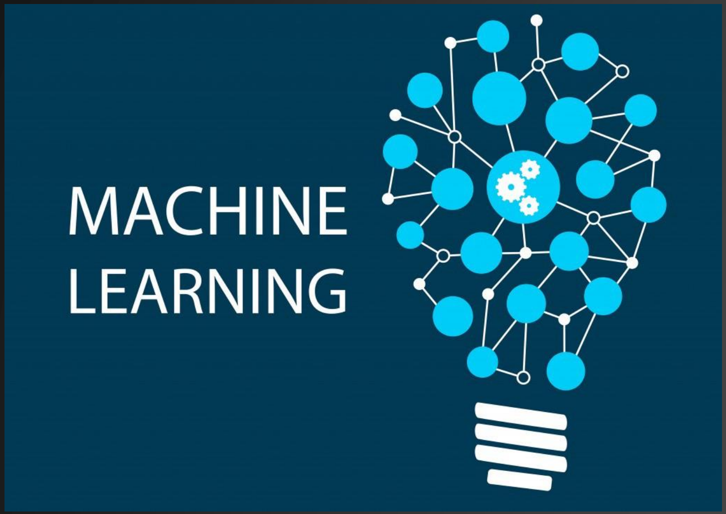
INTRODUCTION TO



FOR

USING R

TUE VU

ADVANCED COMPUTING & DATA SCIENCE (ACDS)

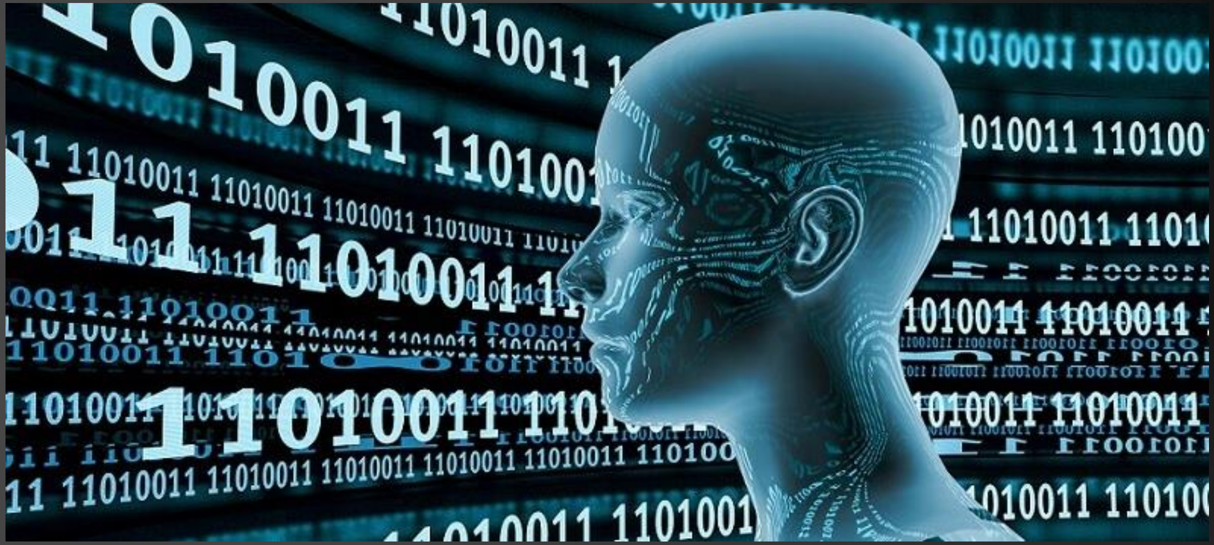
CCIT\CITI

CITI
Cyberinfrastructure Technology Integration

CLEMSON
COMPUTING AND
INFORMATION TECHNOLOGY

OUTLINES

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



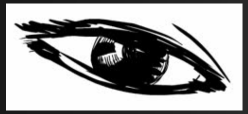
5. Supervised Learning

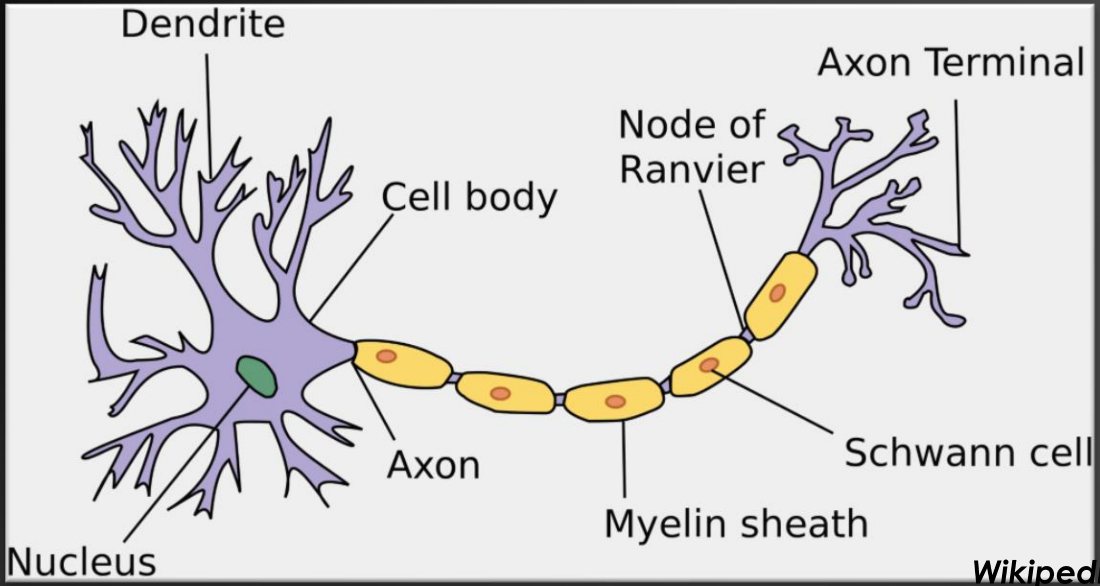
1. Regression
   - Linear Regression
   - Multi-Linear Regression (MLR)
   - Other typical Linear Regression Technique
   - Logistic Regression
2. Decision Tree
3. Ensemble Prediction
   - Random Forest
   - Bagging
   - Boosting
4. Model based Prediction
   - Naïve Bayes
   - Linear Discriminant Analysis
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 Neighbour

5. Supervised Learning

5.7. Artificial Neural Network

Biological Neuron





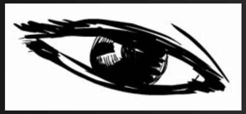
Wikipedia

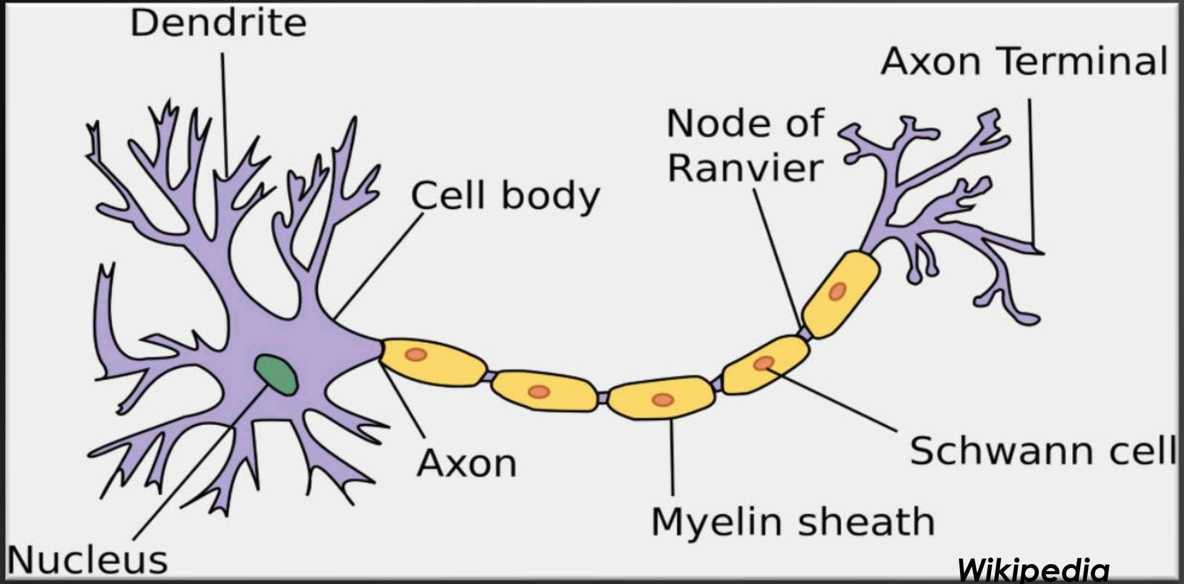
5. Supervised Learning

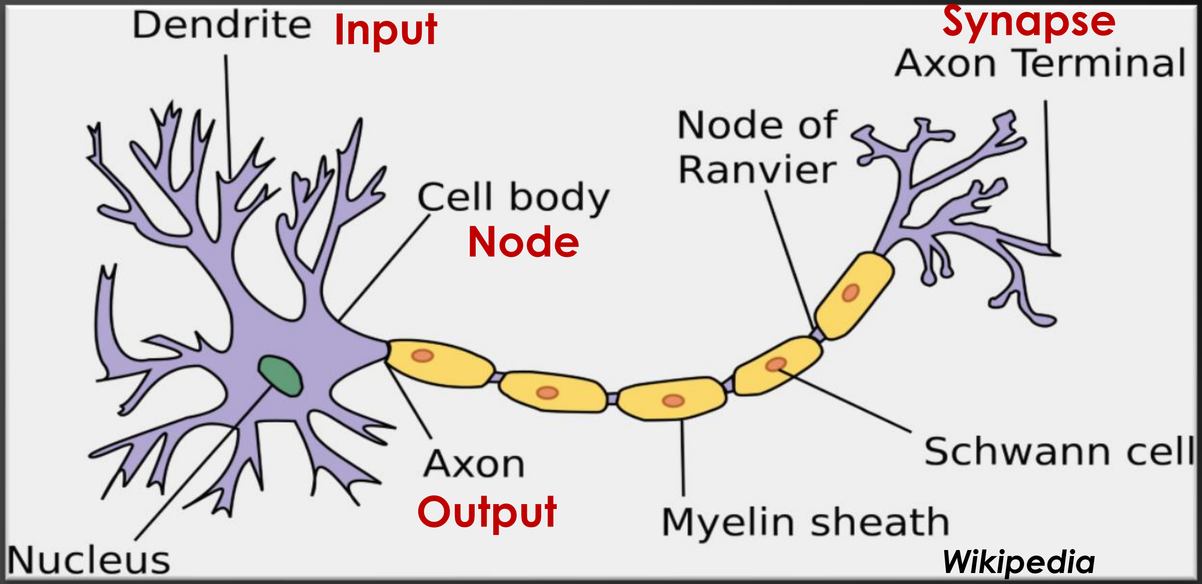
5.7. Artificial Neural Network

Biological Neuron

Artificial Neuron
(Perceptron)







5. Supervised Learning

5.7. Artificial Neural Network

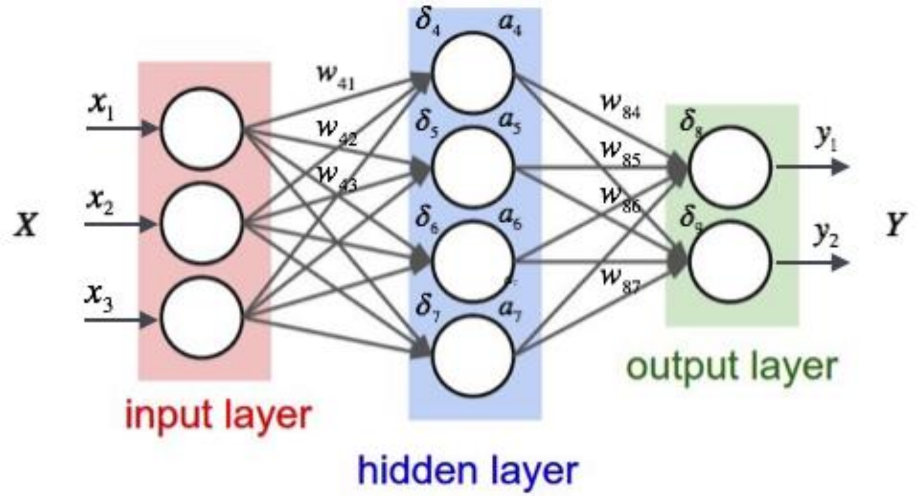
Biological Neural Network

5. Supervised Learning

5.7. Artificial Neural Network

Biological Neural Network

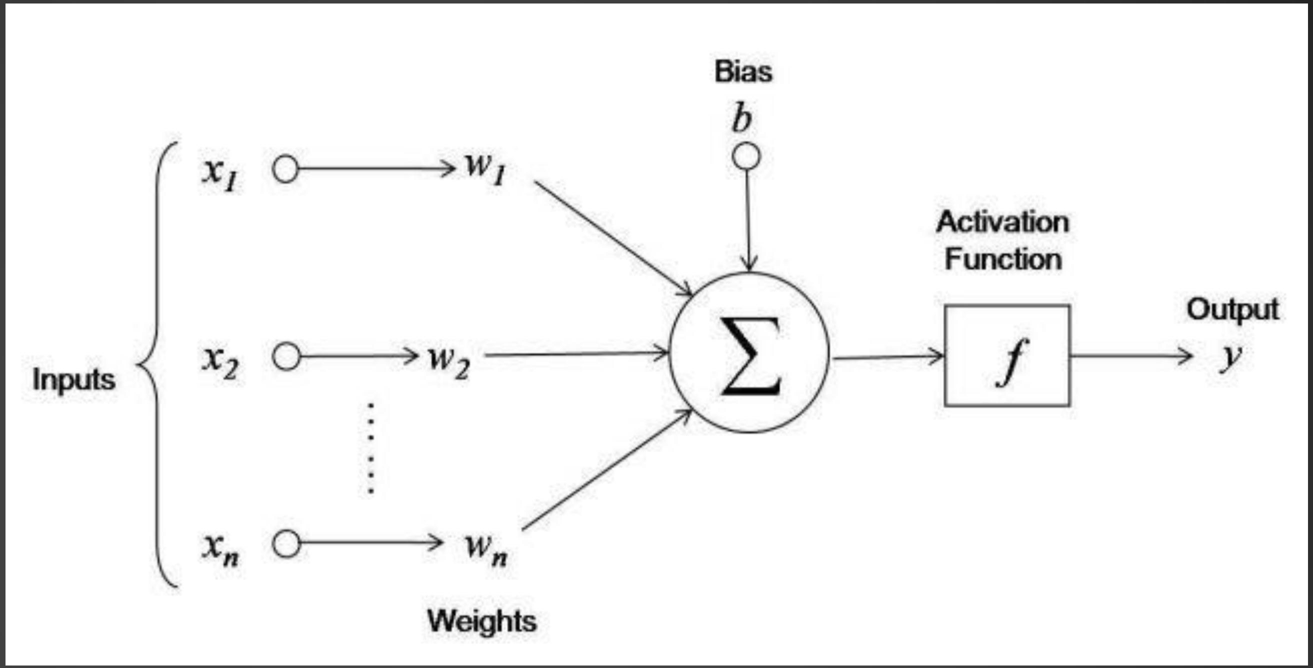
Artificial Neural Network



5. Supervised Learning

5.7. Artificial Neural Network

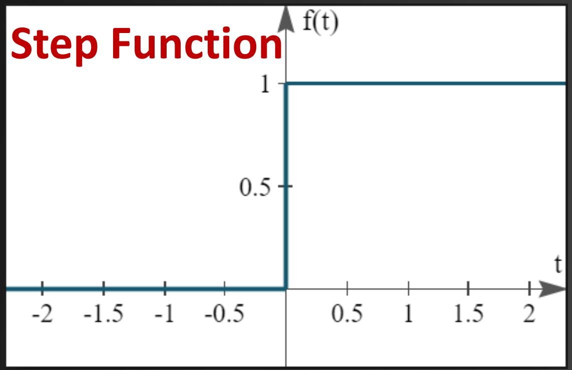
Neuron formulation

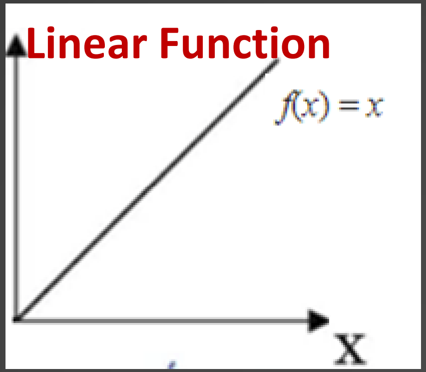


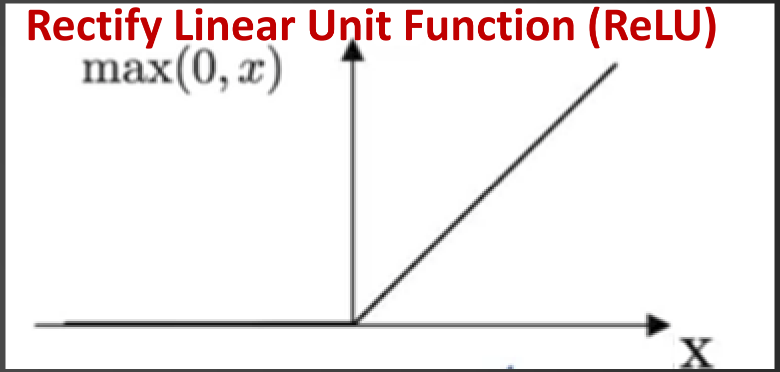
5. Supervised Learning

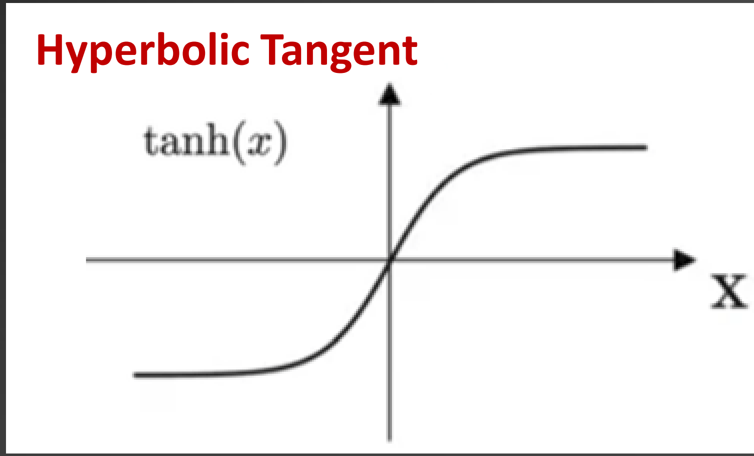
5.7. Artificial Neural Network

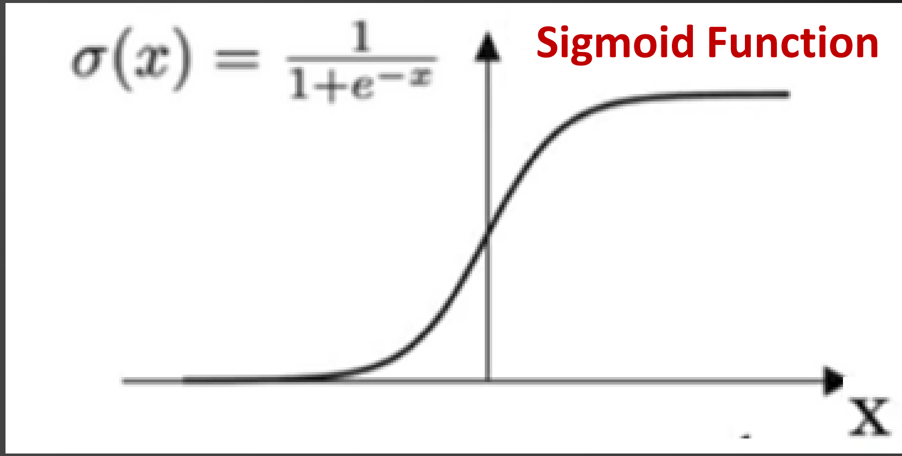
Activation Function







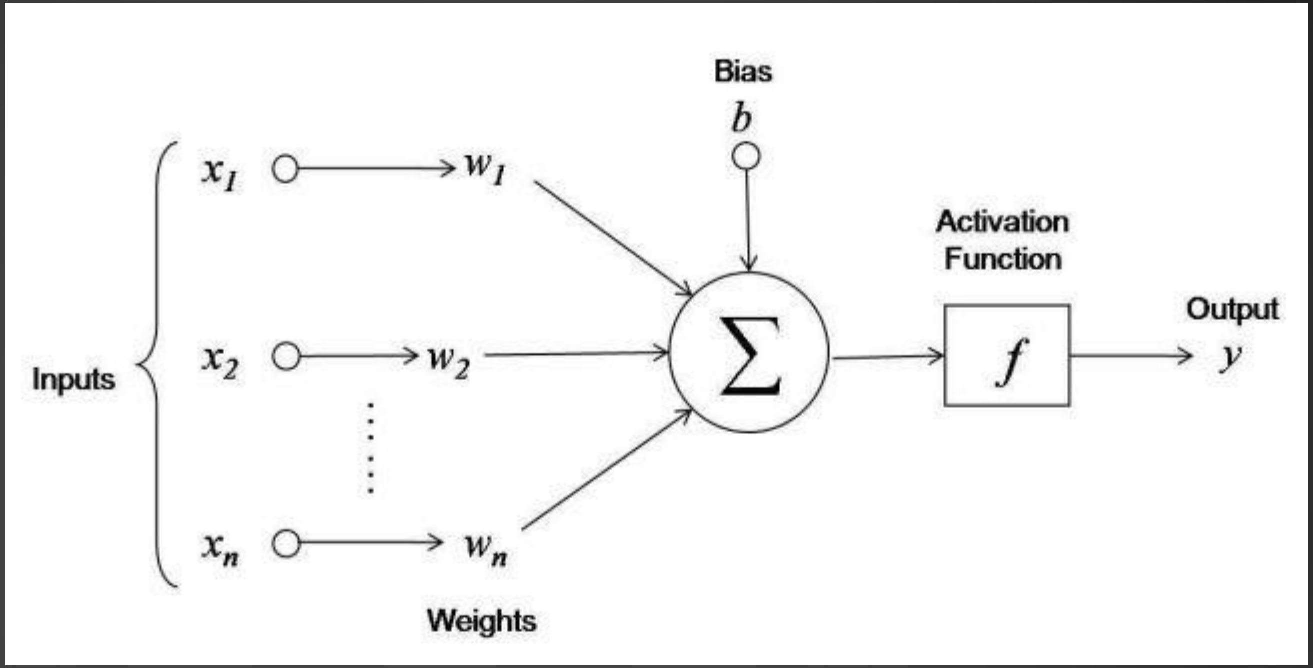




5. Supervised Learning

5.7. Artificial Neural Network

Neuron formulation

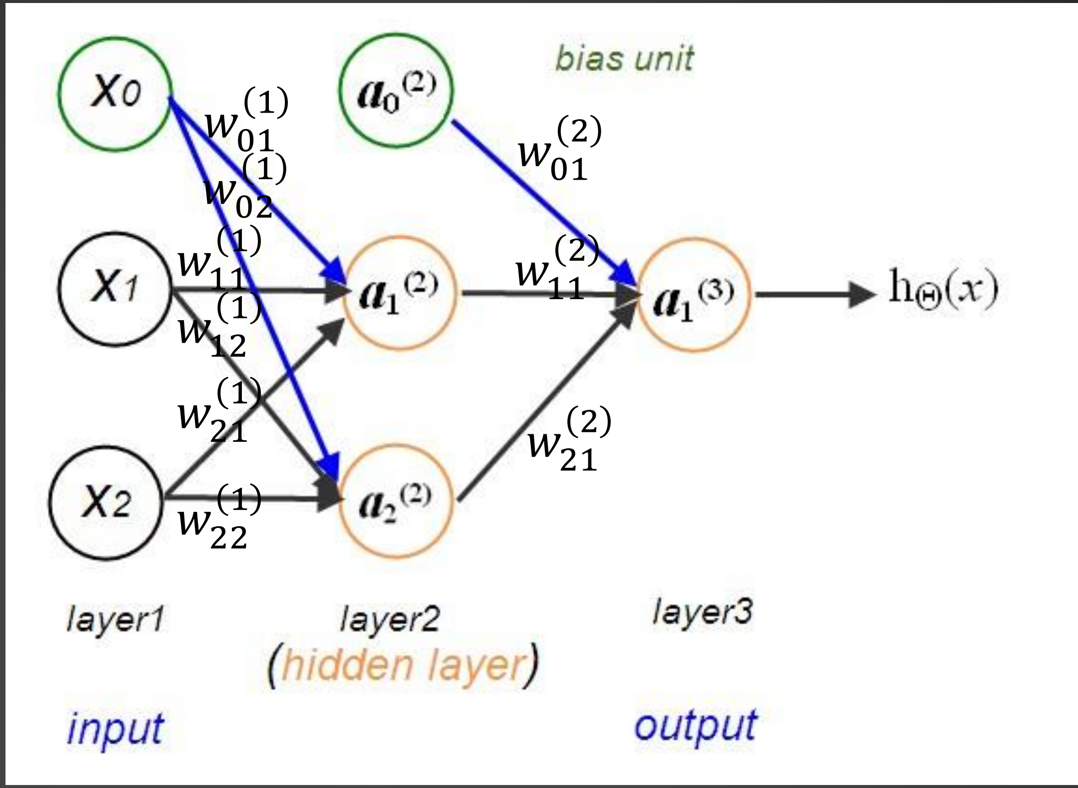


$$y = g(x_1 w_1 + x_2 w_2 + \ldots + x_n w_n + b)$$

5. Supervised Learning

5.7. Artificial Neural Network

Neural Network formulation



$$a_1^{(2)} = g\left(w_{01}^{(1)}x_0 + w_{11}^{(1)}x_1 + w_{21}^{(1)}x_2\right)$$
$$a_2^{(2)} = g\left(w_{02}^{(1)}x_0 + w_{12}^{(1)}x_1 + w_{22}^{(1)}x_2\right)$$
$$h_\Theta(x) = a_1^{(3)} = g\left(w_{01}^{(2)}a_0^{(2)} + w_{11}^{(2)}a_1^{(2)} + w_{21}^{(2)}a_2^{(2)}\right)$$

5. Supervised Learning

5.7. Artificial Neural Network

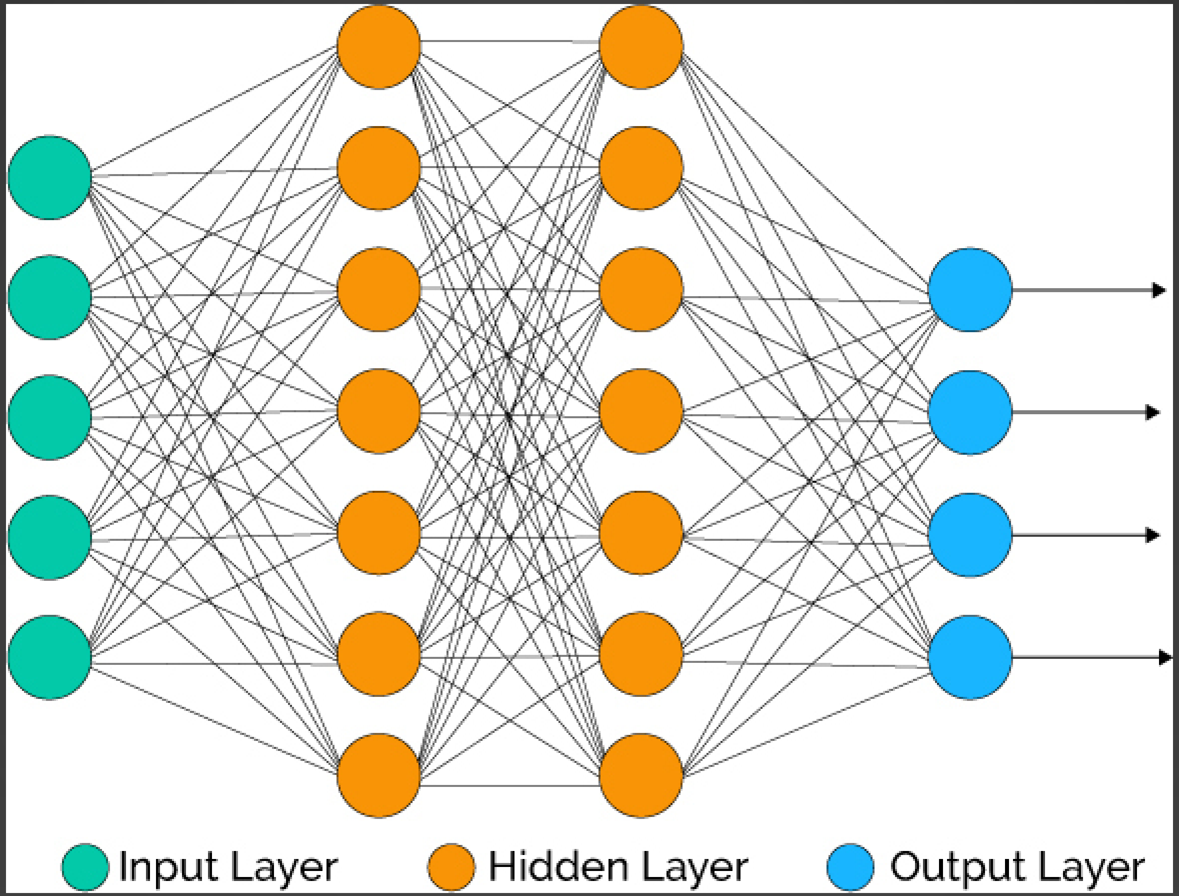
Types of Neural Network

Feed-forward Neural Networks	Feedback (Recurrent) Neural Networks
• Signals to travel one way only: from input to output. • There is no feedback (loops) • They are extensively used in pattern recognition.	• Signals travel in both directions by introducing loops in the network. • Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. • Use in many other applications

5. Supervised Learning

5.7. Artificial Neural Network

Example Neuron Network



5. Supervised Learning

5.7. Artificial Neural Network

Neuron Network in R

```
Install.packages("neuralnet")
```

5. Supervised Learning

5.8. Support Vector Machine

1. What is Support Vector Machine?
2. How does it work?
3. How to implement SVM in R?
4. How to tune Parameters of SVM?
5. Pros and Cons associated with SVM

5. Supervised Learning

5.8. Support Vector Machine

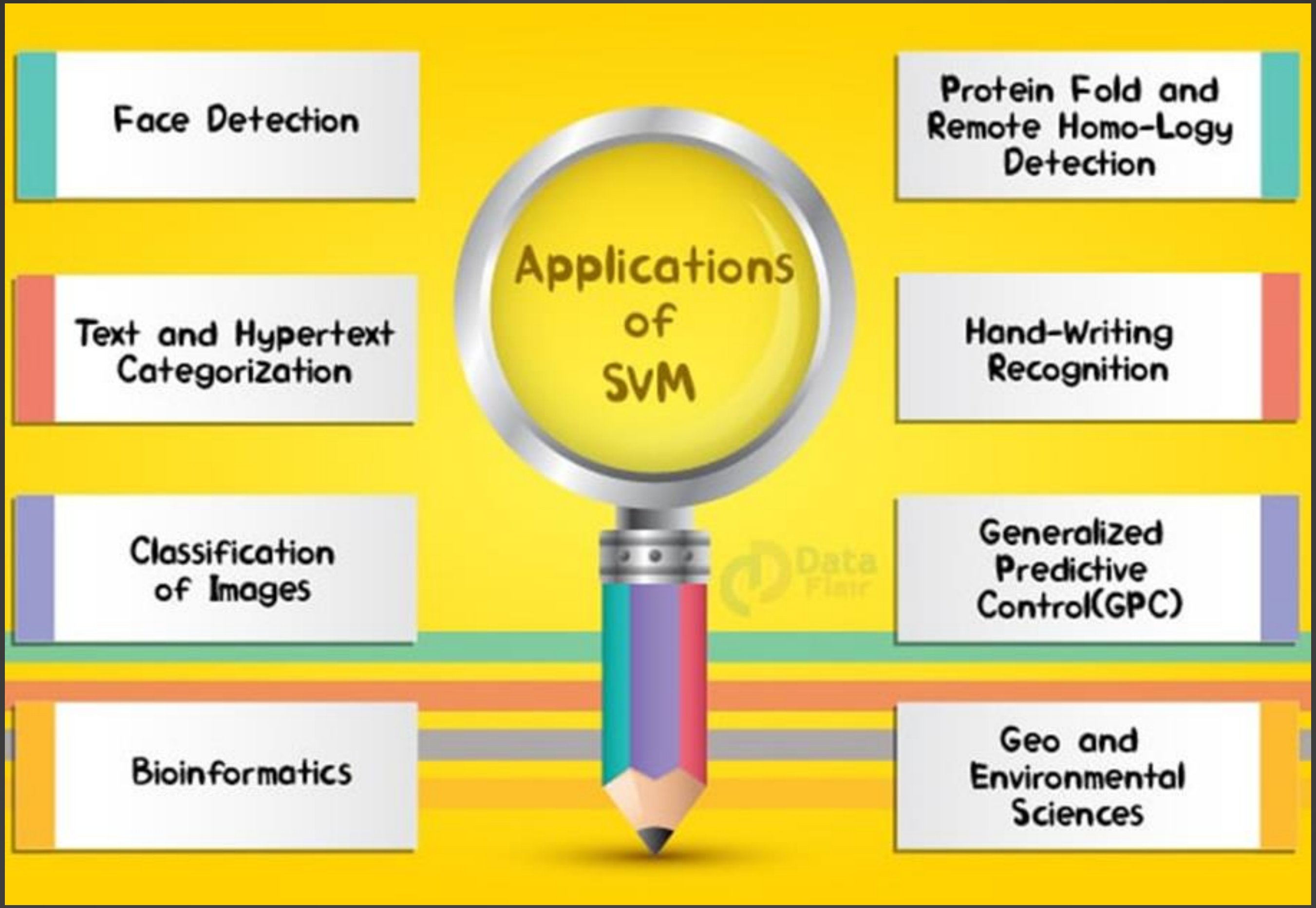
1.What is Support Vector Machine?

- Supervised ML
- Work with both classification & regression
- Best to separate the 2 classes
- Work in both linear and nonlinear classification

5. Supervised Learning

5.8. Support Vector Machine

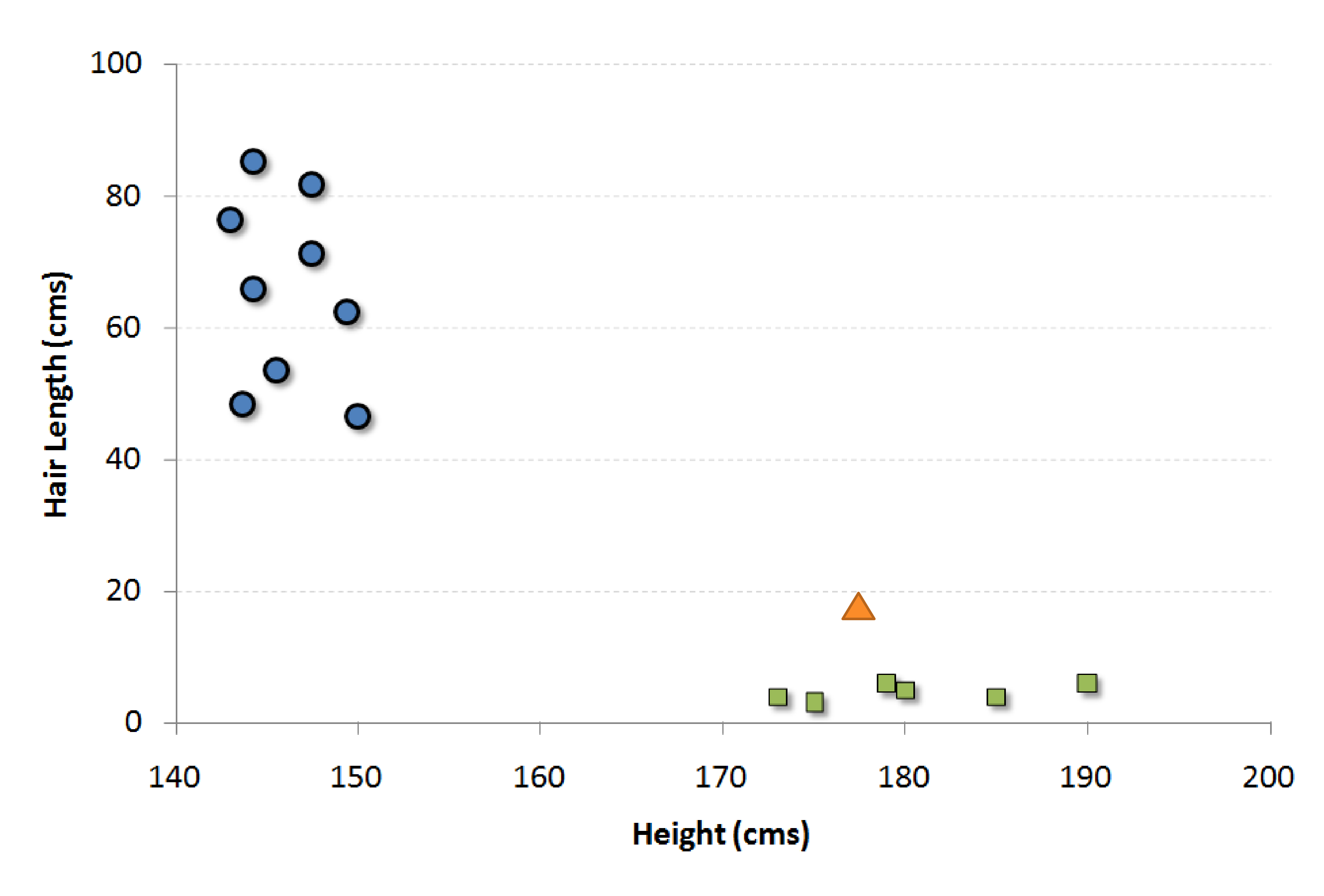
1.What is Support Vector Machine?



5. Supervised Learning

5.8. Support Vector Machine

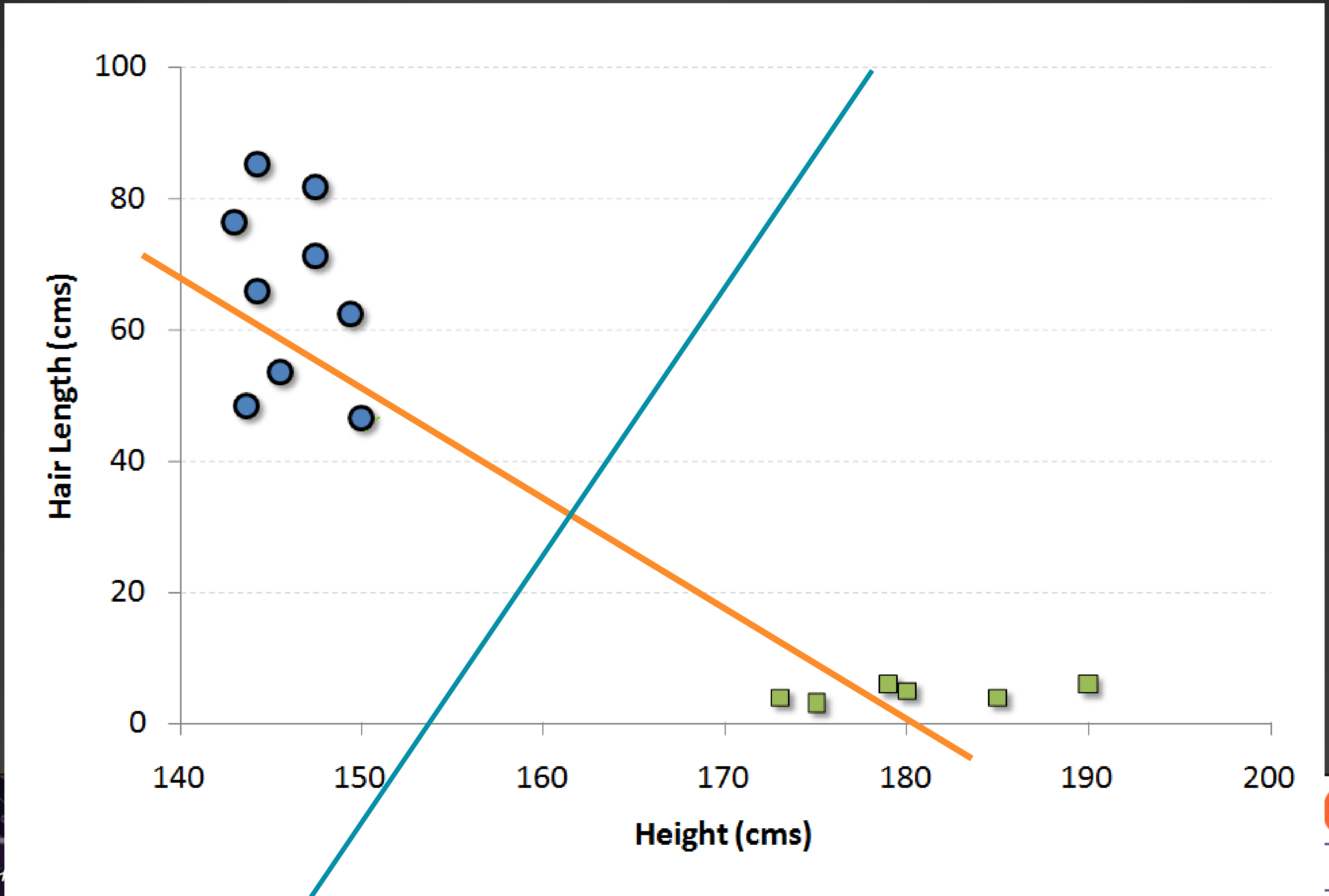
2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

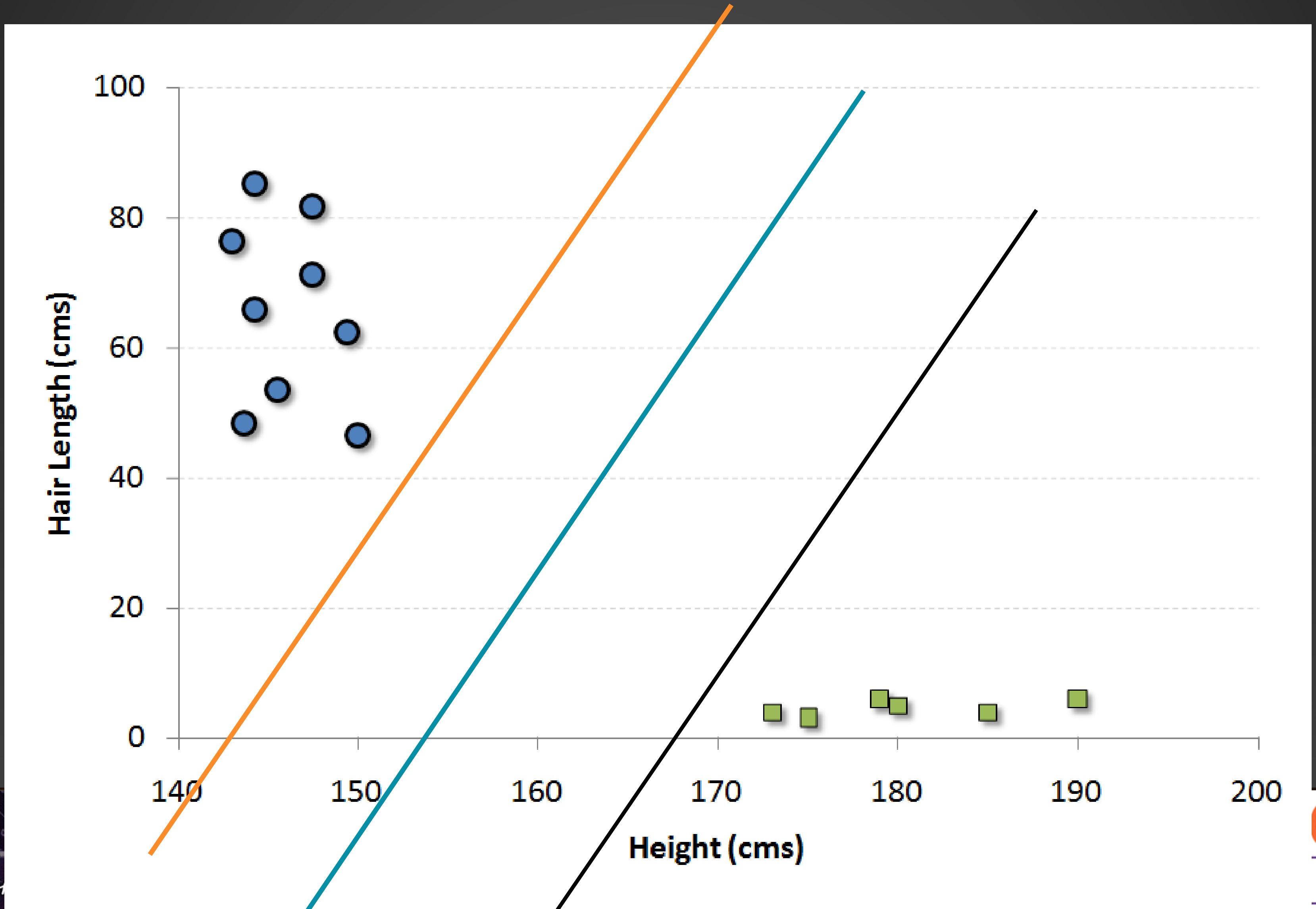
2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

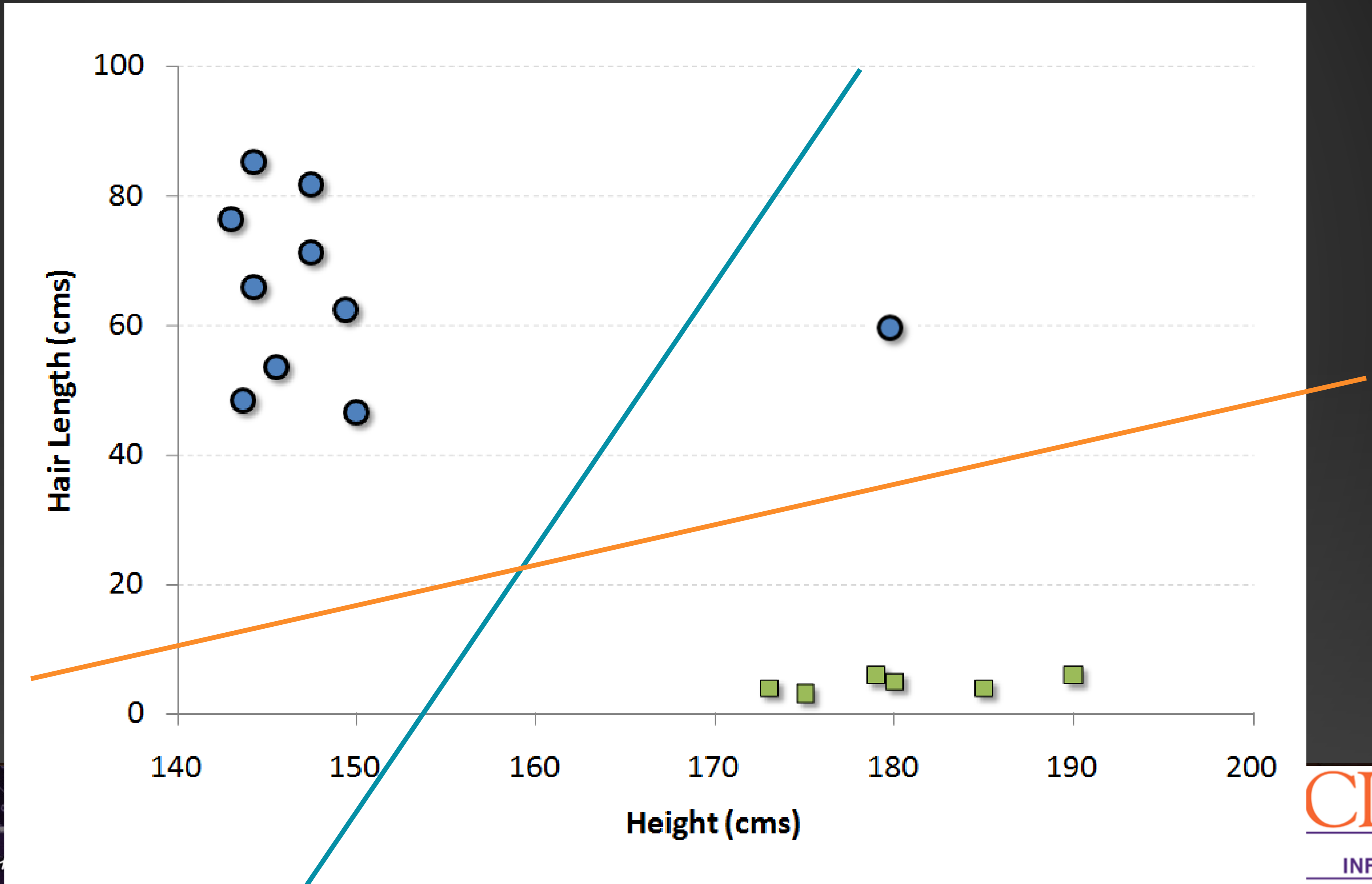
2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

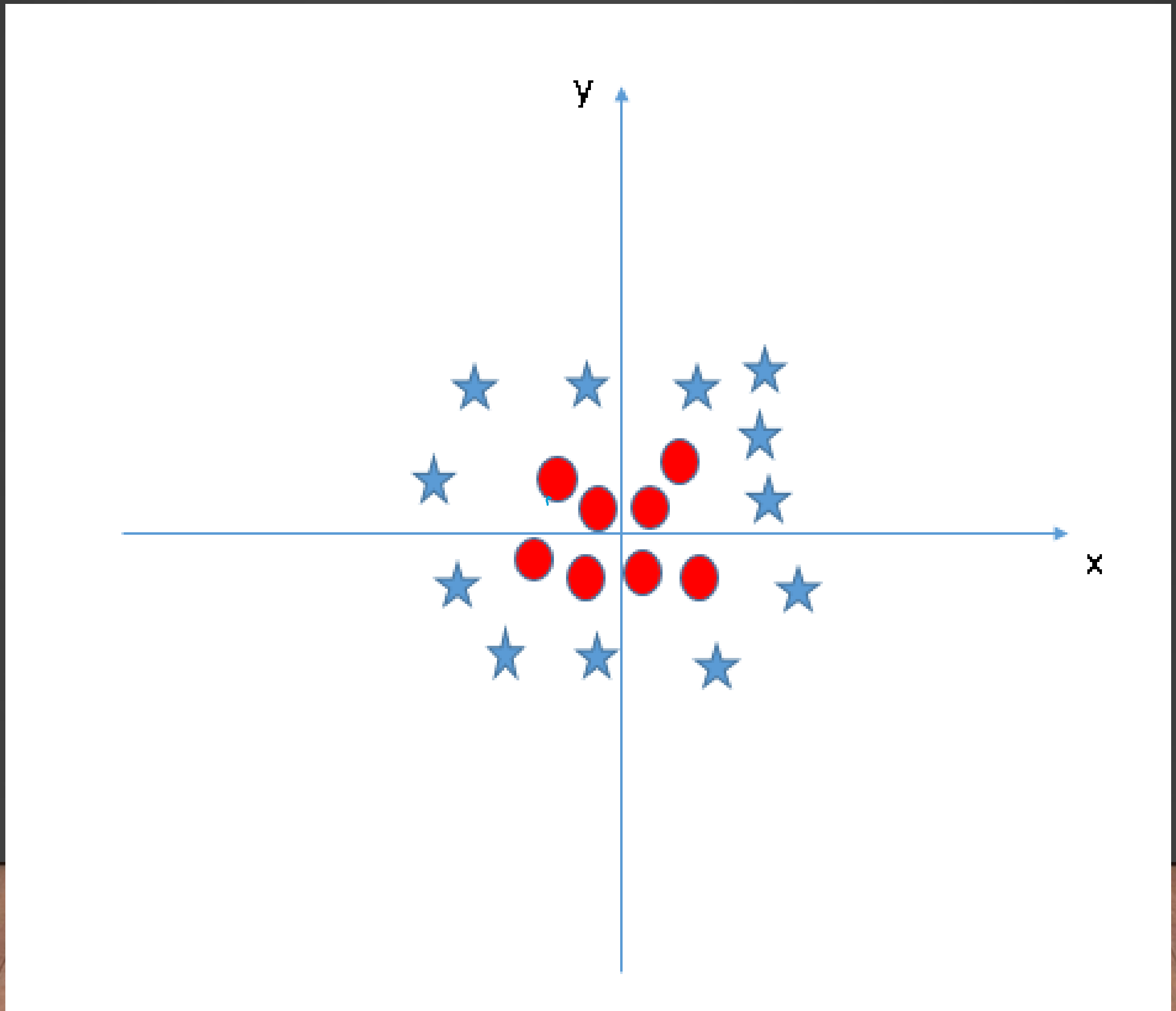
2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

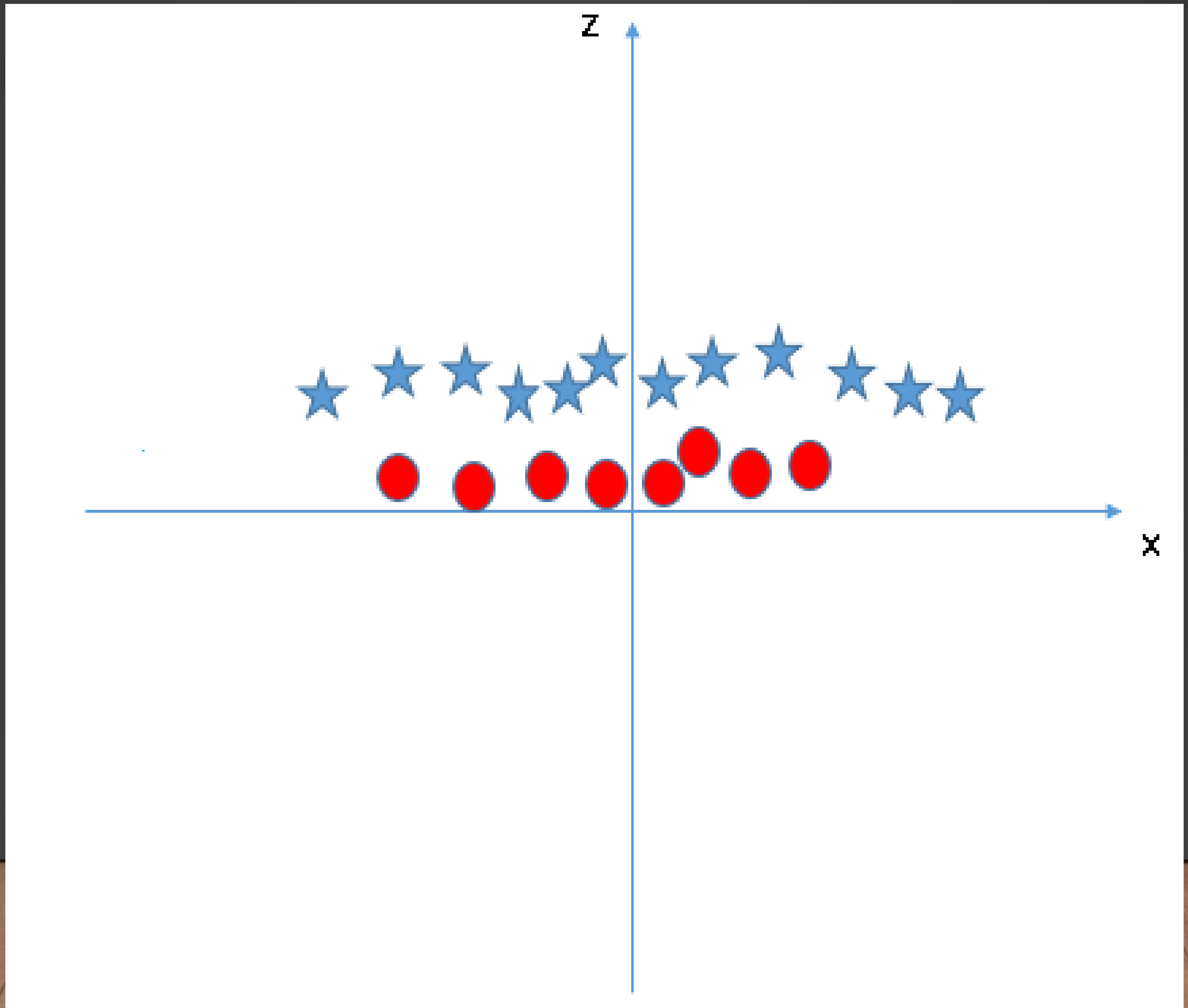
2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

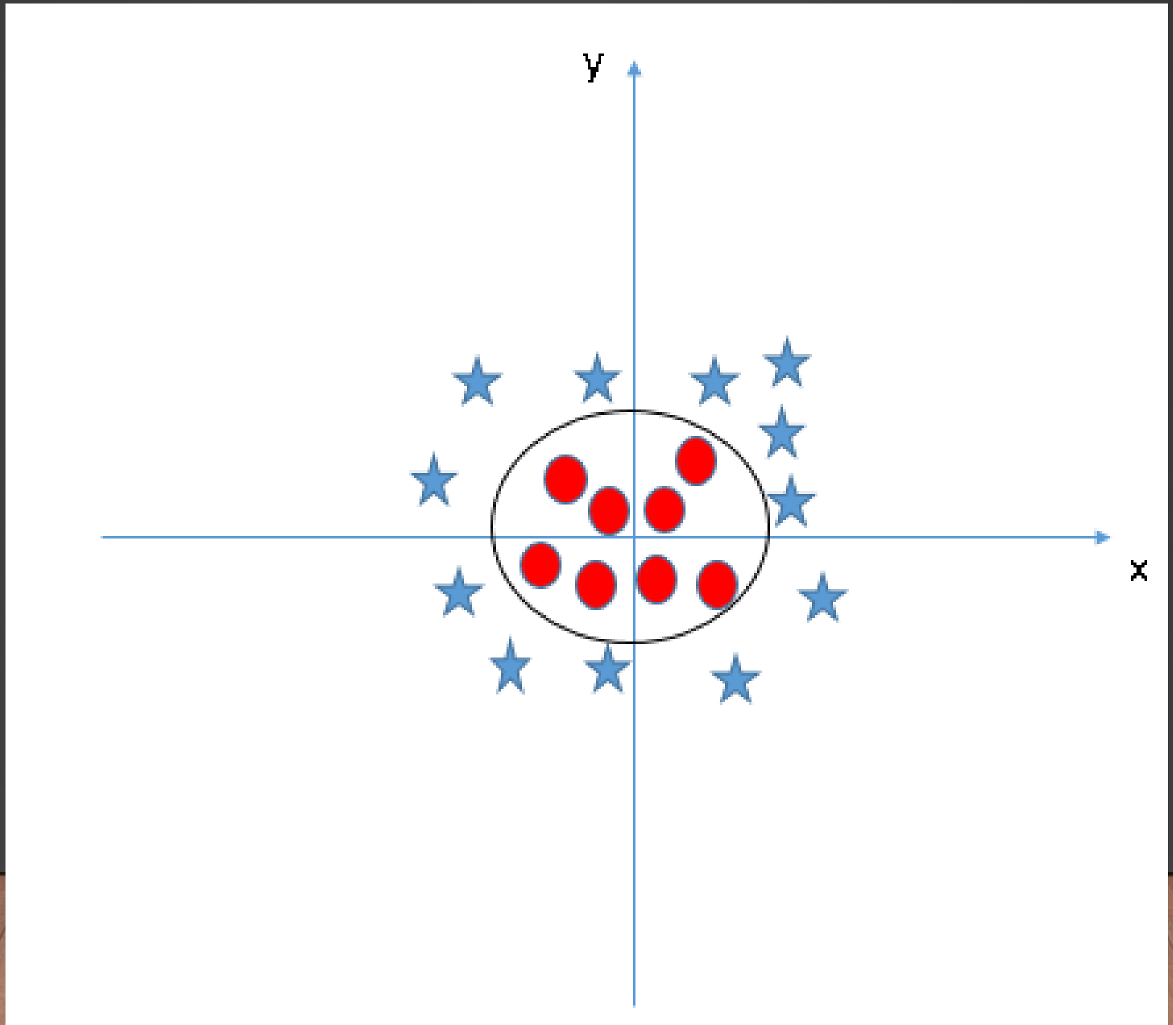
2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

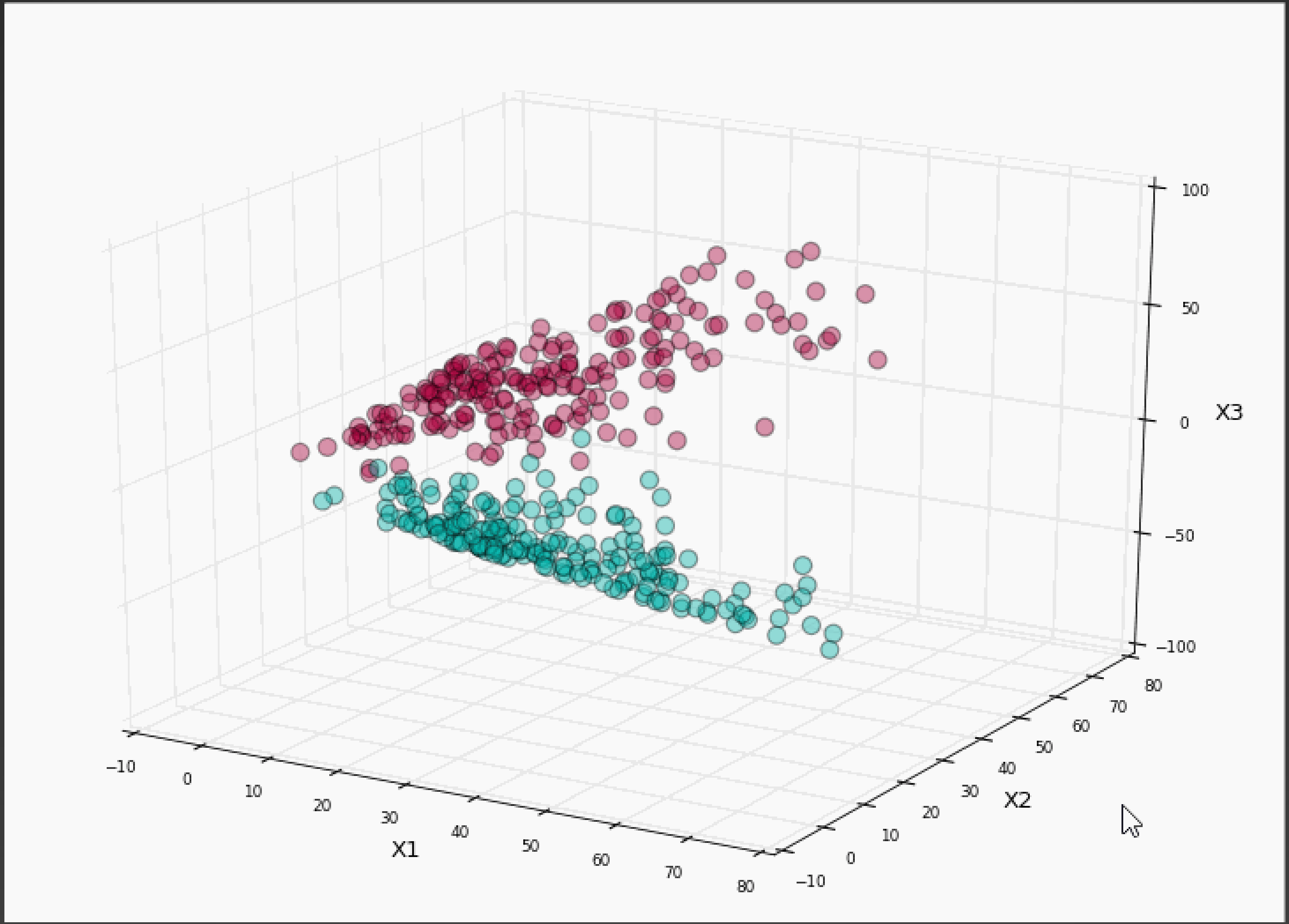
2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

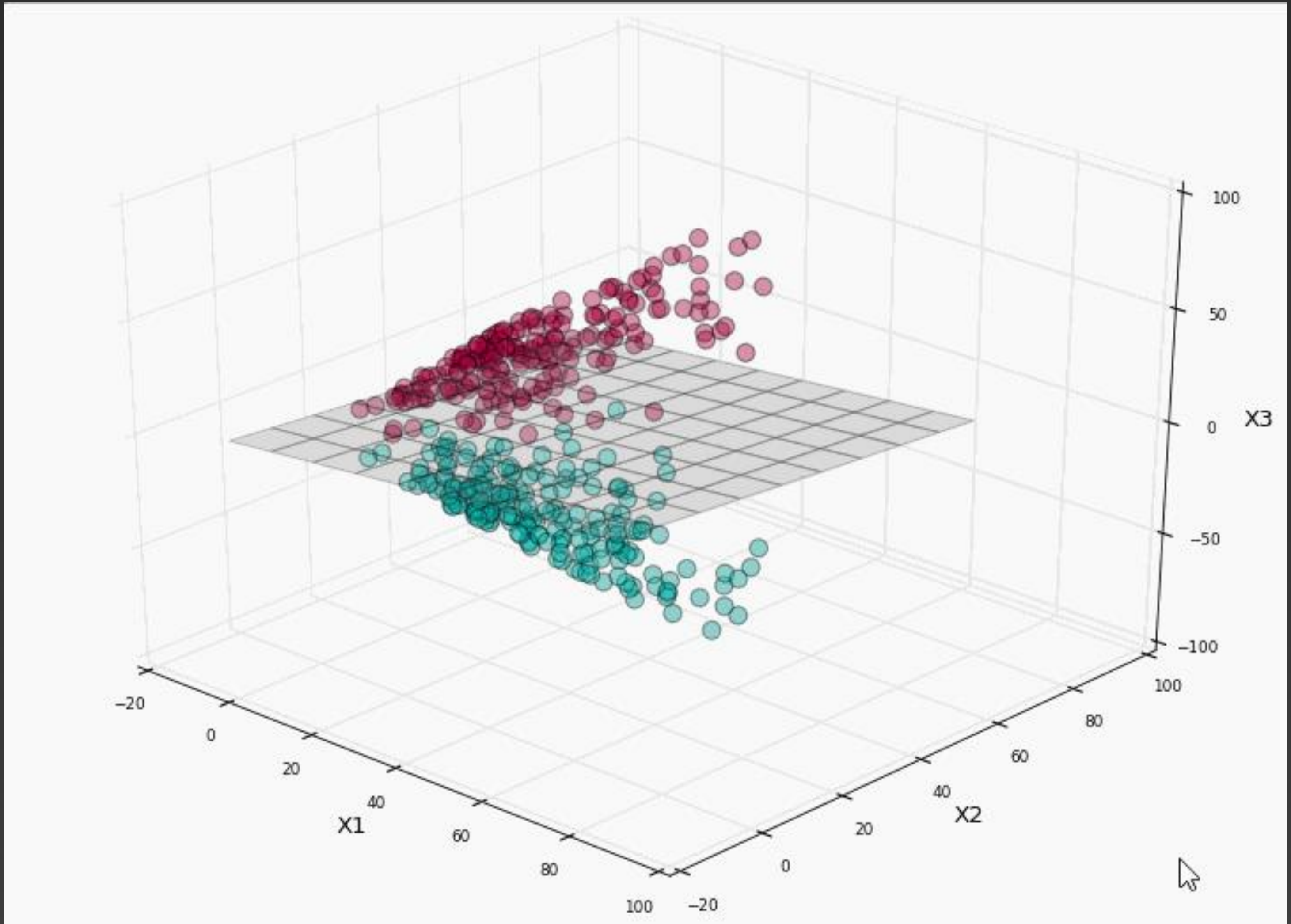
2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

2. How does it work?



5. Supervised Learning

5.8. Support Vector Machine

3.How to implement SVM in R? (caret package)

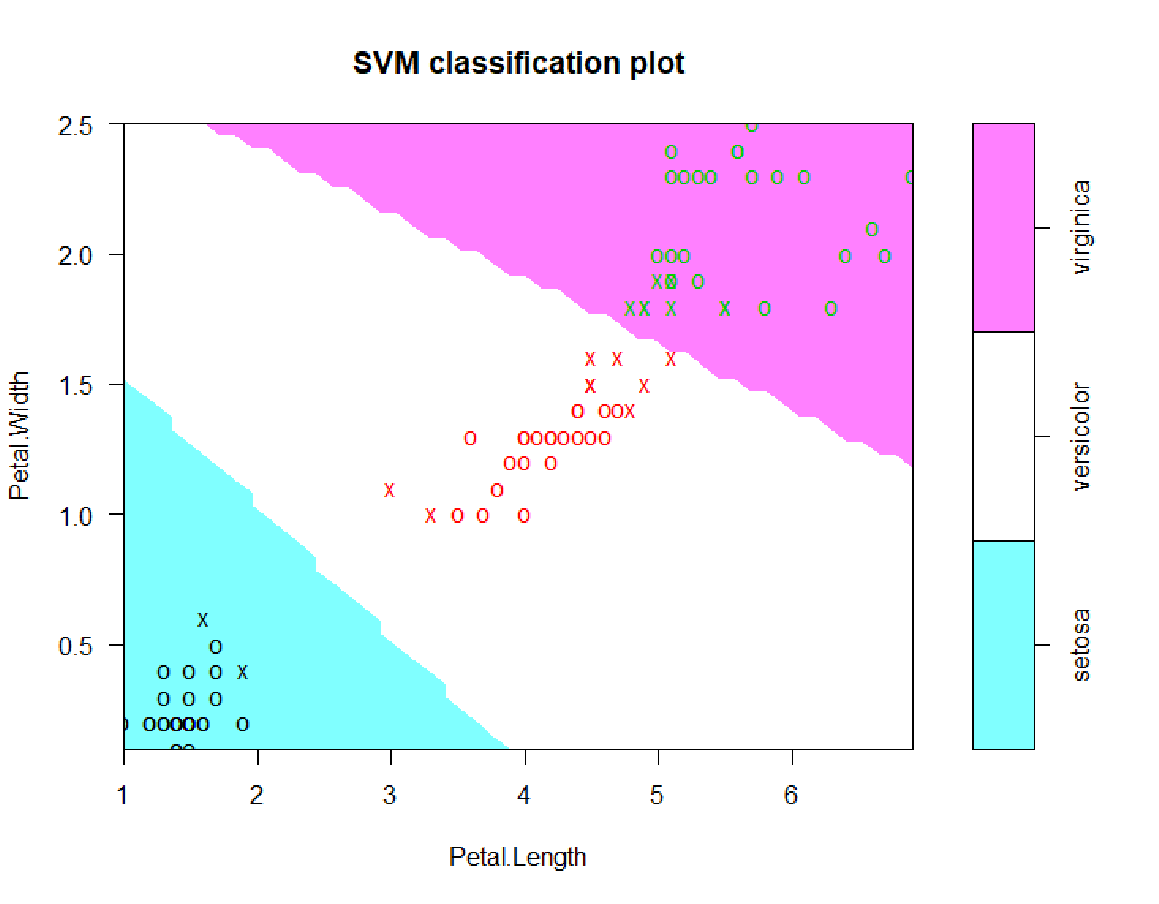
```
library(caret)
data(iris)
set.seed(123)
indT <- createDataPartition(y=iris$Species,p=0.6,list=FALSE)
training <- iris[indT,]
testing  <- iris[-indT,]

ModFit_SVM <- train(Species~.,training,method="svmLinear",preProc=c("center","scale"))
predict_SVM<- predict(ModFit_SVM,newdata=testing)
confusionMatrix(testing$Species,predict_SVM)

#Other function: “svmPoly”, “svmRadial”, “svmRadialCost”, “svmRadialSigma”, etc.
```

5. Supervised Learning

5.8. Support Vector Machine

3.How to implement SVM in R? (e1071 package)

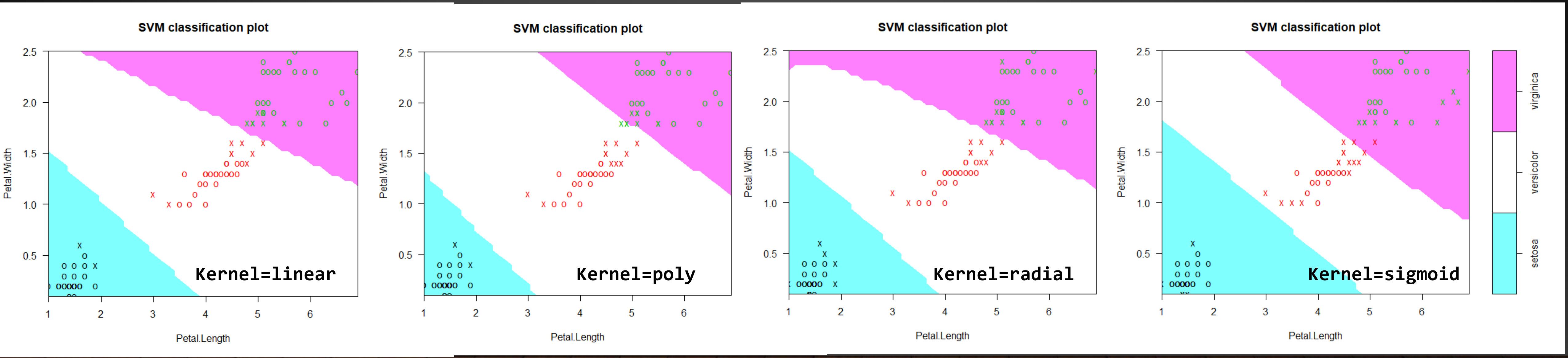
```
library(e1071)
Fit_SVM_ln <- svm(Species~Petal.Width+Petal.Length,
                  data=training,kernel="linear")
plot(Fit_SVM_ln,training[,3:5])
```



5. Supervised Learning

5.8. Support Vector Machine

3.How to implement SVM in R?

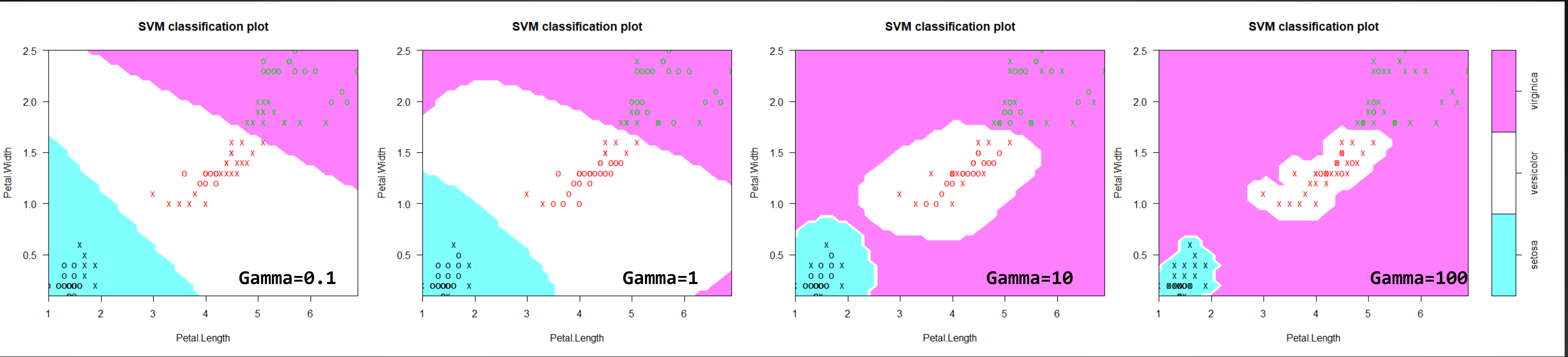


5. Supervised Learning

5.8. Support Vector Machine

3.How to implement SVM in R?

```
Fit_SVM_rbg <- svm(Species~Petal.Width+Petal.Length,
               data=training,kernel="radial",gamma=0.1)
plot(Fit_SVM_rbg,training[,3:5])
```



5. Supervised Learning

5.8. Support Vector Machine

3.How to implement SVM in R?

```
pred_rbg <- predict(Fit_SVM_rbg,testing)
confusionMatrix(testing$Species,pred_rbg)
```

5. Supervised Learning

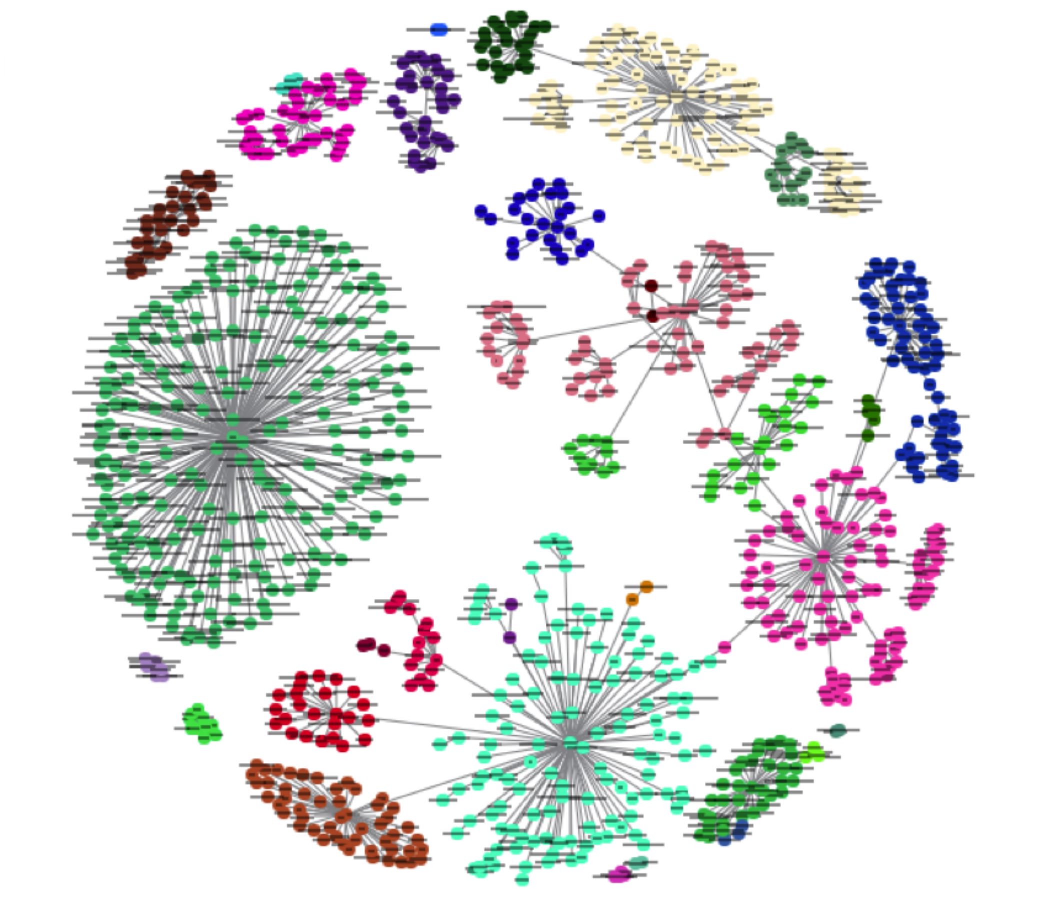
5.8. Support Vector Machine

- **Pros:**
  - It works really well with clear margin of separation
  - It is effective in high dimensional spaces.
  - It is effective in cases where number of dimensions is greater than the number of samples.
  - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- **Cons:**
  - It doesn't perform well, when we have large data set because the required training time is higher
  - It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping
  - SVM doesn't directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is related SVC method of Python scikit-learn library.

5. Supervised Learning

5.9. K-Nearest Neighbour

- Simplicity but powerful and fast for certain task
- Work for both **classification** and regression
- Named as ***Instance Based Learning; Non-parametrics; Lazy learner***
- Work well with small number of inputs

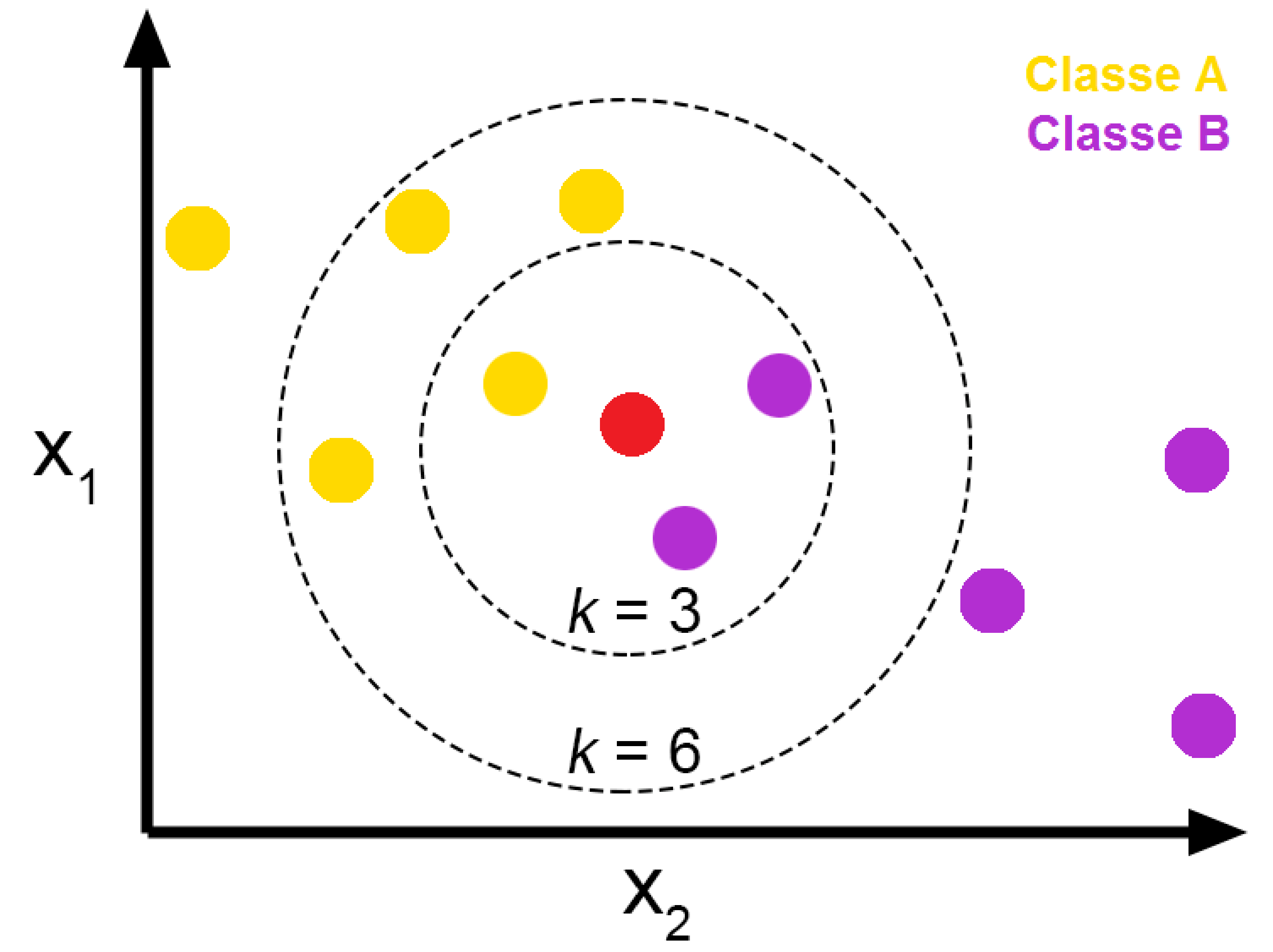


CITI
Cyberinfrastructure Technology Integration

CLEMSON
COMPUTING AND INFORMATION TECHNOLOGY

5. Supervised Learning

5.9. K-Nearest Neighbour



Distances computation:

$$D_{Euclide} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

$$D_{Manhattan} = \sum_{i=1}^{n}|x_i - y_i|$$

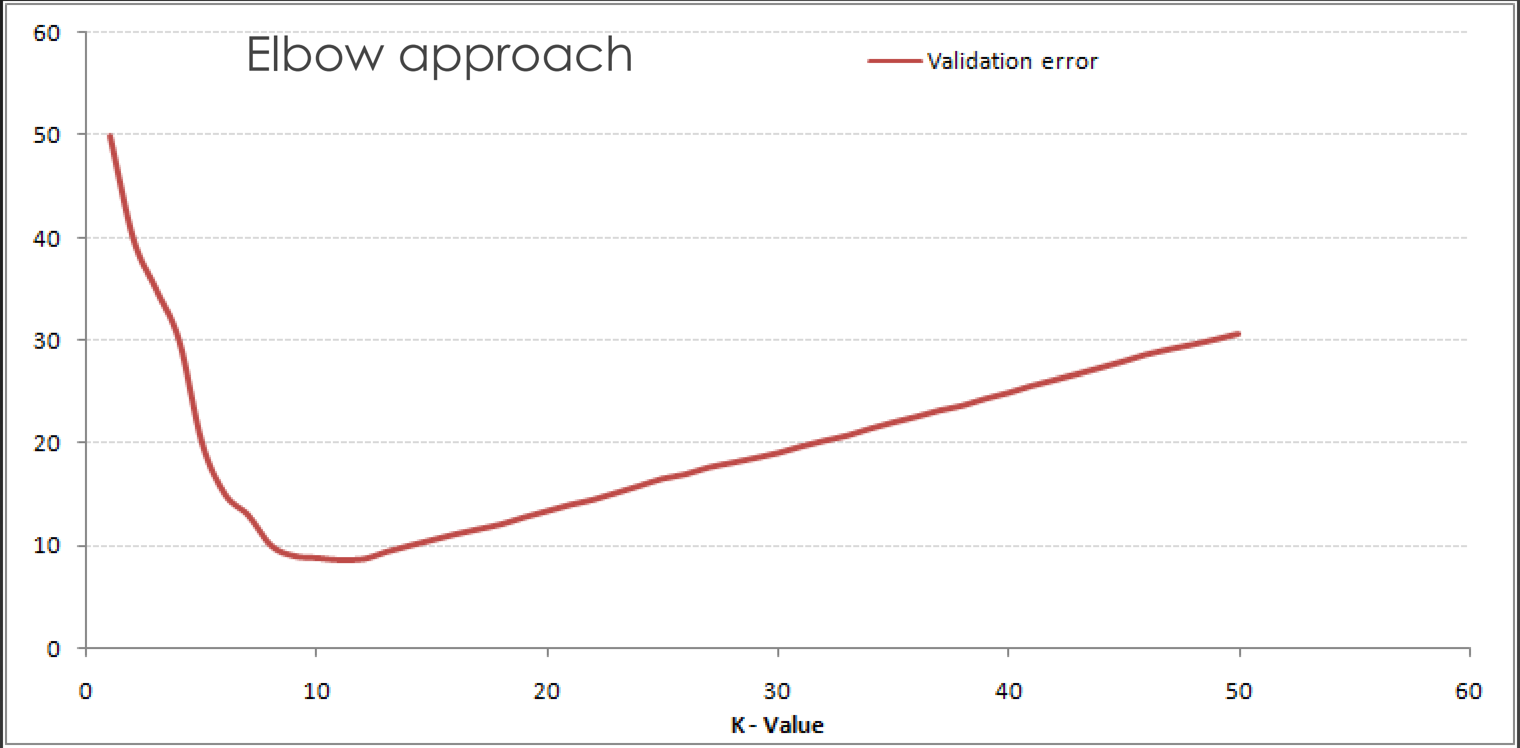
$$D_{Hamming} = \sum_{i=1}^{n}|x_i - y_i| \begin{cases} D = 0(x = y) \\ D = 1(x \neq y) \end{cases}$$

Other distance:
Mahalanobis, Minkowski, Tanimoto, Jaccard

5. Supervised Learning

5.9. K-Nearest Neighbour

Optimal K?



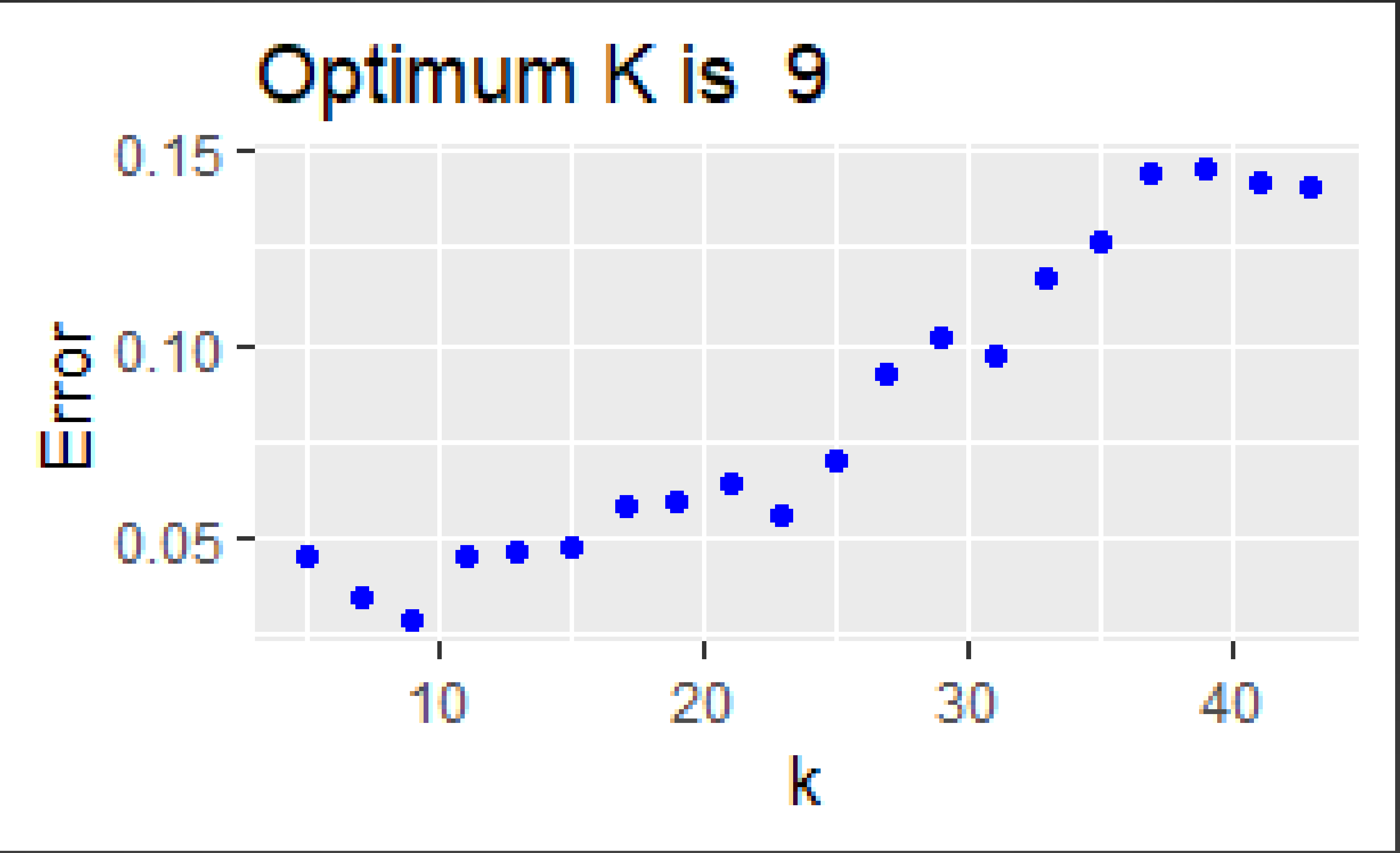
5. Supervised Learning

5.9. K-Nearest Neighbour

```
library(caret)
data(iris)
set.seed(123)
indT <- createDataPartition(y=iris$Species,p=0.6,list=FALSE)
training <- iris[indT,]
testing  <- iris[-indT,]

ModFit_KNN <-
train(Species~.,training,method="knn",preProc=c("center","scale"),tuneLength=20)

ggplot(ModFit_KNN$results,aes(k,AccuracySD))+
      geom_point(color="blue")+
      labs(title=paste("Optimum K is ",ModFit_KNN$bestTune),
           y="Error")

predict_KNN<- predict(ModFit_KNN,newdata=testing)
confusionMatrix(testing$Species,predict_KNN)
```

5. Supervised Learning

5.9. K-Nearest Neighbour



5. Supervised Learning

5.9. K-Nearest Neighbour

Pros	Cons
1.Easy to understand 2.No assumptions about data 3.Can be applied to both classification and regression 1.Works easily on multi-class problems	1.Memory Intensive / Computationally expensive 2.Sensitive to scale of data 3.Not work well on rare event (skewed) target variable 4.Struggle when high number of independent variables

5. Supervised Learning

Conclusions

Model	Sub-model	Type	Note
	Linear Regression	Continuous	
	Multi-Linear Regression	Continuous	
	Principal Component Regression	Continuous	
	Partial Least Square Regression	Continuous	
Regression	Logistic Regression	Categorical	non-linear
	Decision Tree	Both (Categorical)	Ability to map Non-linear, Non-parametric
Tree-based	Random Forest	Both (Categorical)	
	Bagging	Both	
	Boosting-Adaboost	Both	
Ensemble	Boosting Gradient Boosting Machine	Both	
	Naïve Bayes	Both	Naïve assumption of independent variables
Model Based	Linear Discriminant Analyis	Categorical	
	Ridge Regression	Continuous	Good for large data
	LASSO	Continuous	Good for large data
Regularization	ElasticNets	Continuous	Good for large data
Dimension Reduction	PCA	Both (Continuous)	Good for large data
	Neural Network	Both	Applied in many field in supervised/unsupervised
	Support Vector Machine	Both (Categorical)	Not good for large data
	KNN	Both	Small data, skew with outliers

CITI
Cyberinfrastructure Technology Integration

CLEMSON
COMPUTING AND INFORMATION TECHNOLOGY