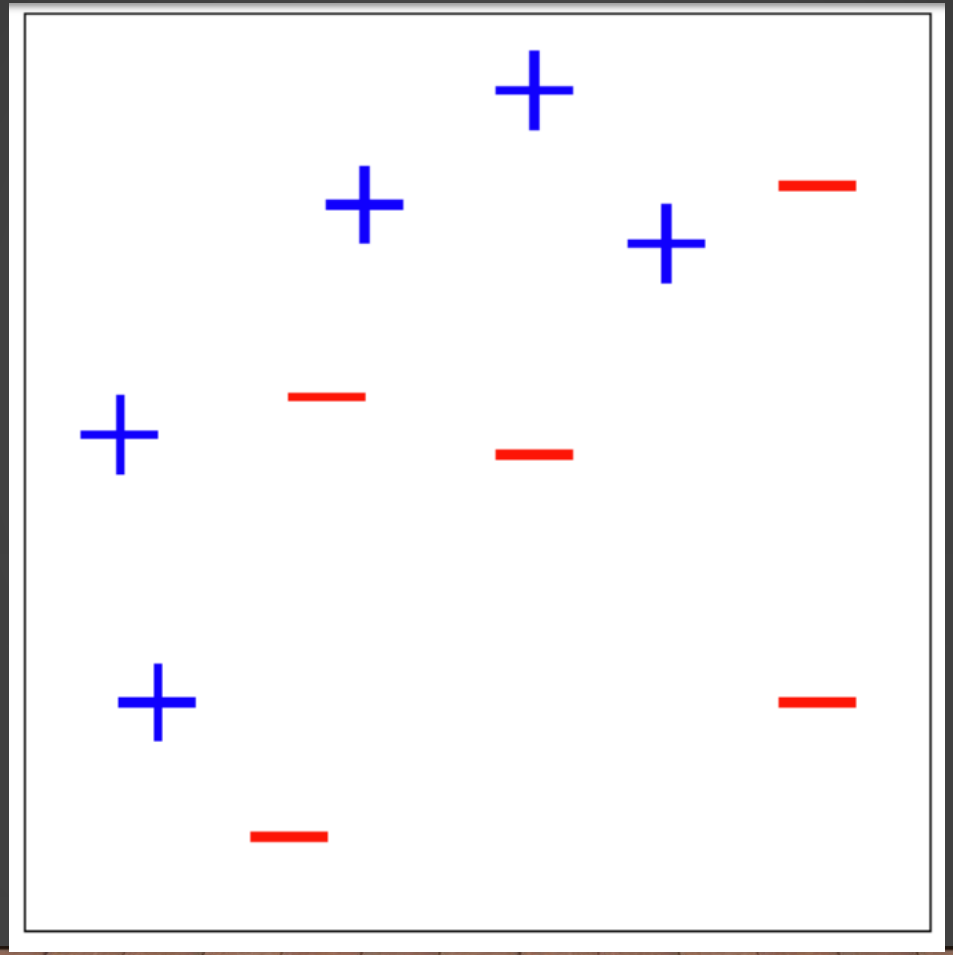
Error rate: $\varepsilon = \sum w_i$

Breiman Voting power: $\alpha = \frac{1}{2}\ln\left(\frac{1-\varepsilon}{\varepsilon}\right)$

5. Supervised Learning

5.3. Ensemble

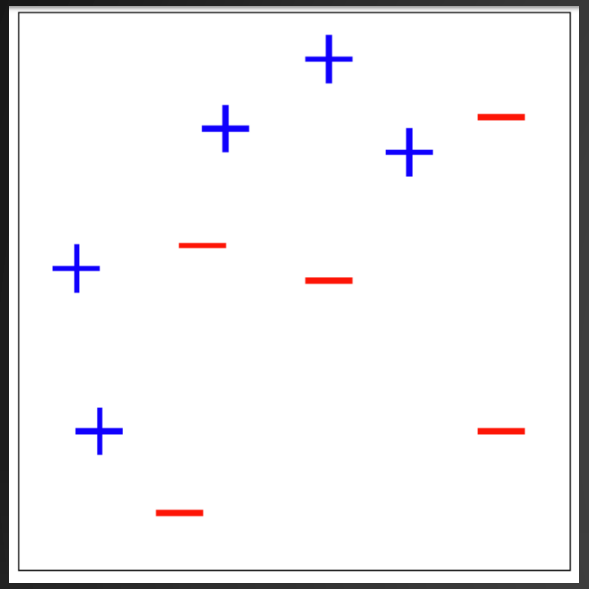
5.3.3. Boosting

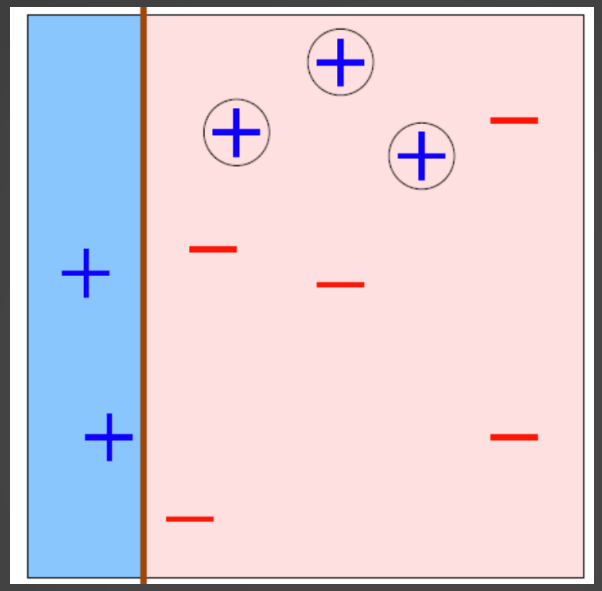


5. Supervised Learning

5.3. Ensemble

5.3.3. Boosting





- Upweight misclassification

$$w_{new} = \begin{cases} \frac{1}{2}\frac{1}{1-\varepsilon} w_{old} & \text{If right} \\ \frac{1}{2}\frac{1}{\varepsilon} w_{old} & \text{If wrong} \end{cases}$$

$w_i = 1/10$

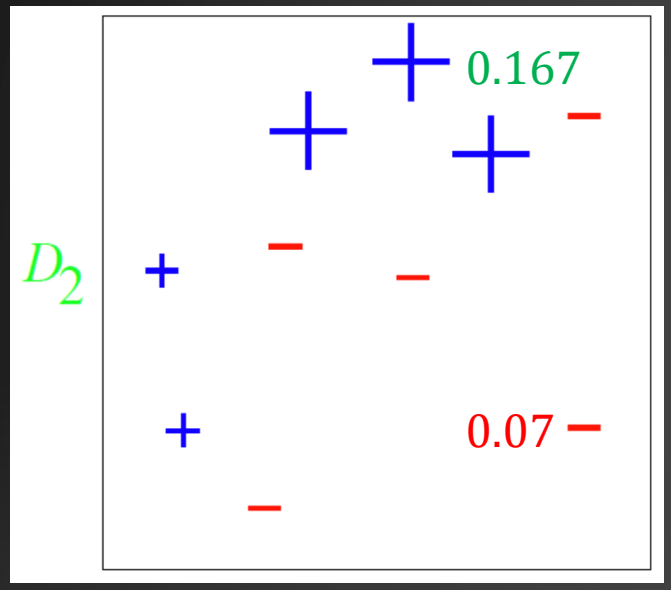
Error rate: $\varepsilon = \sum w_i$

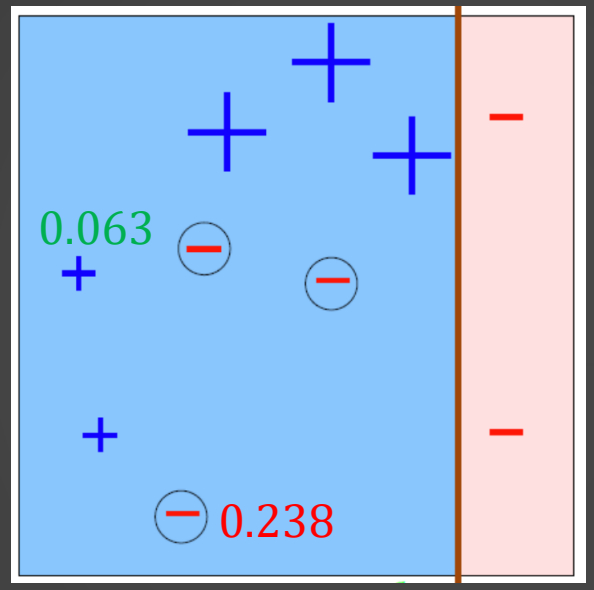
Breiman Voting power: $\alpha = \frac{1}{2}\ln\left(\frac{1-\varepsilon}{\varepsilon}\right)$

5. Supervised Learning

5.3. Ensemble

5.3.3. Boosting





- Upweight misclassification

$$w_{new} = \begin{cases} \frac{1}{2}\frac{1}{1-\varepsilon} w_{old} & \text{If right} \\ \frac{1}{2}\frac{1}{\varepsilon} w_{old} & \text{If wrong} \end{cases}$$

Error rate: $\varepsilon = \sum w_i$

$\varepsilon_2 = 0.21$

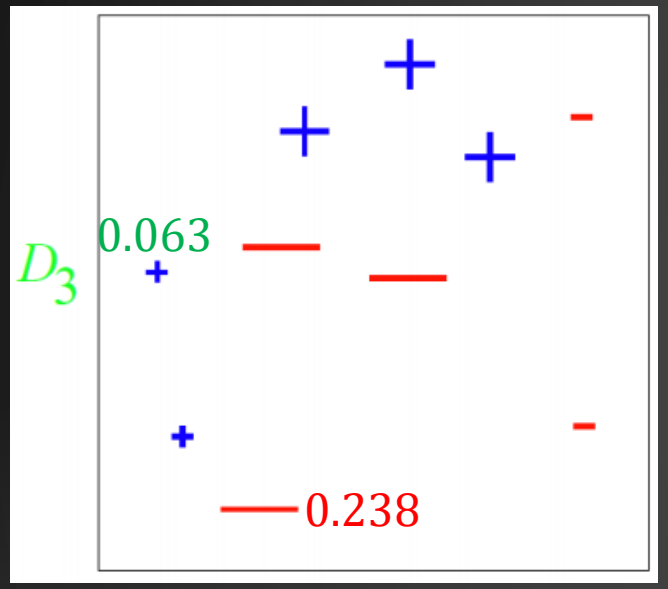
$\alpha_2 = 0.65$

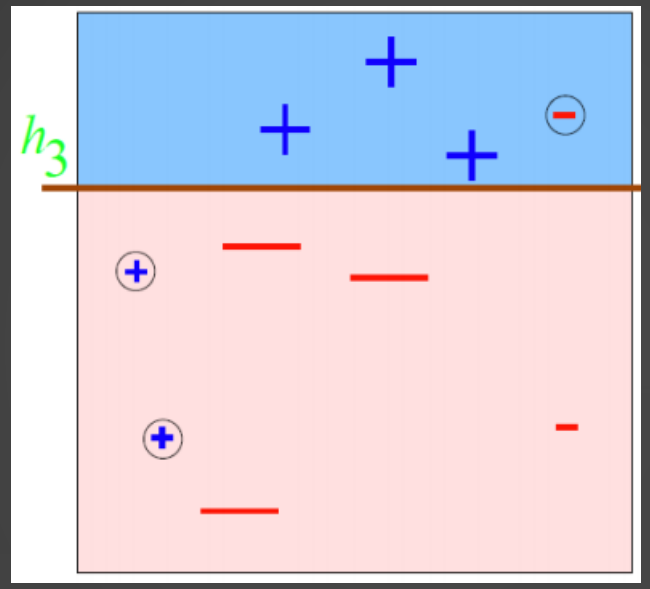
Breiman Voting power: $\alpha = \frac{1}{2}\ln\left(\frac{1-\varepsilon}{\varepsilon}\right)$

5. Supervised Learning

5.3. Ensemble

5.3.3. Boosting





- Upweight misclassification

$$w_{new} = \begin{cases} \frac{1}{2}\frac{1}{1-\varepsilon} w_{old} & \text{If right} \\ \frac{1}{2}\frac{1}{\varepsilon} w_{old} & \text{If wrong} \end{cases}$$

Error rate: $\varepsilon = \sum w_i$

$\varepsilon_3 = 0.14$

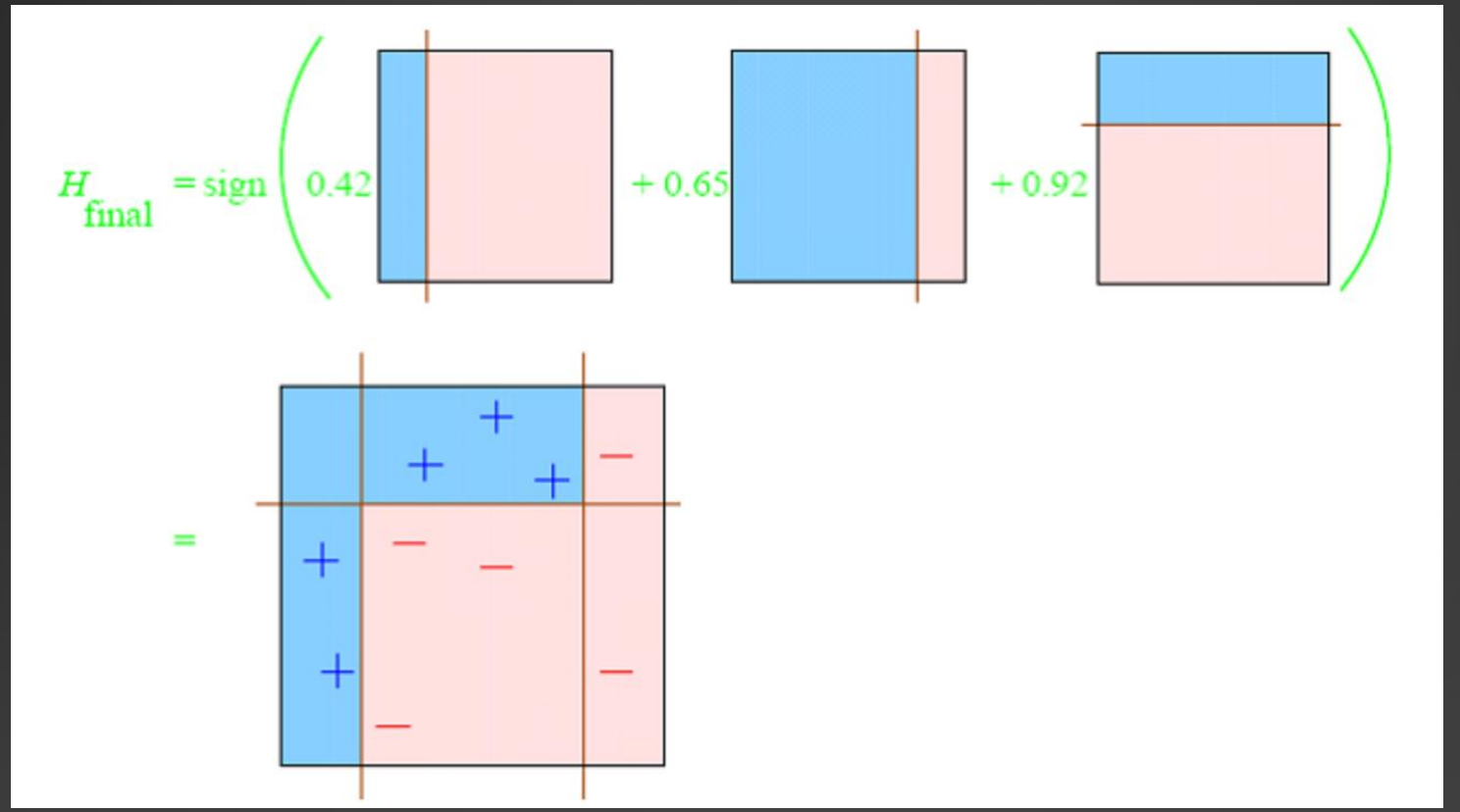
$\alpha_2 = 0.92$

Breiman Voting power: $\alpha = \frac{1}{2}\ln\left(\frac{1-\varepsilon}{\varepsilon}\right)$

5. Supervised Learning

5.3. Ensemble

5.3.3. Boosting



5. Supervised Learning

5.3. Ensemble

5.3.3. Boosting-AdaBoost (Adaptive Boosting)

Adaptive: weaker learners are tweaked by misclassify from previous classifier

AdaBoost is best used to boost the performance of decision trees on binary classification problems.

Better for classification rather than regression.

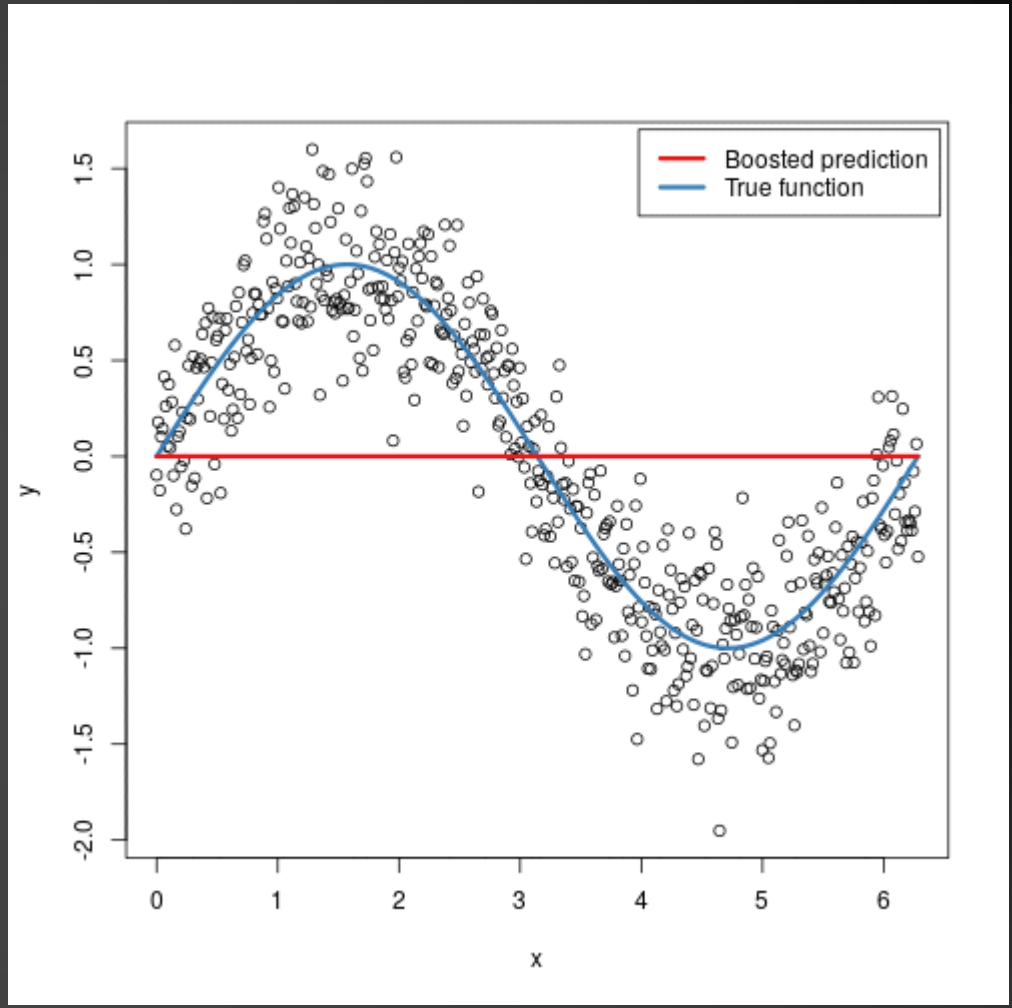
Sensitive to noise

5. Supervised Learning

5.3. Ensemble

5.3.4. Boosting-Gradient Boosting Machines

- Extremely popular ML algorithm
- Widely used in Kaggle competition
- Ensemble of shallow and weak successive tree, with each tree learning and improving on the previous



5. Supervised Learning

5.3. Ensemble

5.3.4. Boosting-Gradient Boosting Machines

- Extremely popular ML algorithm
- Widely used in Kaggle competition
- Ensemble of shallow and weak successive tree, with each tree learning and improving on the previous

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

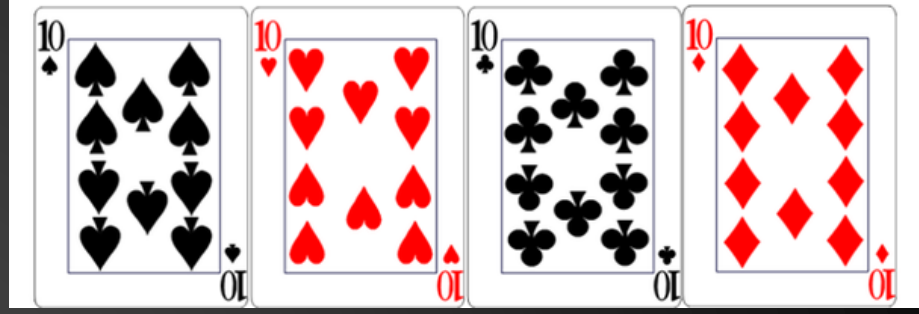
- Assuming data follow a probabilistic model
- Assuming all predictors are independent (Naïve assumption)
- Use Bayes's theorem to identify optimal classifiers

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

Bayes Theorem



Marginal Probability: P(A); P(B)

Joint Probability: P(A,B) or P(A∩B)

Conditional Probability: P(A | B); P(B | A)

Bayes Theorem

Marginal Probability: P(A); P(B)

Joint Probability: P(A,B) or P(A∩B)

Conditional Probability: P(A | B); P(B | A)

$$P(A \cap B) = P(A)P(B \mid A) = P(B)P(A \mid B)$$

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

Bayes Theorem

Bayes Theorem: $$P(A|B) = \frac{P(A)P(B|A)}{P(B)}$$

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

Bayes Theorem

Example: Picnic Day:

You are planning a picnic today, but the morning is cloudy

•Oh no! 50% of all rainy days start off cloudy!

•But cloudy mornings are common (about 40% of days start cloudy)

•And this is usually a dry month (only 3 of 30 days tend to be rainy, or 10%)

=> What is the chance of rain during the day?

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

Bayes Theorem

Example: Picnic Day:

You are planning a picnic today, but the morning is cloudy

- Oh no! 50% of all rainy days start off cloudy! P(Cloud | Rain) = 0.5
- But cloudy mornings are common (about 40% of days start cloudy): P(Cloud)=0.4
- And this is usually a dry month (only 3 of 30 days tend to be rainy, or 10%): P(Rain)=0.1

=> What is the chance of rain during the day? P(Rain|Cloud)

$$P(Rain|Cloud) = \frac{P(Rain)\,P(Cloud|Rain)}{\mathrm{P(Cloud)}} = \frac{0.1 * 0.5}{0.4} = 0.125 = 12.5\%$$

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

Bayes Theorem

Example: Allergy or Not?

Hunter says she is itchy. There is a test for Allergy to Cats, but this test is not always right:

- For people that really do have the allergy, the test says "Yes" 80% of the time
- For people that do not have the allergy, the test says "Yes" 10% of the time ("false positive")

If 1% of the population have the allergy, and Hunter's test says "Yes", what are the chances that Hunter really has the allergy?

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

Bayes Theorem

Example: Allergy or Not?

Hunter says she is itchy. There is a test for Allergy to Cats, but this test is not always right:

- For people that really do have the allergy, the test says "Yes" 80% of the time P(Yes | Allergy)=0.8
- For people that do not have the allergy, the test says "Yes" 10% of the time ("false positive") P(Yes | NoAllergy)=0.1
- 1% of the population have the allergy: P(Allergy)=0.01
- 99% of the population have NO allergy: P(NoAllergy)=0.99

Hunter's test says "Yes", what are the chances that Hunter really has the allergy?: P(Allergy | Yes)

$$P(Allergy|Yes) = \frac{P(Allergy)\ P(\mathrm{Yes}|Allergy)}{\mathrm{P}(Yes)}$$

$$\mathrm{P}(Yes) = \mathrm{P}(\mathrm{Yes}|\mathrm{Allergy})*\mathrm{P}(\mathrm{Allergy}) + \mathrm{P}(\mathrm{Yes}|\mathrm{NoAllergy})*\mathrm{P}(\mathrm{NoAllergy})$$

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

Bayes Theorem

$$P(Allergy|Yes) = \frac{P(Allergy)\,P(\text{Yes}|Allergy)}{\text{P(Allergy)}*\text{P(Yes|Allergy)}+\text{ P(NoAllergy)}*\text{P(Yes|NoAllergy)}}$$

$$= \frac{0.01*0.8}{0.01*0.8+0.99*0.1} = 0.0747 \sim 7\%$$

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

Bayes Theorem

Special Bayes Theorem: $$P(A|B) = \frac{P(A)P(B|A)}{P(A)P(B|A) + P(notA)P(B|notA)}$$

Suppose we have many predictors A_i:

$$P(A_1|B) = \frac{P(A_1)P(B|A_1)}{P(A_1)P(B|A_1) + P(A_2)P(B|A_2) + P(A_3)P(B|A_3)}$$

$P(A_1)+P(A_2)+P(A_3)=1$

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

- A classification technique based on Bayes Theorem
- Assume Independence among predictors: A_1, A_2, $A_{3,}$ ... A_n
- Outcome is Independence variable Class B

Predictand Prior Probability Likelihood

Posterior Prob $$P(B|A) = \frac{P(B)P(A|B)}{P(A)}$$

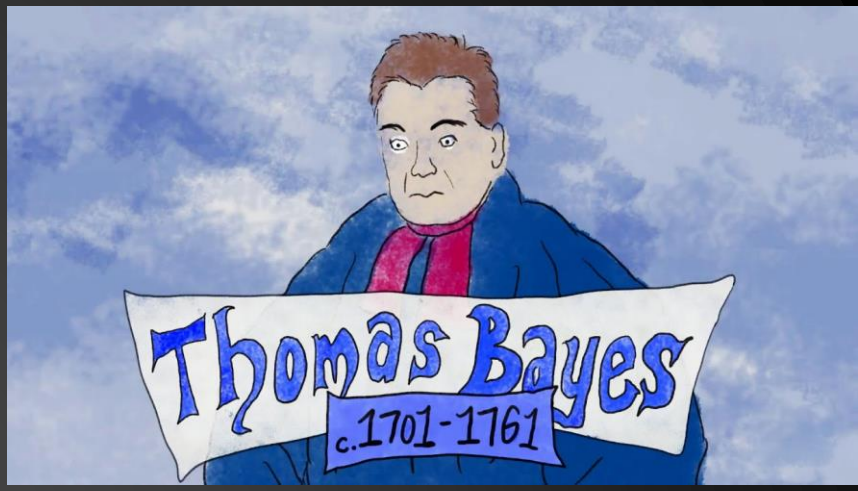
Predictor Prior Prob

5. Supervised Learning

5.4. Model Based Prediction

5.4.1. Naïve Bayes

- Assuming data follow a probabilistic model
- Assuming all predictors are independent (Naïve assumption)
- Use Bayes's theorem to identify optimal classifiers



Pros	Cons
- Easy to predict - With independent variables, Naive Bayes performs better compared to other - Computationally convenient - Perform well in categorical compared to numerical variables	- Have to make assumption about independence predictors - If variable has not observed in training data sets then model will assign 0 probability and unable to make decision

5. Supervised Learning

5.4. Model Based Prediction

5.4.2. Linear Discriminant Analysis

- **LDA** is a *supervised learning* model that is similar to *logistic regression* in that the outcome variable is categorical and can therefore be used for *classification*.
- LDA is useful with two or more class of objects

5. Supervised Learning

5.4. Model Based Prediction

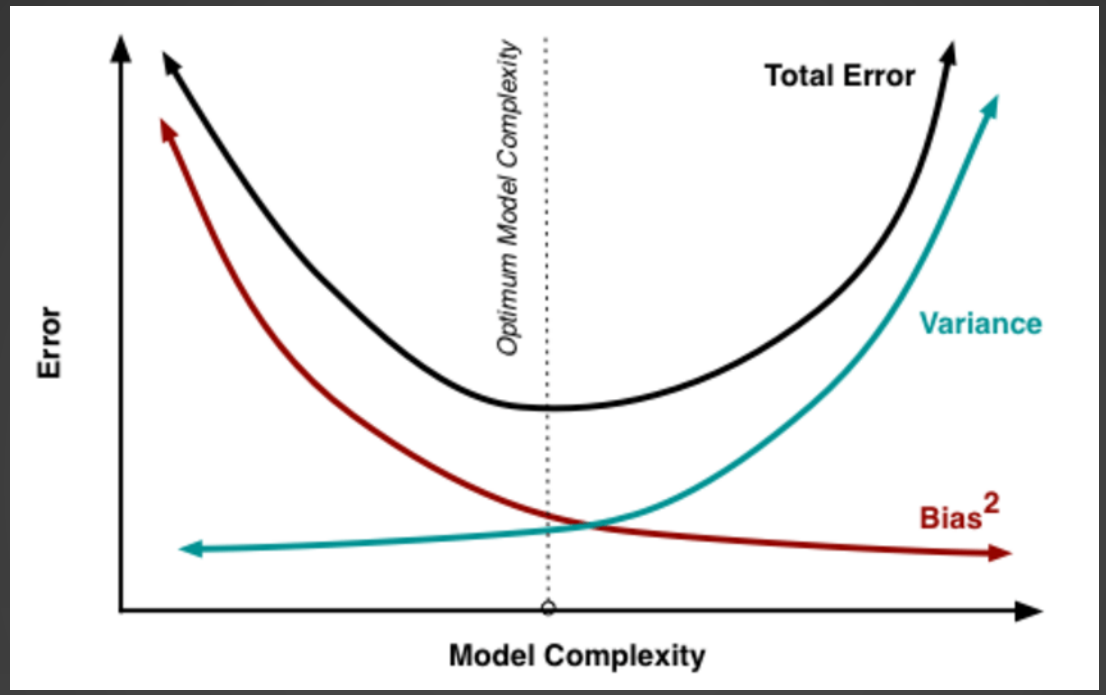
5.4.2. Linear Discriminant Analysis

Logistic Regression	Linear Discriminant Analysis
- Supervised Learning with Categorical outcome - Deal with only binary outcome - Use logistic function to model probability	- Supervised Learning with Categorical outcome - Deal with more than 2 outcomes - Used Bayes Theorem to model probability

Naïve Bayes	Linear Discriminant Analysis
- Categorical outcome - Assume Independence input (cross-correlation is ~ 0)	- Categorical outcome - Assume Gaussian Distribution for input - Sensitive to Overfitting

5. Supervised Learning

5.5. Regularization and Variable selection



In order to reduce the Model Complexity or to avoid Multi-Collinearity, one needs to reduce the number of covariates (or set the coefficient to be zero). If the coefficients are too large, let's penalize them to enforce them to be smaller

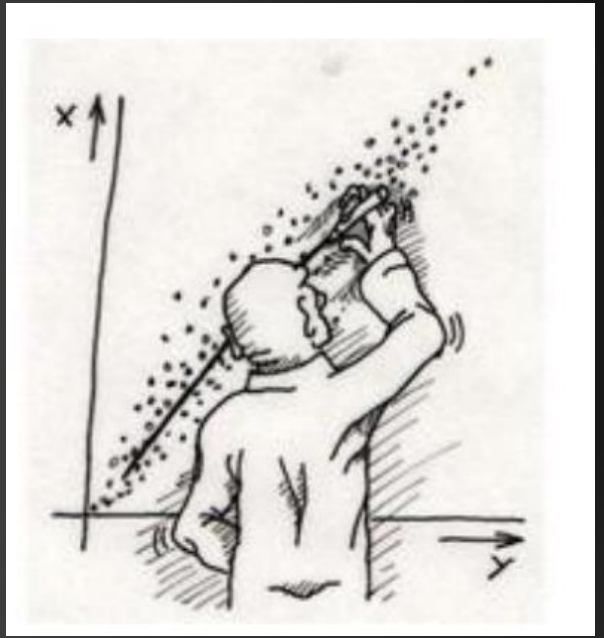
5. Supervised Learning

5.5. Regularization and Variable selection

5.4.1. Regularization

Recap about Linear Regression:

- Observation (output): y
- Independent variable (input): X
- Linear Regression model: $y = \beta_0 + \beta x + \varepsilon$
- Multi-Linear Regression: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \ldots . + \beta_n x_n + \varepsilon$



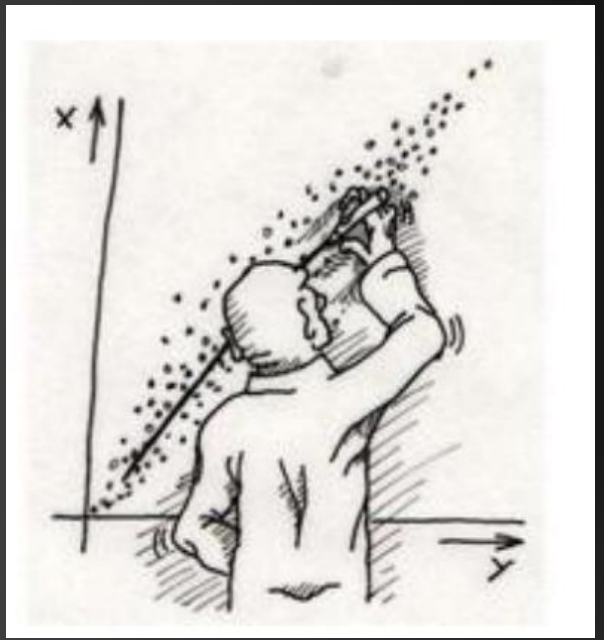
5. Supervised Learning

5.5. Regularization and Variable selection

5.4.1. Regularization

To determine the optimal β, we resolve the least square problem using the "Lost Function":

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - \beta x_i)^2 = \underset{\beta}{\operatorname{argmin}} \|y - \beta x\|_2^2$$



Need to add some penalty term

5. Supervised Learning

5.5. Regularization and Variable selection

5.4.1. Regularization

Ridge Regression Lost Function:

$$\hat{\beta}_{ridge} = \sum_{i=1}^{n} (y_i - \beta_i x_i)^2 + \lambda \sum_{j=1}^{m} \beta_j^2$$

- Shrink the coefficient that contribute most error
- Reduce magnitude of coef that contribute the most to increasing L

λ: Regularization Penalty, to be selected that the model minimized the error
β: vector with many components as predictors

- As $\lambda \to 0$, $\hat{\beta}_{ridge} \to \hat{\beta}_{OLS}$;
- As $\lambda \to \infty$, $\hat{\beta}_{ridge} \to 0$.

5. Supervised Learning

5.5. Regularization and Variable selection

5.4.1. Regularization

LASSO
Lost Function

$$\sum_{i=1}^{n}(y_i - \beta_i x_i)^2 + \lambda \sum_{j=1}^{m} |\beta_j|$$

- Set these coefficient to 0
- Retain only most important coef

λ: Regularization Penalty, to be selected that the model minimized the error
β: vector with many components as predictors

5. Supervised Learning

5.5. Regularization and Variable selection

5.4.1. Regularization

Package GLMNET that minimizes the Lost Function:

$$\sum_{i=1}^{n}(y_i - \beta_i x_i)^2 + \lambda\left[(1-\alpha)\sum\beta_j^2 + \alpha\sum_{j=1}^{m}|\beta_j|\right]$$

$\alpha = 0$: pure Ridge Regression
$\alpha = 1$: pure LASSO
$0 < \alpha < 1$: Elastic Nets

5. Supervised Learning

5.5. Regularization and Variable selection

5.4.1. Regularization

Lcavol: log(cancer volume)
Lweight: log(prostate weight)
Age: age
Lbph: log(benign prostatic hyperplasia amount)
Svi: seminal vesicle invasion
Lcp: log(capsular penetration)
Gleason: Gleason score
Pgg45: percentage Gleason scores 4 or 5
Lpsa: log(prostate specific antigen)

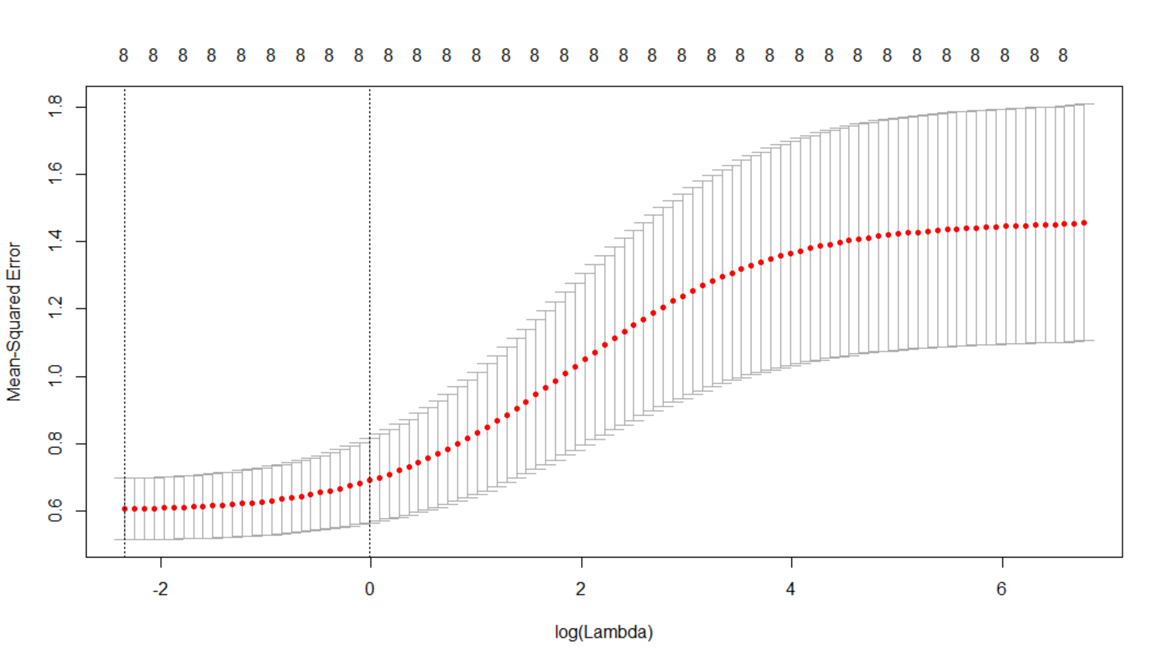
5. Supervised Learning

5.5. Regularization and Variable selection

5.4.2. Ridge Regression

```
library(glmnet)
y <- training$lpsa
x <- training[,-c(9,10)]
x <- as.matrix(x)
Ridge    <- cv.glmnet(x,y,alpha=0)
plot(Ridge)
```

```
> cvfit_Ridge$lambda.min
[1] 0.09645702
> cvfit_Ridge$lambda.1se
[1] 1.189167
```



5. Supervised Learning

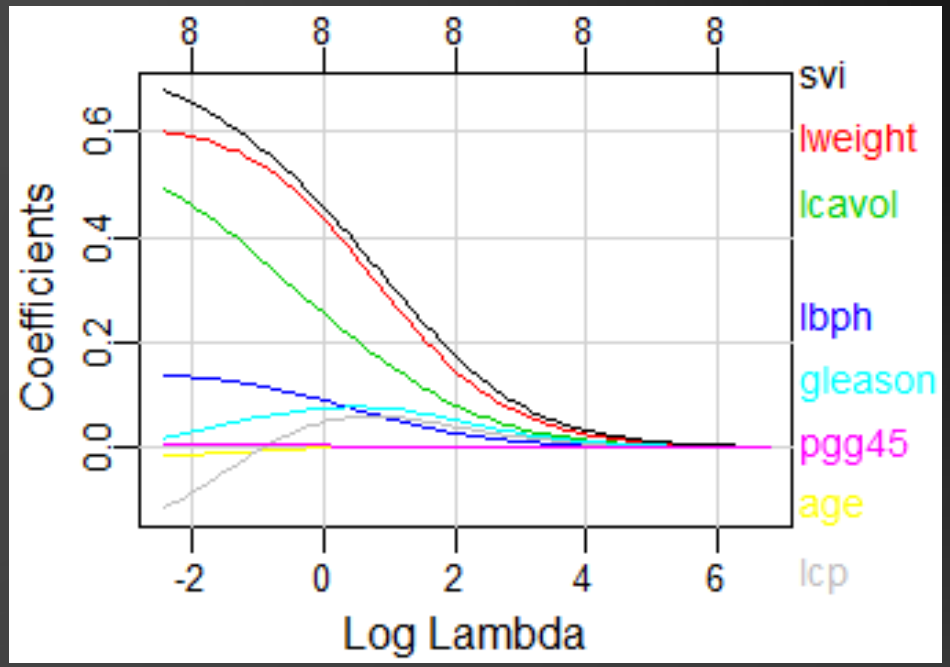
5.5. Regularization and Variable selection

5.4.2. Ridge Regression

```
Fit_Ridge <- glmnet(x,y,alpha=0,standardize = TRUE)
plot_glmnet(Fit_Ridge,label=TRUE,xvar="lambda",
            col=seq(1,8),,grid.col = 'lightgray')
```

```
> coef(cvfit_Ridge,s=cvfit_Ridge$lambda.min)

9 x 1 sparse Matrix of class "dgCMatrix"
                      1
(Intercept)  0.075575516
lcavol       0.486389709
lweight      0.599589889
age         -0.014473630
lbph         0.137304129
svi          0.674693010
lcp         -0.110437780
gleason      0.019949363
pgg45        0.006929742
```

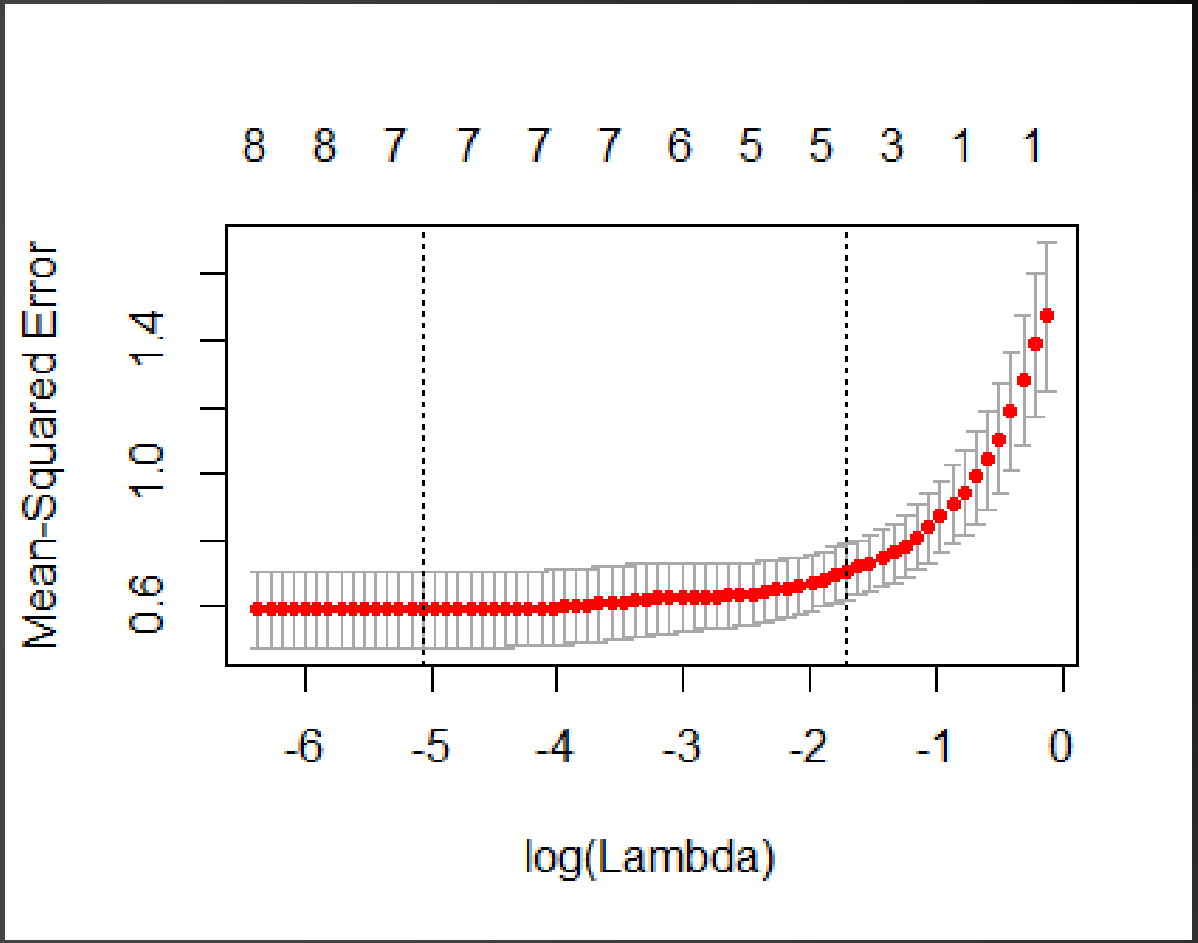


5. Supervised Learning

5.5. Regularization and Variable selection

5.4.3. LASSO

```
cvfit_LASSO c- cv.glmnet(x,y,alpha=1)
plot(cvfit_LASSO)
```

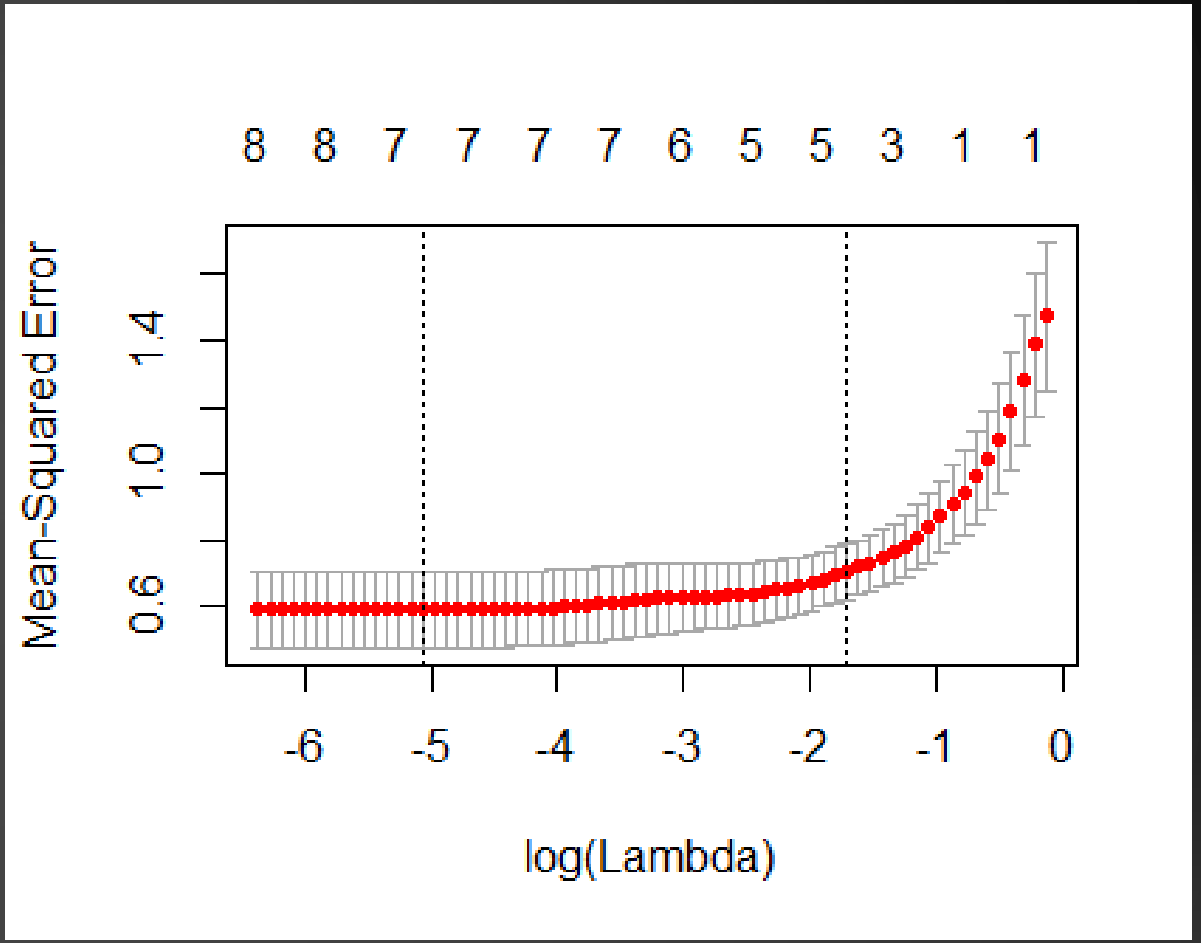


5. Supervised Learning

5.5. Regularization and Variable selection

5.4.3. LASSO

```
> cvfit_LASSO$lambda.min
[1] 0.006346228
> cvfit_LASSO$lambda.1se
[1] 0.1807428
```

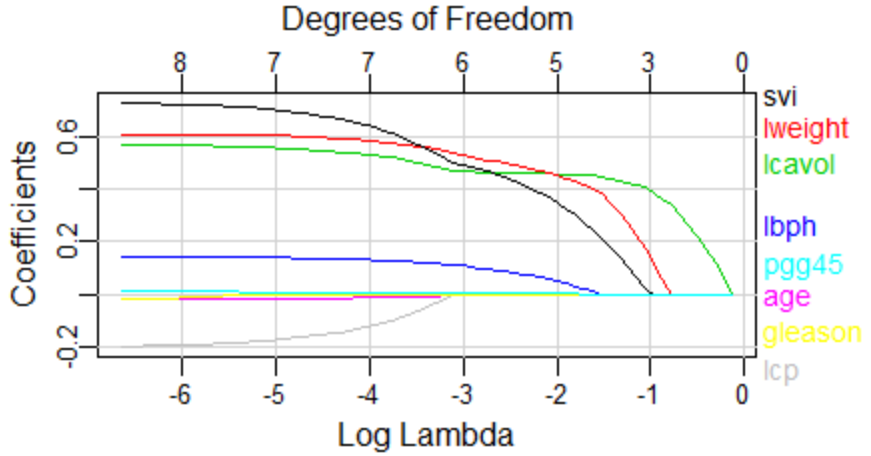


5. Supervised Learning

5.5. Regularization and Variable selection

5.4.3. LASSO

```
> coef(cvfit_LASSO,s=cvfit_LASSO$lambda.1se)
9 x 1 sparse Matrix of class "dgCMatrix"
                            1
(Intercept) 0.2602328553
lcavol      0.4549883478
lweight     0.4183048097
age         .
lbph        0.0199337240
svi         0.2745819504
lcp         .
gleason     .
pgg45       0.0005623797
```

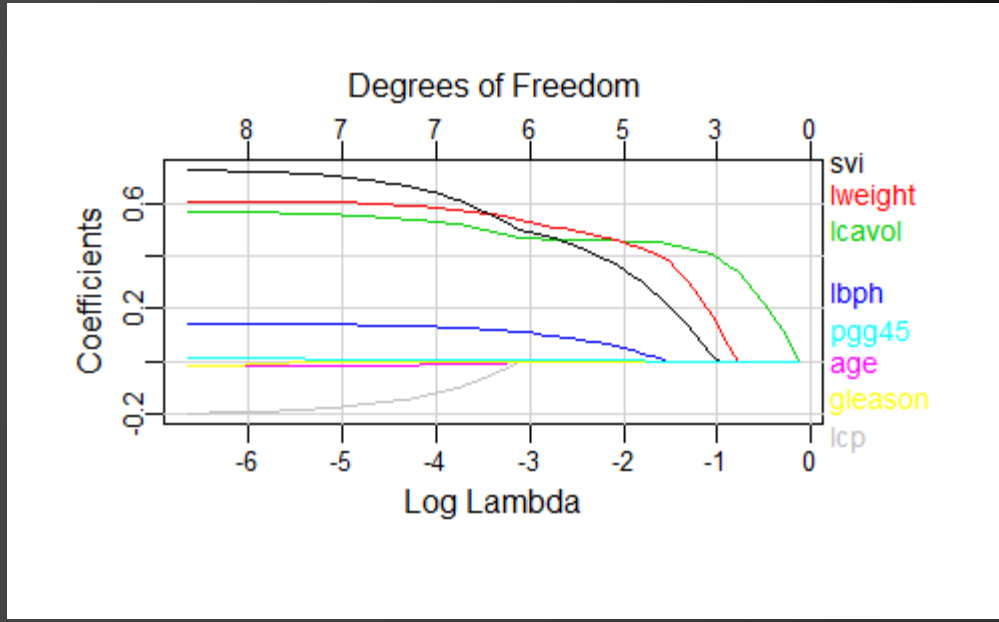


5. Supervised Learning

5.5. Regularization and Variable selection

5.4.3. LASSO

```
Fit_LASSO <- glmnet(x,y,alpha=1)
plot_glmnet(Fit_LASSO,label=TRUE,xvar="lambda",
            col=seq(1,8),,grid.col = 'lightgray')
```



5. Supervised Learning

5.5. Regularization and Variable selection

5.4.4. ELASTICNET?

5. Supervised Learning

5.6. Dimension Reduction

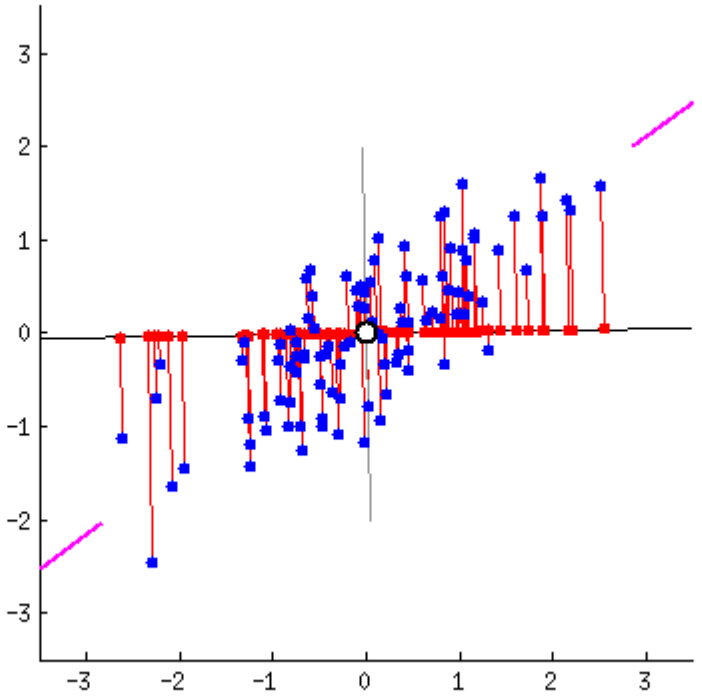
- When there are lots of covariates
- Computing resources limited
- Reduce covariates based on coherence
- Reduce noise

5. Supervised Learning

5.6. Dimension Reduction

Principal Component Analysis

- Handy with large data
- Where many variables correlate with one another, they will all contribute strongly to the same principal component
- Each principal component sums up a certain percentage of the total variation in the dataset
- More PCs, more summarization of the original data sets



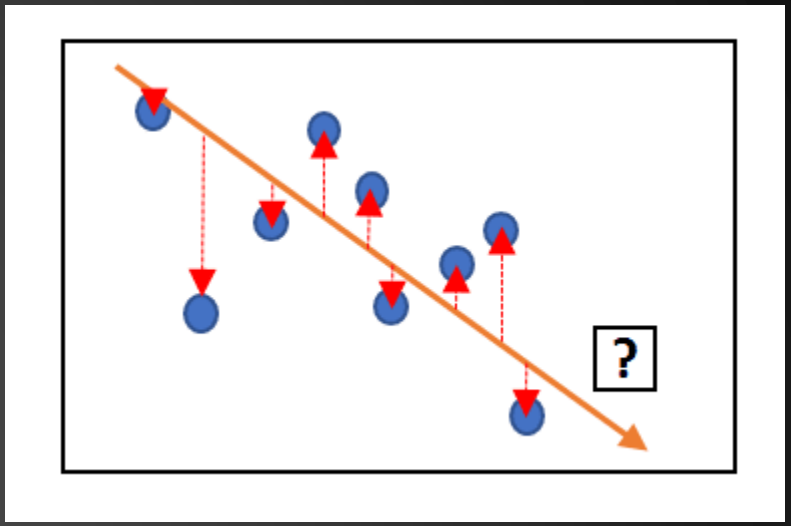
5. Supervised Learning

5.6. Dimension Reduction

Principal Component Analysis

Variance: Measure of the spread of data with its mean

$$\sigma^2 = \mathrm{var}(\mathrm{X}) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n-1}$$



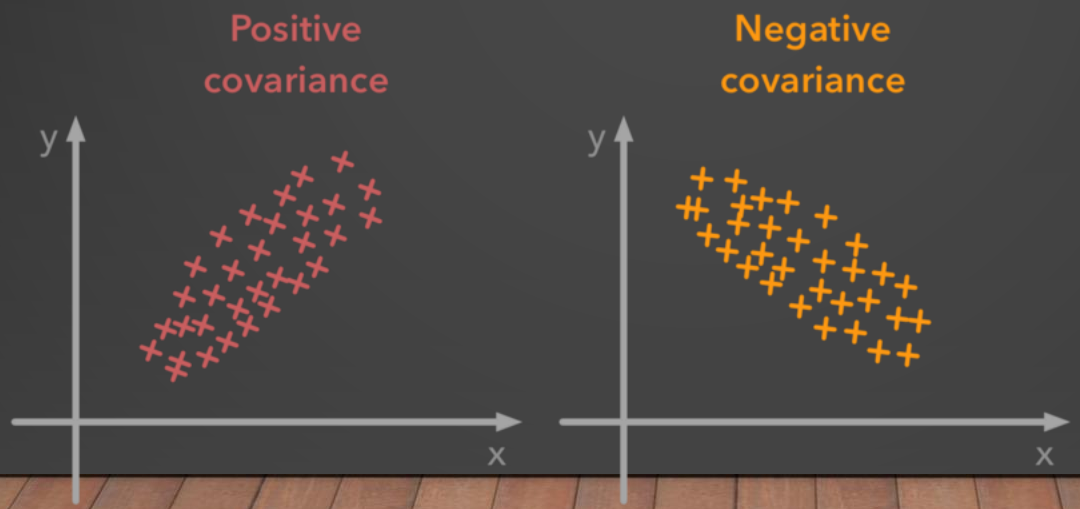
5. Supervised Learning

5.6. Dimension Reduction

Principal Component Analysis

Covariance: Measure how much each of the dimensions varies from the mean with respect to each other

$$\mathrm{cov}(X,Y) = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{n-1}$$



5. Supervised Learning

5.6. Dimension Reduction

Principal Component Analysis

Covariance Matrix: A square matrix for 3 or more dimension data

$$M = \begin{bmatrix} Cov(X,X) & Cov(X,Y) & Cov(X,Z) \\ Cov(Y,X) & Cov(Y,Y) & Cov(Y,Z) \\ Cov(Z,X) & Cov(Z,Y) & Cov(Z,Z) \end{bmatrix}$$

- Diagonal: variances of variables
- Cov(X,Y) = Cov(Y,X)
- m dimensional data results in mxm matrix

5. Supervised Learning

5.6. Dimension Reduction

Principal Component Analysis

Eigenvalue problem: $M.v = \lambda.v$

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 12 \\ 8 \end{bmatrix} = 4 \times \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

M: m x m matrix
v: m x 1 non-zero vector
λ: scalar

λ = 4: eigenvalues
v = $\begin{bmatrix} 3 \\ 2 \end{bmatrix}$: eigenvectors

Eigenvector: direction
Eigenvalue: how much variance there is in the eigenvector

5. Supervised Learning

5.6. Dimension Reduction

Principal Component Analysis

Some characteristics of eigenvalues/eigenvectors

- Given mxm matrix, we can find m eigenvectors and m eigenvalues
- Eigenvectors can only be found for square matrix.
- Not every square matrix has eigenvectors
- A square matrix **A** and its transpose have the same eigenvalues but different eigenvectors
- The eigenvalues of a diagonal or triangular matrix are its diagonal elements.
- Eigenvectors of a matrix **A** with distinct eigenvalues are linearly independent.

5. Supervised Learning

5.6. Dimension Reduction

Principal Component Analysis

Covariance Matrix: A square matrix for 3 or more dimension data

$$M = \begin{bmatrix} Cov(X,X) & Cov(X,Y) & Cov(X,Z) \\ Cov(Y,X) & Cov(Y,Y) & Cov(Y,Z) \\ Cov(Z,X) & Cov(Z,Y) & Cov(Z,Z) \end{bmatrix}$$

Find m eigenvectors and m eigenvalues of the Matrix M

5. Supervised Learning

5.6. Dimension Reduction

Principal Component Analysis

Eigenvector with the largest eigenvalue forms the first principal component of the data set

… and so on …