**Table of Contents**

[Chapter 1 Basic Terminology and Concepts 3](#_Toc10840314)

[Chapter 2 linear Regression 4](#_Toc10840315)

[Linear Regression 4](#_Toc10840316)

[Chapter 3 Classification 7](#_Toc10840317)

[Logistic 7](#_Toc10840318)

[LDA (Linear Discriminant Analysis) 8](#_Toc10840319)

[QDA (Quadratic Distriminant Analysis) 10](#_Toc10840320)

[Naïve Bayes 10](#_Toc10840321)

[KNN (K-nearest neighbors) 10](#_Toc10840322)

[Chapter 4 Resampling Methods 11](#_Toc10840323)

[Cross-Validation 11](#_Toc10840324)

[Bootstrap 11](#_Toc10840325)

[R 11](#_Toc10840326)

[Chapter 5 Variable Selection 11](#_Toc10840327)

[Subset Selection 12](#_Toc10840328)

[Shrinkage 13](#_Toc10840329)

[Dimension Reduction 14](#_Toc10840330)

[Chapter 6 Non-Linear Models 15](#_Toc10840331)

[Polynomial Regression 15](#_Toc10840332)

[Chapter 7 Decision Trees 16](#_Toc10840333)

[No1: Bagging 17](#_Toc10840334)

[No2: Random Forests 17](#_Toc10840335)

[No3: Boosting 17](#_Toc10840336)

[Chapter 8 Support Vector Machines 18](#_Toc10840337)

[Support Vector Machines (SVM) – Regression and classification 18](#_Toc10840338)

[Chapter 9 Unsupervised Learning 20](#_Toc10840339)

[PCA (Principal Components Analysis) 21](#_Toc10840340)

[Clustering <- Discovering unknown subgroups in data 21](#_Toc10840341)

[Deep Learning 23](#_Toc10840342)

[artificial Neural Network 24](#_Toc10840343)

[Activation function σ(WiXi + b) 24](#_Toc10840344)

[Convolutional neural networks (CNN) 29](#_Toc10840345)

[Recurrent Neural Networks (RNN) 32](#_Toc10840346)

[Self Organizing Maps 34](#_Toc10840347)

[Boltzmann machines 36](#_Toc10840348)

[autoencoders 38](#_Toc10840349)

[Others 41](#_Toc10840350)

# Chapter 1 Basic Terminology and Concepts

1. **Application of Machine Learning**
   1. Database mining
      1. Large datasets from growth of automation/web.
      2. E.g., Web click data, medical records, biology, engineering
   2. Applications can’t program by hand.
      1. E.g., Autonomous helicopter, handwriting recognition
   3. Natural Language Processing (NLP), Computer Vision
   4. Self-customizing programs
      1. E.g., Amazon, Netflix product recommendations
   5. Understanding human learning (brain, real AI).
2. First definition
   1. Field of study that gives computers the ability to learn without being explicitly programmed
3. **Supervised and Unsupervised**
   1. Supervised
      1. Dependent Variable Y: Output, Response, Target
      2. Independent Variable X: Inputs, Variables, Predictors, Features, Attributes
   2. Unsupervised
      1. No output
      2. Fuzzy objective
         1. Find groups of samples that behave similarly
         2. Find features that behave similarly
         3. Find linear combinations of features with the most variation
4. **Regression and Classification**
   1. Regression
      1. Regression models (both linear and non-linear) are used for predicting a continuous value, like salary for example.
         1. If your independent variable is time, then you are forecasting future values
      2. Models:
         1. Linear Regression
         2. Polynomial Regression
         3. Support Vector for Regression (SVR)
         4. Decision Tree Classification
         5. Random Forests Regression.
   2. Classification
      1. Unlike regression where you predict a continuous number, you use classification to predict a category (discrete value).
      2. Models:
         1. Logistic Regression (linear)
            1. When you want to rank the predictions by their probability
         2. Naïve Bayes (nonlinear)
            1. Same as LR
         3. SVM/ Kernel SVM
            1. When you want to predict to which segment your customers belong to
         4. Decision Tree Classification
            1. When you want to have clear interpretation of your model results
         5. Random Forests Classification
            1. When you are just looking for high performance with less need for interpretation
         6. K-Nearest Neighbors (K-NN)
5. **Two reasons to estimate f(X)**
   1. Prediction
   2. Inference
      1. Which predictors are associated with the response?
      2. What is the relationship between the response and each predictor?
      3. Does the relationship linear or more complicated?
6. **Bias-Variance Trade-off**
   1. Model with high complexity “F(X)” 🡪 Larger variance (overfitting), Less bias
   2. Variance – How much varies between the average of the model and estimate (testing)
      1. Can use a simpler model to lower the variance
      2. Error due to complexity of the models
   3. Bias – How far between the average of the model and the truth
      1. Refers to the modeling error that occurs because you haven’t properly identified the underlying real-world scenario
      2. Will disappear if you add more training data
7. **Assessing Model Accuracy**
   1. Regression:
      1. RMSE/MSE
      2. R Square
   2. Classification:
      1. Misclassification error rate
      2. MLE (Maximum likelihood estimation)
8. **Training and Testing**
   1. Training
      1. Accurately predict unseen test cases
      2. Understand which inputs affect the outcome, and how
      3. Assess the quality of predictions and inferences

# Chapter 2 linear Regression

## Linear Regression

Concept 🡪 Formula (Beta) 🡪 Least Squares & RSS 🡪 Hypothesis Test (F/t) 🡪 p-value 🡪 Confidence Interval 🡪 RSE/R2 (Problem) 🡪 Outliers/Residual/Collinearity

1. **Concept:**
   1. Predict a quantitative response Y by a single/more predictor for variable X
2. **Questions for what we need to solve:**
   1. Any Relationship (multiple)?
      1. H1: at least one Beta(j) is non-zero
         1. Expect F > 1
         2. p must <= n
   2. How strong for the relationship/how well does the model fit the data?
      1. RMSE/RSE
      2. R2 (Coefficient of determination)
   3. Which variables contribute?
      1. P-value
   4. How accurately of prediction?
      1. Confidence interval 🡪 pertain to a mean or other statistic calculated from multiple values
         1. May narrower, because the correlation among the error terms (Such as time series data)
      2. Prediction interval 🡪 pertains to uncertainty around a single value
         1. Wider than CI because it considers the uncertainty associated with
      3. 95% Confidence interval
         1. There is approximately a 95% chance that the interval [A, B] will contain the true value of population mean
   5. Is the relationship linear?
      1. Residual plot
         1. U-shape if non-linearity
   6. Is there Synergy?
      1. Also called - Interaction effect
3. **Formula**
   1. 1. β 🡪 slope/coefficients/parameters
      2. We interpret βj as the average effect on Y of a one unit increase in Xj, holding all other predictors fixed
   2. Sometimes X can be dummy variable
      1. So we can build (n-1) dummy columns instead
4. **Estimation of parameters by least squares**
   1. 1. The least squares approach chooses to minimize the RSS (cost function)
         1. With RSS/2n will be more accuracy
      2. Assessing the accuracy of the coefficient estimates
         1. Confidence intervals
5. **Hypothesis Testing**
   1. H0: There is no relationship between X and Y (null hypothesis)
      1. H0: β1 = 0
   2. HA: There is some relationship between X and Y (alternative hypothesis)
      1. HA: β1 ≠ 0
   3. T-distribution with n-2 degrees of freedom
   4. P-value
      1. Determine the significance (a) of results after a statistical hypothesis test.
      2. Representing the probability of the occurrence of a given event (Null hypothesis)
      3. P-value < 0.05 🡪 Strong evidence against the null hypothesis 🡪 reject
6. **Assessing the over accuracy of the model (RSE & R2)**
   1. 1. Assessing model accuracy
      2. Estimate the SD of error
      3. The average amount that the response will deviate from the true regression line
      4. Same as RMSE, but adjusted for degrees of freedom

* + 1. Measures how close the data are to the fitted regression line, from 0 to 1
    2. When add more variables, R2 always increase, because the RSS will keep finding a good coefficient to lower the RSS
       1. Problem: we don’t know whether the new variable is helpful
       2. Solution: use Adj R Square
          1. P can be a penalization factor. When p increase, the adj R square will decrease a little bit. We are going to compare the effect difference between p and R square
  1. Correlation coefficient
     1. Correlation between X and Y - a measure of how close data are to a line
     2. (Similar with R square, but not good for multiple regression since this association between a single pair of variables)

1. **Other Considerations in the regression model**
   1. Qualitative Predictors
      1. Dummy variable
      2. Result:

|  |  |
| --- | --- |
|  | Coefficient |
| Intercept | 500 |
| Gender [Female] | 20 |

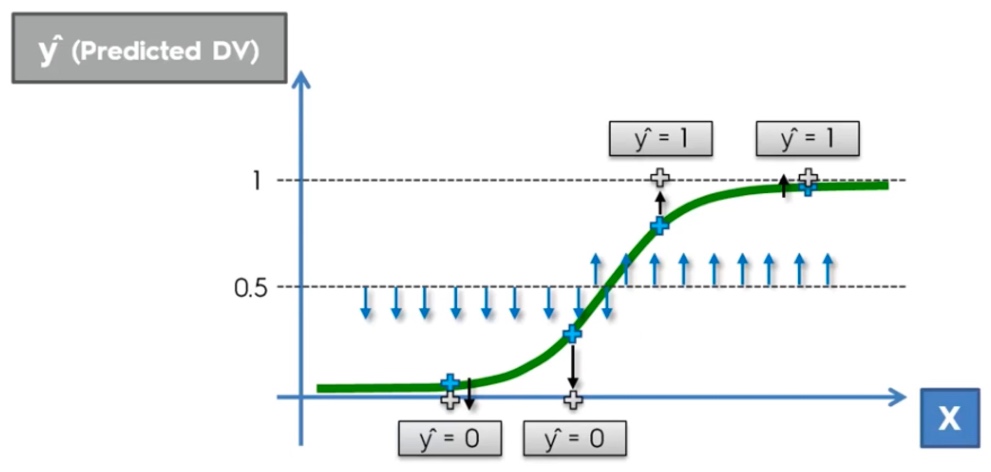
* + - 1. The average credit card debt for males are estimated $500, and females are $520
  1. Interactions (synergy effect)
     1. TV \* radio
     2. Hierarchy Principle: We must include the main effects: TV and radio
  2. Nonlinearity
     1. TV2

1. **Potential Problems**
   1. Outliers
      1. A point that the Yi is far from the value predicted by the model
      2. Residual plots
         1. Sometimes we don’t know how far as an outlier
         2. Plot the studentized residuals (ei /Estimated standard error)
            1. Possible outliers: SR>3
         3. Or check standardized residual
   2. High-leverage points
      1. Has a sizable impact on the estimated regression line (unusual Xi)
         1. Hard to detect in multiple linear regression
      2. How to measure the leverage (the degree of influence that a single record has on a regression equation)
         1. High leverage 🡪 leverage statistic > (p+1)/n
   3. Collinearity
      1. A situation in which two or more predictor variables are closely related to one another
      2. How to detect:
         1. Use correlation if P=2
         2. Compute the VIF (variance inflation factor) that exceeds 5 or 10
      3. Solutions:
         1. Drop one of the problematic variables
         2. Combine the collinear variable together

# Chapter 3 Classification

Concept 🡪 Why not linear? 🡪 Formula (sigmoid/odds) 🡪 Maximum likelihood

## Logistic

1. Concept:
   1. Probability of an observation belonging to each category
2. The Bayes Classifier
   1. Assigns each observation the most likely class, given its predictor values
3. Why not linear regression?
   1. Linear regression might produce probabilities less than zero or bigger than one
   2. Linear regression is not appropriate for implying that the difference between class 1 and class 2 is the same as between class 2 and class 3
   3. Logistic is very popular for classification when K=2
4. Formula:
   1. How do we get the formula?
   2. A one-unit increase in is associated with an increase in the log odds of Y by units.
5. Estimating the Regression Coefficients
   1. Maximum likelihood -> pick all the β to maximize the probability of the observed zeros and ones respectively in the data.
   2. Another cost function: Cross entropy

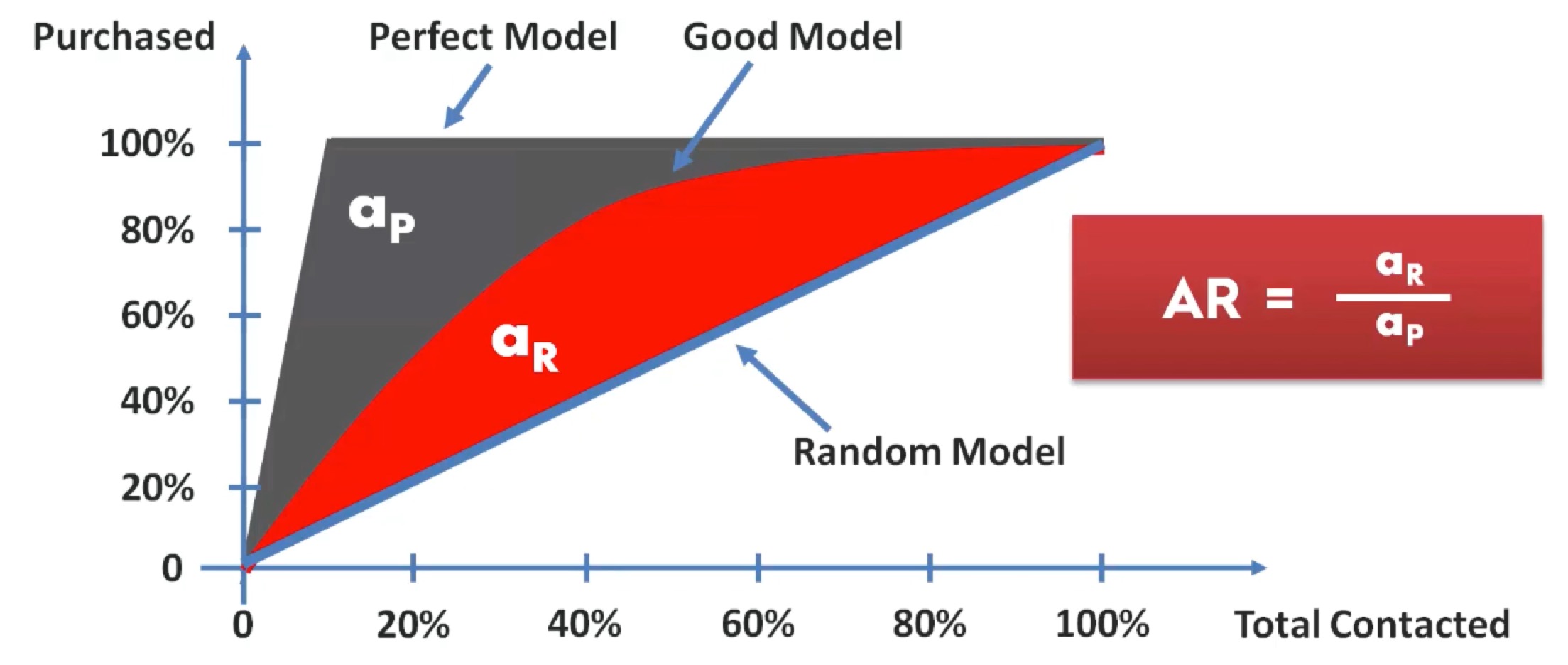
## LDA (Linear Discriminant Analysis)

Concept 🡪 Bayes Theorem 🡪 Why? 🡪 Confusion Matrix (Type I/II error/ROC)

1. Concept:
   1. Model the distribution of X in each of the classes separately and use Bayes theorem to determine the probability of a record belonging to a certain class P(Y|X).
      1. Bayes’ Theorem
         1. Logic
            1. P(A|B) = P(A∩B)/P(B)
            2. P(B|A) = P(A∩B)/P(A) = P(A|B)\*P(B) / P(A)
      2. Similar to PCA – reduce dimension
   2. LDA uses full likelihood P(X, Y) and Logistic Regression uses conditional likelihood P(Y|X)
   3. ? Used as a dimensionality reduction technique
      1. Unlink PCA (unsupervised) – finding the component axises
      2. LDA – interested in the axes that maximize the separation between multiple classes
   4. ? Used in the pre-processing step for pattern classification
2. Why LDA?
   1. When the classes are well-separated
   2. When n is small, and the distribution of the X is approximately normal (more stable)
   3. When there are more than two response classes.
3. Confusion matrix
   1. A table that is often used to describe the performance of a classification model on a set of test data for which the true values are known

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Confusion Matrix | | | | |
|  |  | Actual Value | | |
| N = 555 |  | No | Yes | Total |
| Predict Value | No | TN=300 | FN=50 | 350 |
| Yes | FP=5 | TP=200 | 205 |
| Total | 305 | 250 |  |

* 1. Accuracy: How often is the classifier correct?
     1. (TN+TP)/N = (300+200)/555 = 0.9
  2. Misclassification (Error) Rate: How often is wrong?
     1. (FP+FN)/N = (5+50)/555 = 0.1
  3. Type I error (False Positive) = 1 - Specificity (TNR)
     1. P(Predict\_N|Actual\_Y) = 50/250 = 0.2
  4. Type II error (False Negative) = 1 - Sensitivity (TPR)
     1. P(Predict\_Y|Actual\_N) = 5/305 = 0.02
  5. (Profit from fraud = Type II error 抓到fraud save的钱 – type I error导致不好的影响（冤枉好人并且降低信誉的损失）
  6. ROC curve (Receiver Operator Characteristic)
     1. Diagonal line 🡪 randomly guesses
     2. X-axis: 1-Specificity; Y-axis: Sensitivity
     3. Higher AUC (area under the curve) is good
  7. Cumulative Accuracy Profile (CAP)
     1. CAP Analysis
        1. Check AR Percentage (see below)
        2. Check the purchased % when Total contacted equal 50%



## QDA (Quadratic Distriminant Analysis)

Concept 🡪 Why?

1. Concept:
   1. Alternative approach for LDA
   2. QDA classifier results from assuming that the observations from each class are drawn from a standard normal distribution (Gaussian distribution)
   3. It assumes that there is a covariance matrix for the kth class (∑k)
      1. An observation from the kth class is of the form X ~ N(μk,∑k)
2. Why covariance matrix?
   1. Bias-variance trade-off
      1. QDA has a higher variance than LDA
   2. QDA is recommended if the training set is very large

## Naïve Bayes

1. These rely on Bayes's theorem, which is an equation describing the relationship of conditional probabilities of statistical quantities.
   * In Bayesian classification, we're interested in finding the probability of a label given some observed features
   * P (L | features) = P (features | L ) \* P(L) / P(features)
2. We are trying to decide between two labels 🡪 compute the ratio of the posterior probabilities for each label:

## KNN (K-nearest neighbors)

Concept (steps) 🡪 Distance metrics 🡪 Standardize

1. Concept:
   1. A non-parametric method used for classification and regression
   2. KNN: K=5
      1. When there is a new data point X0 coming
      2. Identify 5 points (training) closest to X0 (testing)
      3. K grows -> less flexible -> low variance
   3. Regression:
      1. Average of the values
   4. Classification (simplest)
      1. The most common class (voting)
2. Distance metrics 🡪 determine the similarity (nearness)
   1. Euclidean distance – two records (x1, x2,…, xp) and (y1, y2,…, yp)

* 1. Manhattan distance
  2. For the large scale, we can standardize the data
     1. Normalization
     2. Standardization

# Chapter 4 Resampling Methods

Cross-validation 🡪 Step 🡪 Bootstrap

## Cross-Validation

1. The Validation Set Approach
   1. Used to estimate the test/prediction error
      1. Evaluate model’s performance (model assessment)
      2. Select the proper level of flexibility (model selection)
   2. MSE for quantitative response; misclassification rate for qualitative (discrete) response
2. K-fold Cross-validation
   1. K=5
   2. Randomly divide the data into 5 equal-sized parts: leave kth parts, fit the model to the other K-1 parts
   3. Average the 5 MSE:

## Bootstrap

1. Concept:
   1. Bootstrap can be used to estimate the standard errors or confidence interval of the coefficients.
   2. Obtain distinct data sets by repeatedly sampling observations from the original data set with replacement

# Chapter 5 Variable/feature Selection

Why 🡪 forward 🡪 how evaluate 🡪 ridge 🡪 lasso 🡪 PCA

1. Concept:
   1. Also called feature selection/attribute selection
   2. Why consider variable selection:
      * 1. Make sure variables used for model building are the most critical/relevant ones to detect fraud (model interpretability)
        2. Reduce dimensions to avoid collinearity and overfitting (prediction accuracy)
   3. Three ways:
      1. Subset Selection
      2. Shrinkage
      3. Dimension Reduction
2. Other ways for regression (p-value):
   1. Backward Elimination
      1. Select a significance level (0.05)
      2. Fit the full model with all possible predictors
      3. Consider the predictor with the highest P-value, if >0.05 🡪 remove
      4. Fit the model again until all p-value <0.05
   2. Forward Selection
      1. (i. ii. Same as above)
      2. Keep the variable with lowest p-value and
      3. Fit model again with all other one extra variable until the new lowest p-value > 0.05
      4. Done (keep the last variable selections)

## Subset Selection

1. We identify a subset of the p predictors that we believe to be related to the response. We then fit a model using least squares on the reduced set of variables.
2. Best subset selection
   1. Total 2^P-1 possible subsets
   2. Start with Null model: M0
   3. For k = 1, 2, … p:
      1. M1 – Mp <- Model that contains exactly k predictor
   4. Pick the best among each model -> estimating the training error
      1. Smallest RSS
      2. largest R^2
   5. Select the best model among M0 – Mp -> estimating the test error
      1. Indirectly Adjustment:
         1. Mallow’s Cp

* + - 1. AIC

         2. For small sample sizes
      2. BIC
         1. Since log(n) > 2 for any n>7 -> heavier penalty
      3. Adjusted R2
         1. Maximizing the adjust R2 is equivalent to minimizing RSS/(n-d-1)

When the number of variables in the model increases, RSS always decreases, but the RSS/(n-d-1) may increase or decrease, due to the presence of d in the denominator.

* + - 1. We need smaller Cp, AIC, BIC but larger Adjusted R2
    1. Direct estimate:
       1. Cross-validation prediction error
          1. Advantage:

Doesn’t require an estimate of the error variance σ2

Doesn’t need the model degrees of freedom

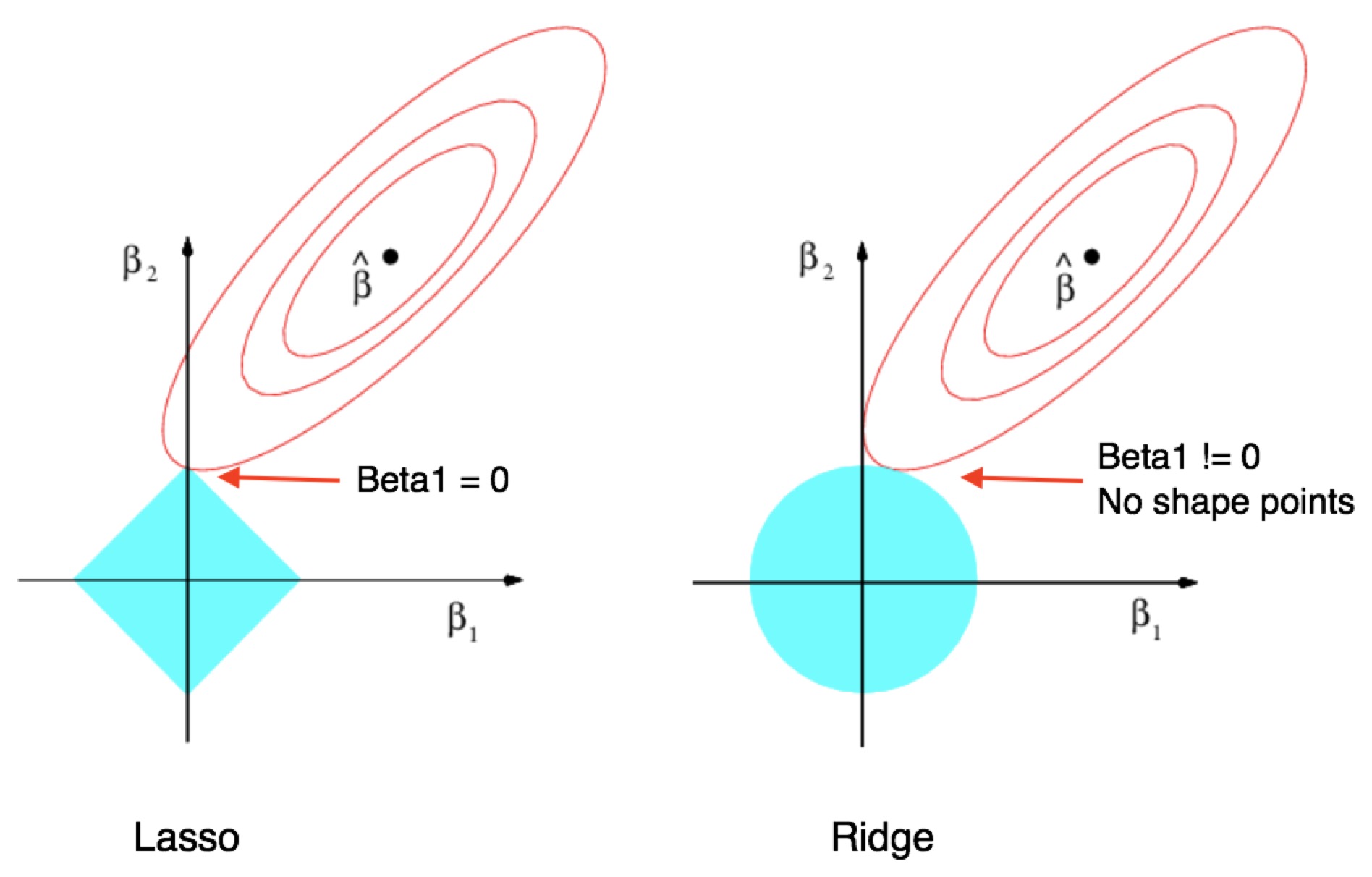
* + - * 1. Better to standardize the data
  1. Drawback: high variance and overfitting

1. Forward/Backward selection
   1. Forward Selection
      1. No predictor (Null model) -> add one predictor -> add … -> all p predictors
         1. Different with best subset -> Must keep the previous predictors
   2. Backward Selection
      1. Start from the full model
      2. Condition: Sample N > Variables P
2. Filter the set of predictors that produce optimal model performance – top 20

## Shrinkage

1. Difference
   1. Subset selection methods: Use least squares to fit a linear model that contains a subset of the predictors
   2. Shrinkage methods: Fit a model involving all p predictors, and constrains or regularizes the coefficient estimates
      1. shrinks the coefficient estimate towards zero -> reduce variance
2. Ridge

   2. Ridge regression seeks coefficient estimates that fit the data well, by making the RSS small
   3. The tuning parameter λ control the relative impact of shrinking the estimates of βj towards zero
      1. λ can be choice by cross-validation
   4. It is best to apply ridge regression after standardizing the predictors
3. Lasso
   1. Shrink the coefficient estimates towards zero
   2. Performs variable selection
   3. In general, lasso perform better when there is only relatively small number of predictors.
4. Different between Lasso and Ridge by chart



* 1. Each cycle stands for different RSS 🡪 The smaller cycle, the smaller RSS
  2. Tangent to constraint region (blue area) 🡪 Smallest RSS within the budget

## Dimension Reduction

1. We project the p predictors into a M-dimensional subspace, where M<p. Then these M projections are used as predictors to fit a linear regression model by least squares.
2. PCA (Unsupervised learning)
   1. An approach for deriving a low-dimensional set of features from a large set of variables
   2. First PC (Principal Component)
      1. (normalized) linear combination of the variables with the largest variance
      2. Chosen to minimize the sum of the squared perpendicular distances to each point (closed)
   3. Second PC
      1. Has largest variance, subject to being uncorrelated with the first
3. PCR (Principal Components Regression)
   1. PCR identifies linear combinations, or directions, that best represent the predictors X1,…Xp
   2. Use first PC as x-axis and second PC as y-axis
      1. -> draw a regression line as PCR
   3. Similar to Ridge
   4. Drawback:
      1. Unsupervised
4. PLS (Partial Least Squares)
   1. A supervised way
   2. Use the response Y in order to identify new features that not only approximate the old features well, but also that are related to the response
      1. Attempts to find directions that help explain both the response and the predictors

# Chapter 6 Non-Linear Models

* Nonlinear Regression 🡪 models that can’t be fit using least squares

## Polynomial Regression

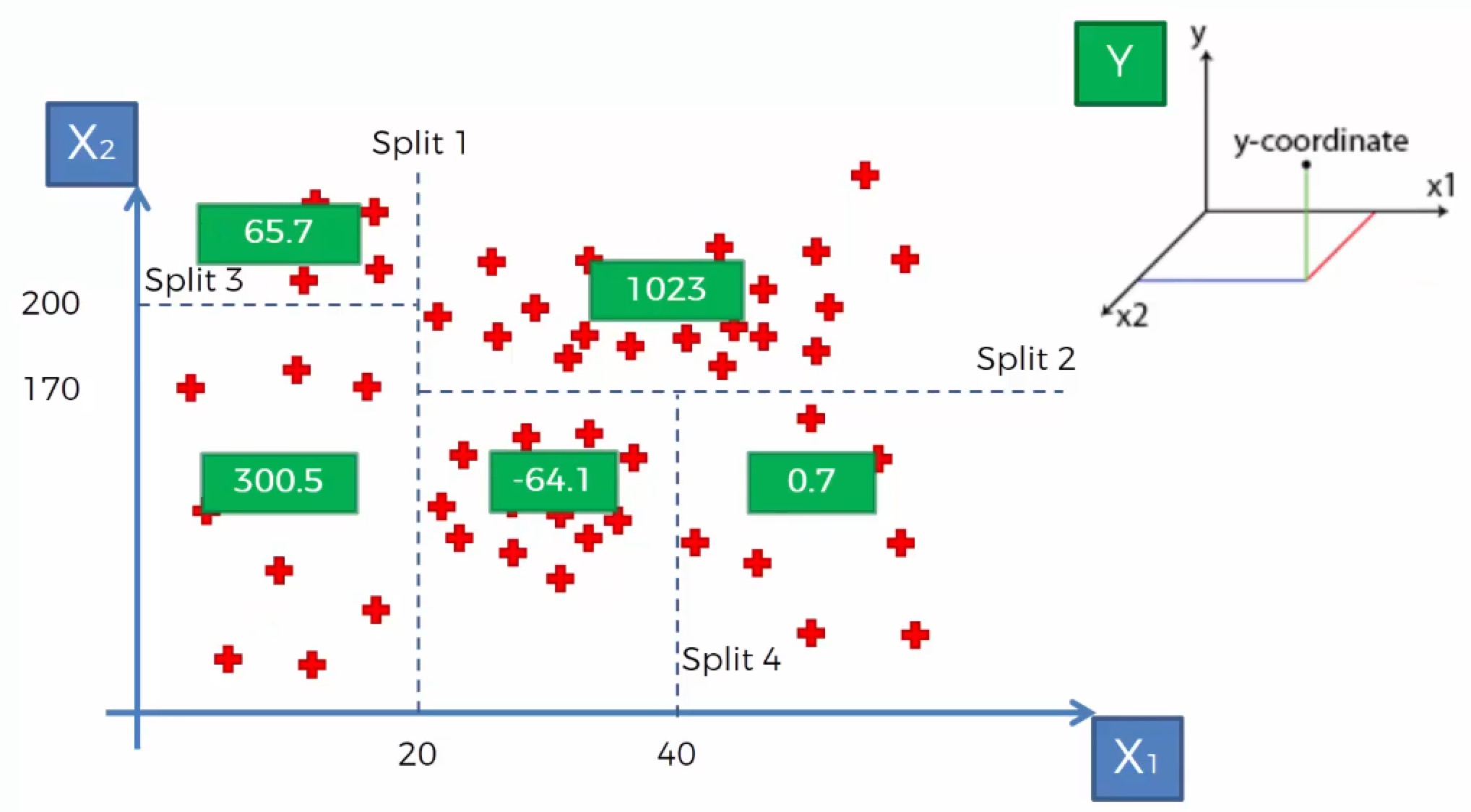
* 1. Extends the linear model by adding extra predictors, obtained by raising each of the original predictors to a power.
     1. Still a linear regression
  2. Including polynomial terms to a regression equation
     1. Can use cross-validation to choose d

1. Step Functions
   1. Cut the range of a variable into K distinct regions in order to produce a qualitative variable.
   2. For example: creates a series of dummy variables representing each group
      1. I(Year < 2005) \* Age, I(Year >= 2005) \* Age
      2. K (Knots)
         1. The points where the coefficients changes
         2. Choose by cross-validation
2. Regression splines
   1. Combine with polynomials and step functions
   2. Fitting separate low-degree polynomials over different regions of X
   3. Smooth at the knots (region boundaries)
      1. The polynomial pieces are smoothly connected at a series of fixed points in a predictor variable (knot)
      2. Coefficients are not interpretable
   4. Cubic spline (4+K degrees of freedom); Linear Splines
   5. Natural spline:
      1. Add more constraints with additional boundary
3. Smoothing splines
   1. Similar to Regression splines
      1. First Term: Encourages g(x) to fits the data well, minimizing an RSS
      2. Second Term: A penalty term that penalizes the variability in g
         1. First derivative g’(t) –> measures the slope of a function at t
         2. Second derivative g’’(t) -> Corresponds to the amount by which the slope is changing (roughness)
         3. The lamda can avoid the knot-selection issue
4. Local regression
   1. Choosing “s” fractions
      1. Choice by CV
   2. Calculate each point in the neighborhood
      1. Further point ~ weight zero
      2. Closer point ~ higher weight point
   3. The value is given by fitting a weighted linear regression
5. GAM (Generalized additive models)
   1. Allow to use (smooth) non-linear functions to replace each variable (“Additivity”)
      1. Automatically specifying the knots in a splines

# Chapter 7 Decision Trees

Plot 🡪 Regression 🡪 Classification 🡪 Advantage 🡪 Bagging 🡪 Random Forest 🡪 Boosting

1. Terminology:
   1. Terminal Nodes (leaves) -> Bottom
      1. Predict value -> average of each terminal nodes
   2. Internal Nodes -> The points along the tree where the predictor space is split



1. Tree-building process: (Regression)
   1. Recursive binary splitting
      1. Top-down
      2. Greedy -> best split is made at that particular step, rather than looking ahead
   2. Details: select the predictors Xj and the cutpoint s such that splitting the predictor space into the regions {X|Xj < s} and {X|Xj >= s} leads to the greatest possible reduction in RSS
   3. Strategy: Pruning a tree (reduce overfitting)
      1. Grow a large tree (until terminal node meets some threshold)
      2. Prune it back to obtain a subtree
         1. Decided by Alpha
            1. Penalty(tuning) parameter for the number of terminal nodes of the tree
            2. Choice by CV
2. Classification trees:
   1. Predict that each observation belongs to the most commonly occurring class
   2. Use classification error rate instead of RSS
   3. A better measure: Gini Index
      1. Total variance across the K classes
      2. G is small if all of the Pmk are close to zero or one
      3. Refer as a measure of node purity
         1. A small value indicates that a node contains predominantly observations from a single class
         2. (If two terminal nodes are same, it means one node is purer than the other)
3. Advantage & Disadvantage:
   1. A: Easy to explain, good for nonlinear
   2. A: Closer with human decision-making
   3. A: Trees can be displayed graphically
   4. D: May has worse predictive accuracy
   5. D: Small change can cause a large disaster
   6. D: Overfitting can easily occur

## No1: Bagging

1. Goal: Averaging a set of observations reduces variance
2. Process: Bootstrap
   1. Taking repeated samples from the training data set
   2. Build the decision tree associated to those data points
      1. Regression: Average all the predictions
      2. Classification: Use majority vote

## No2: Random Forests

1. Improved for bagging
   1. Decorrelates the trees
   2. Reduces the variance
2. A random selection of m predictors is chosen as split candidates from the full set of p predictors (with replacement)
   1. From 16 predictors in training, we only choose 4 of them to considered at each split
      1. so that other predictors will have more chance

## No3: Boosting

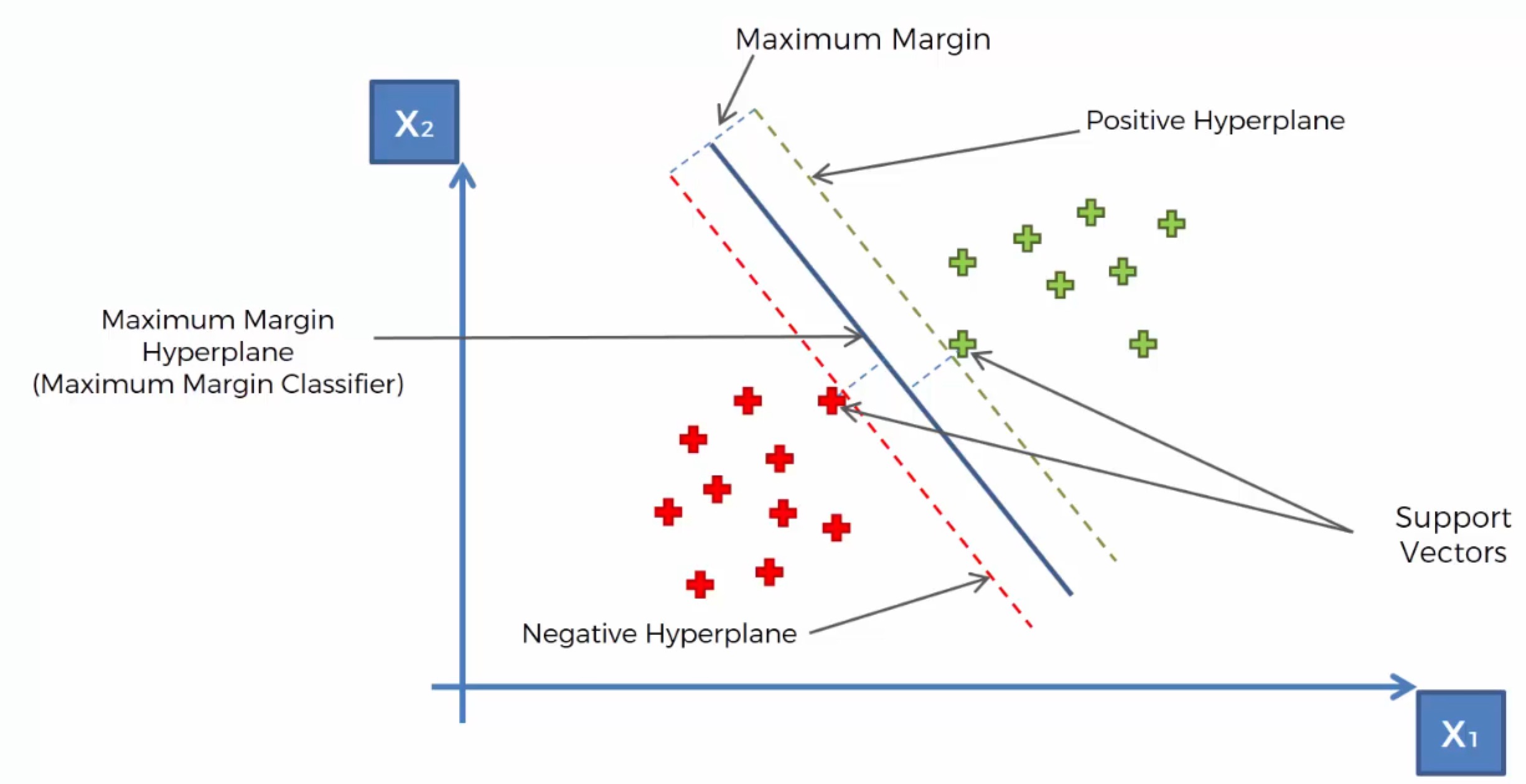
1. Each tree is grown using information from previously grown trees.
   1. Given the current model, we fit a decision tree to the residuals from the model. We than add this new decision tree into the fitted function in order to update the residuals
   2. Grown sequentially and slowly -> reduce overfitting
2. Tuning parameters for boosting
   1. Number of trees -> CV
      1. Too large trees can cause overfit
   2. Shrinkage parameter 入
      1. Control the rate at which boosting learns
      2. Typical values = 0.01/0.001
   3. Number of splits
      1. Controls the complexity of the boosted ensemble
      2. d is the interaction depth, controls the interaction order of the boosted model
      3. Typical value: d=1
3. XGBoost
   1. Most widely used

# Chapter 8 Support Vector Machines

Concept 🡪 Support Vector Classifier 🡪 nonlinear (Kernel) 🡪 Feature of SVM 🡪 Why SVM?

## Support Vector Machines (SVM) – Regression and classification

1. Concept:
   1. SVM – A data classification method that separates data using hyperplanes
   2. Find the hyperplane that makes the biggest gap or margin between two classes
      1. Create a best boundary between two classes so that we will know where the new points will fall



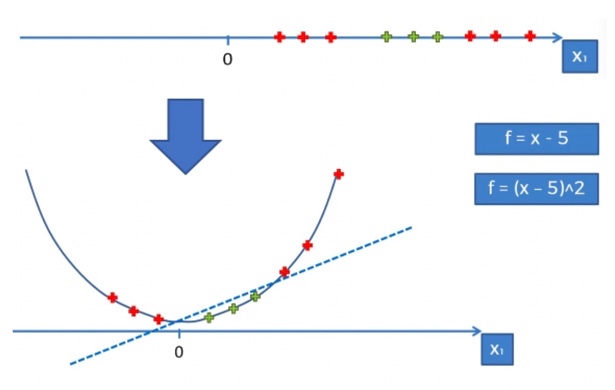
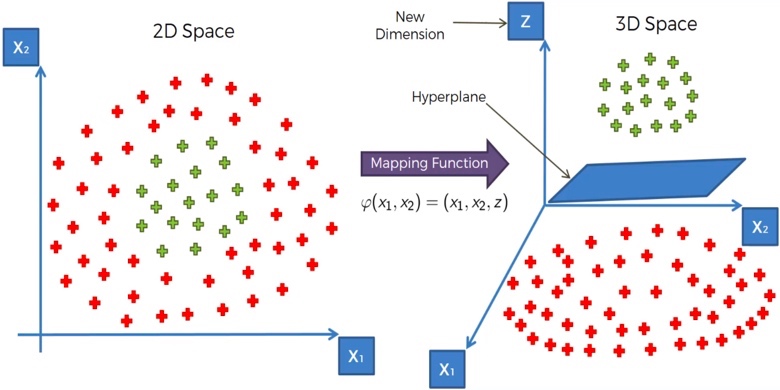
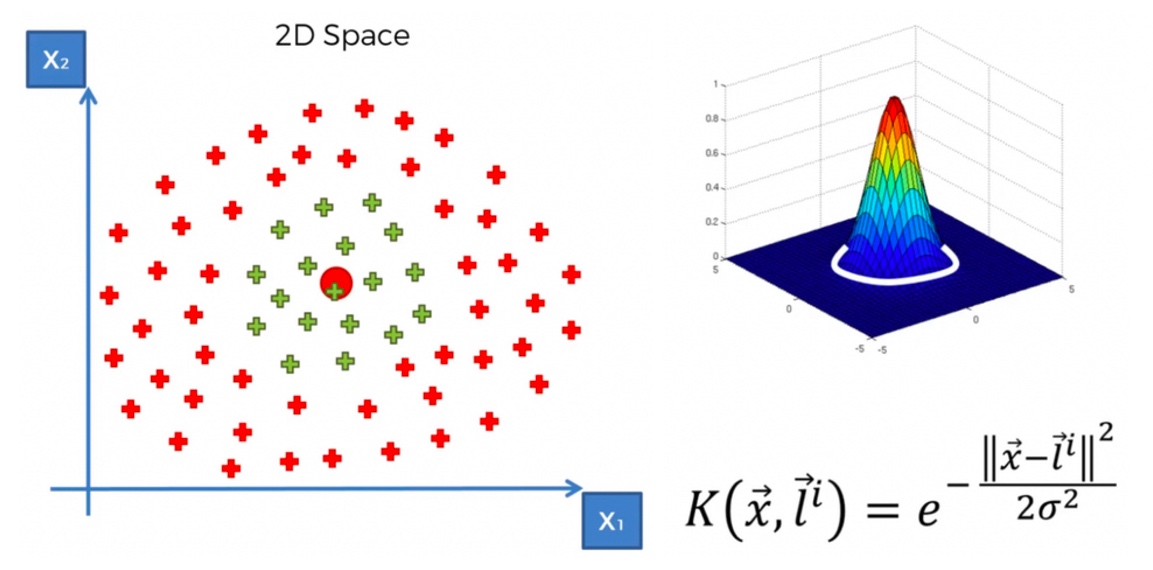
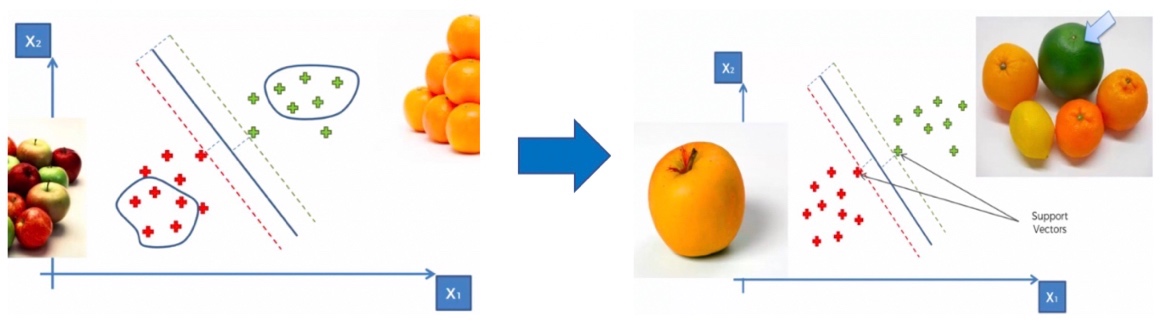
1. Maximal Margin Classifier
   1. Which hyperplanes we need to choose - optimization problem
      1. Maximize M (perpendicular distance from observation to hyperplane)
         1. Beta is a unit vector

* + - * 1. Class One: yi = 1, f(x\*) > 0
        2. Class Two: yi = -1, f(x\*) < 0
        3. => yi \* f(x\*) > 0 (always)

1. Support Vector Classifier
   1. Allow some observations to be on the incorrect side of the margin/hyperplane
      1. Maximize M
         1. C is the sum of
            1. A budget for the amount that the margin can be violated
            2. Choice by CV
            3. When C increases

Lower variance

Wider margin

* + - * 1. If slack variable = 0 🡪 correct side
        2. If slack variable > 0 🡪 wrong side of the margin
        3. If slack variable > 1 🡪 wrong side of the hyperplane
  1. Feature Expansion - nonlinear
     1. Support vector machine
        1. Use instead of just (X1, X2)
        2. Nonlinear decision boundaries
        3. Inner products of the observations
     2. Kernels
        1. A function that quantifies the similarity of two observations
        2. Kernel SVM is good for nonlinear
        3. We can use many different SVM functions
     3. After adding a mapping function and transfer to a higher dimensional space, the dataset is available to classify.
     4. Example: The Gaussian RBF Kernel
        + 1. l 🡪 landmark
          2. sigma 🡪 decide how wide the circumference
        1. The closer to the center, K close to 1. We can define the class by whether the K is bigger than 0 or not.
  2. What’s so special about SVM?
     1. Instead of looking at the most common points in each type of class, SVM is looking at the support vector (close to the boundary 🡪 the point that has the most similar features with the point in another class)
     2. For instance, SVM looks at the apple that most similar to an orange
  3. Why use SVM?
     1. When classes are nearly separable, better than LR
        1. Otherwise -> Similar with LR (with ridge penalty)
     2. LR is prefer when with to estimate probabilities
     3. LR is trying to minimize the error between the prediction and data. The goal for SVR is to make sure that errors do not exceed the threshold

# Chapter 9 Unsupervised Learning

Unsupervised Learning 🡪 PCA 🡪 K-means clustering 🡪 Hierarchal clustering

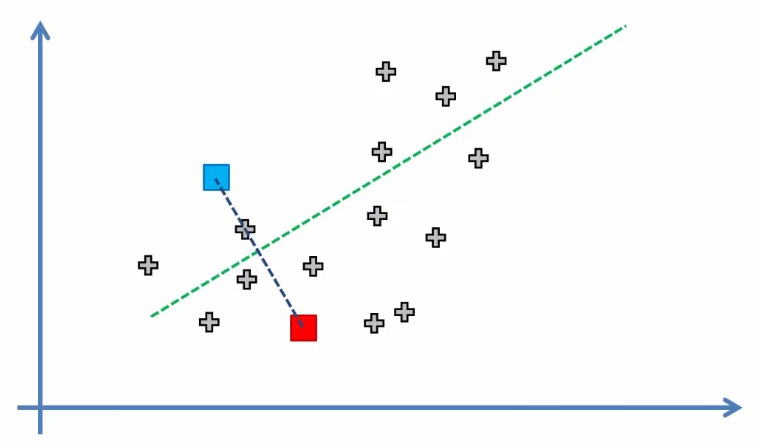
1. Unsupervised Learning
   1. Don’t have an associated response variable Y
   2. PCA
      1. Looks for a low-dimensional representation of the observations that explains a good fraction of the variance
      2. For the nonlinear problem, we use kernel PCA
   3. Clustering
      1. Looks for homogeneous subgroups among the observations

## PCA (Principal Components Analysis)

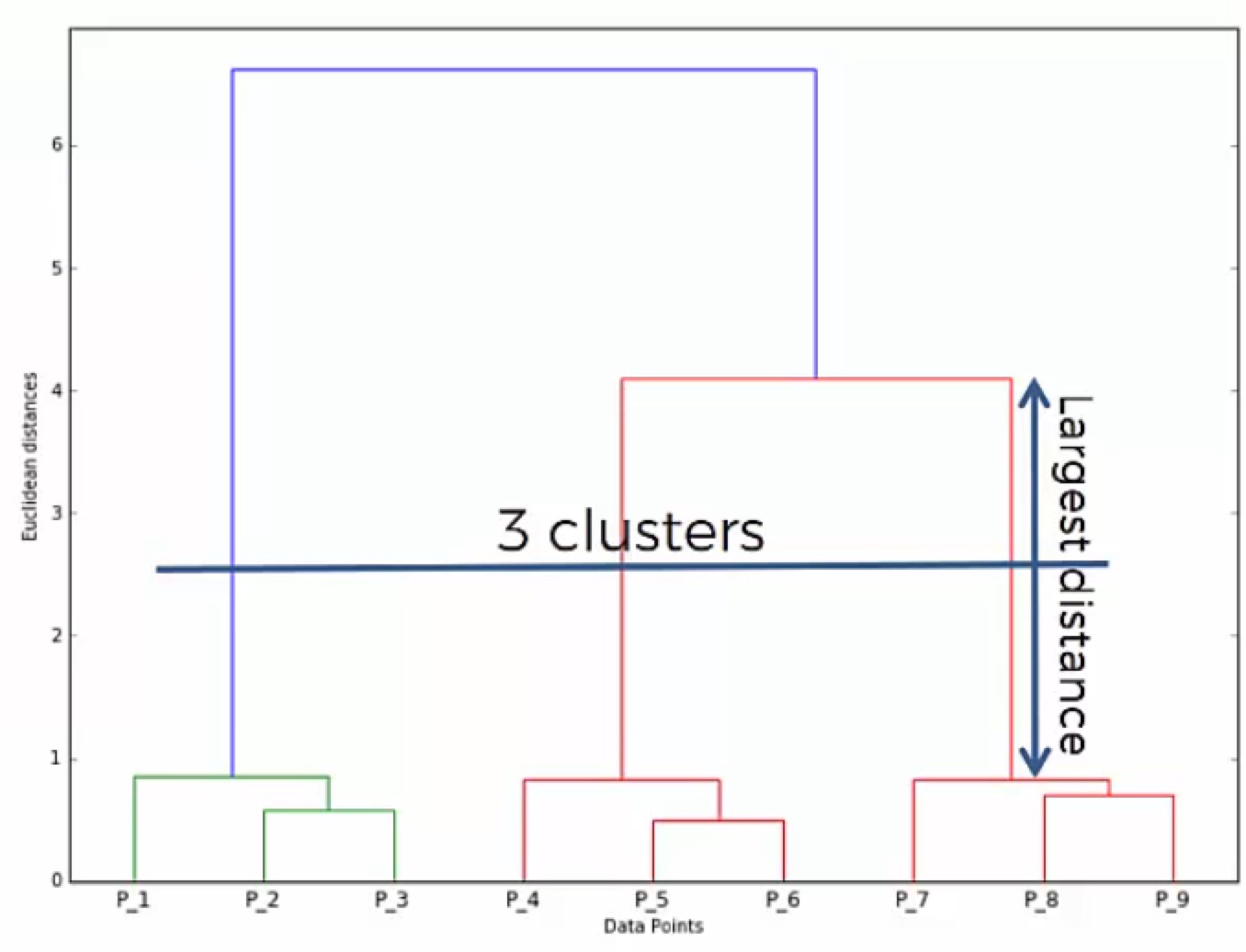
1. Concept:
   1. Used for data visualization or data pre-processing
   2. Summarize a large set of correlated variables with a smaller number of representative variables that explain most of the variability in the original set
      1. Reduce the dimensionality if there is a strong correlation.
      2. Find the directions of maximum variance in a high dimensional data and then project it into a smaller dimensional subspace while retaining most of the information
2. First principal component
   1. A set of features X1, X2, …, Xp is the normalized linear combination of the features below that has the largest variance
      1. Since we only interested in variance -> we assume the mean equal to zero
      2. Variance =
      3. Z => scores of the first principal component
      4. Subject to
         1. Make sure the variance z is not too big
3. Second principal component:
   1. The linear combination of X1, X2, …, Xp that has maximal variance among all linear combinations that are uncorrelated with Z1
4. Biplot
   1. PCA1 and PCA2 can tell a different information about the data
5. Another Interpretation
   1. The first PC defines the line in p-dimensional space that is closest to the n observations
      1. Different with SVM hyperplane because it is unsupervised without y response
      2. Need to scale each variable in the same units
6. Proportion variance explained (PVE)
   1. Strength of each component
   2. Goes down
   3. Use smallest number of PC required to get a good understanding of the data

## Clustering <- Discovering unknown subgroups in data

1. Concept:
   1. Seek a partition of the data into distinct groups so that the observations within each group are quite similar to each other
   2. Two clustering methods
      1. K-means clustering
         1. Cons: Need to choose the number of clusters
      2. Hierarchical clustering
         1. Cons: Not appropriate for large datasets
2. K-means clustering
   1. Seek to partition the observations into a pre-specified number of clusters
   2. Properties:
      1. Each observation belongs to at least one of the K clusters
      2. The clusters are non-overlapping: no observation belongs to more than one cluster
   3. Goal: get WCV (within-cluster variation) as small as possible
      1. The observations within a cluster differ from each other for each K cluster
      2. Sum them all
   4. Process
      1. Randomly assign a number K=3
         1. For each of the K clusters -> compute the cluster centroid (mean)
         2. Assign each observation to the cluster whose centroid is closest
            1. By using Euclidean distance
         3. Compute and place the new centroid of each cluster
         4. Repeat
      2. The clustering will continually improve until the result no longer changes -> A local optimum has been reached
   5. How many clusters should we choose?
      1. Within Cluster Sum of Square (WCSS)
      2. In the following case, we will choose 3 clusters



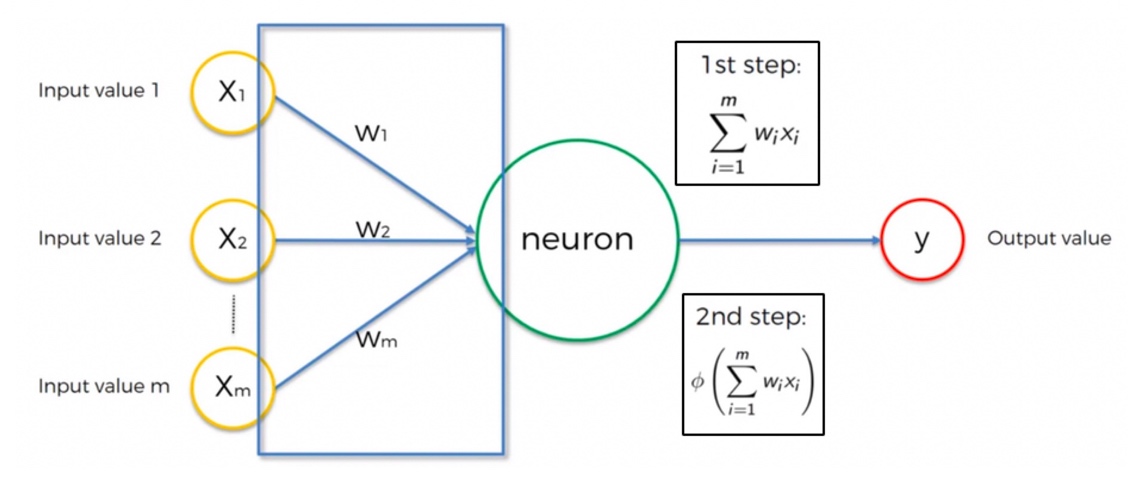
1. Hierarchical clustering
   1. We don’t know in advance how many clusters we want. In fact, we end up with a tree-like visual representation of the observations, called a dendrogram.
      1. No need K
      2. Bottom-up
   2. Process
      1. Start with each point in its own cluster
      2. Identify the closest two clusters and merge them
         1. Defined by linkage:
            1. Complete – longest distance between two observation
            2. Single – shortest distance between two observation
            3. Average – average distances between each two observation
            4. Centroid – distance between two centroids
         2. Two dissimilarity measure
            1. Euclidean distance: Shoppers who have bought few items overall will be clustered together (similar value)
            2. Correlation-based distance: Shoppers with similar preferences will be clustered together
      3. Repeat
      4. Ends when all points are in a single cluster
      5. Create dendrogram (like a tree)
         1. Vertical axis -> How different the two observations are
            1. By squared distance between two observations
      6. Cut -> to obtain the cluster
         1. Find the cutline between the largest “line distance” in Dendrogram



# Deep Learning

1. Purpose: Mimic how the human brain works

|  |  |  |
| --- | --- | --- |
| Deep Learning Framework | | |
| Supervised | Artificial Neural Network | Regression & Classification |
| Convolutional Neural Network | Computer Vision |
| Recurrent Neural Network | Time Series Analysis |
| Unsupervised | Self-Organizing Maps | Feature Detection |
| Deep Boltzmann Machines | Recommendation Systems |
| AutoEncoders | Recommendation Systems |

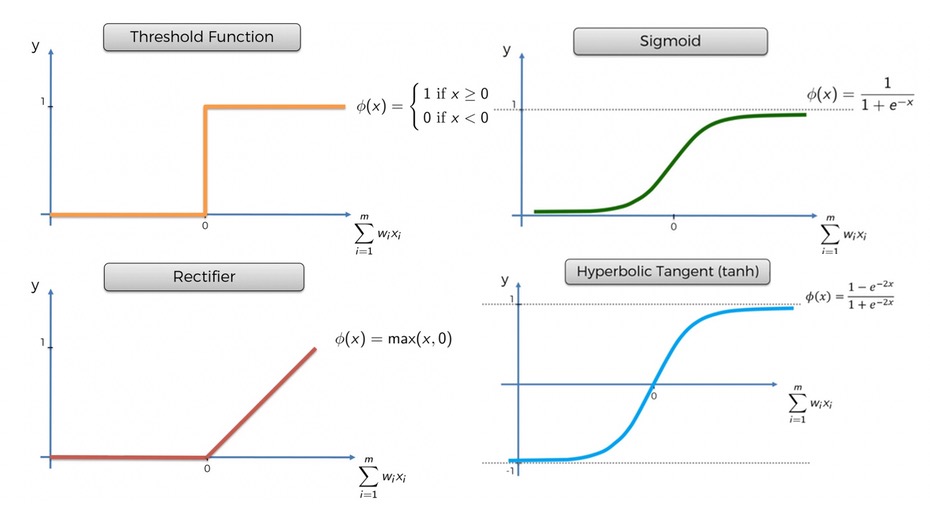
1. Each x stands for different features. X1~Xm 🡪 all for one single row of data
   * Each input value is independent
   * Standardize the values

## artificial Neural Network

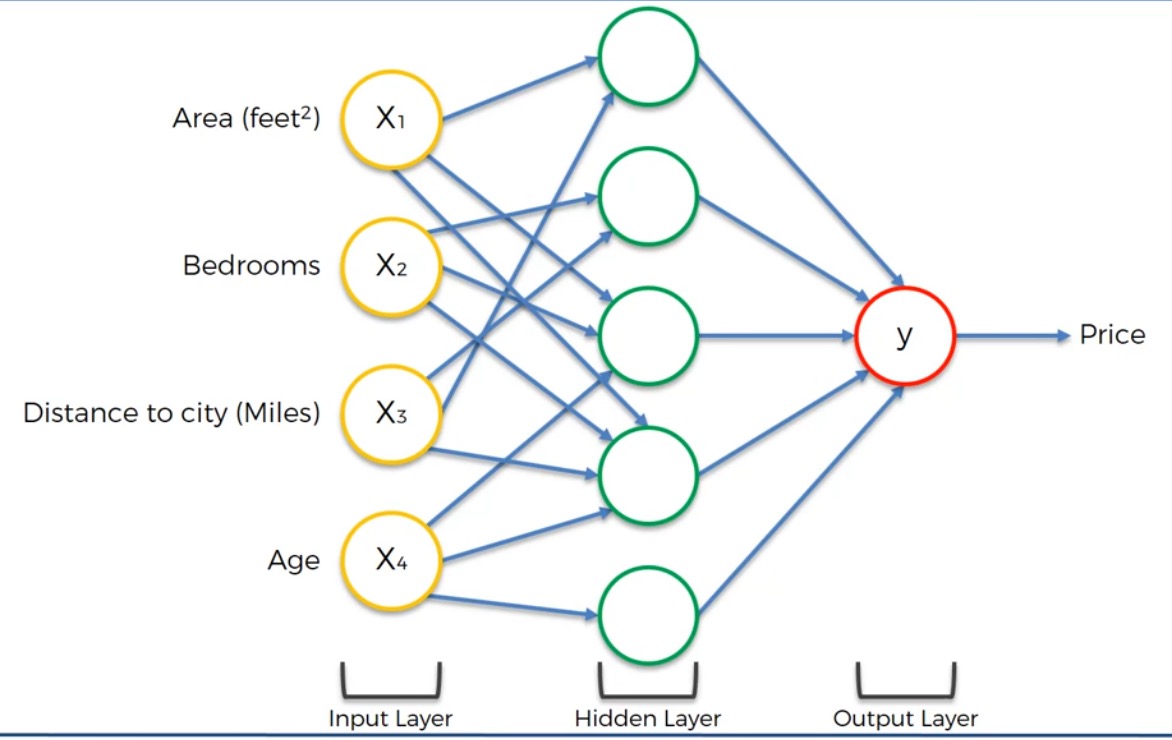
1. Definition
   1. Neural network takes data through layers of interconnected nodes and perform a series of linear and nonlinear transformations to produce an output.
2. Structure and property
   1. Input (x) 🡪 aggregated signal 🡪 activation function 🡪 output with transfer function
      1. With different weight
         1. WiXi = W1X1+W2X2 …
         2. For every single case, the weight can decide what single signal is poor and what is important
   2. Each hidden layer will focus on different specific features (by using different (weight of) inputs) and then in combination
      1. Increase the flexibility of NN
      2. Example of features
         1. Predicted whether the output is a dog, we want to see its nose, legs, eyes…
   3. Single layer feedforward neural (perception) – basic NN with one-layer NN

## Activation function σ(WiXi + b)

1. b is the bias 🡪 make the output non-zero or scale up the system response
2. We will apply to both hidden layer and output layer
3. Examples:
   1. When the dependent variables are binary
      1. Threshold Function
         1. Y equal exactly 1 or 0
      2. Sigmoid activation function (logistic)
         1. Range: (0, 1)
         2. The probability of Y being yes/1 or no
   2. Softmax activation function
      1. Extension of the sigmoid function
      2. Use for multiple classification output activation (p>2)
      3. Its outputs p probabilities for the k classes 🡪 and make their sum equal to 1
   3. Rectifier function (ReLU)
      1. Better to get the gradient descent 🡪 stable increase trend & still has a higher derivative when x gets larger
      2. Used for hidden activation
   4. Hyperbolic Tangent (tanh)
      1. Range: (-1 to 1)

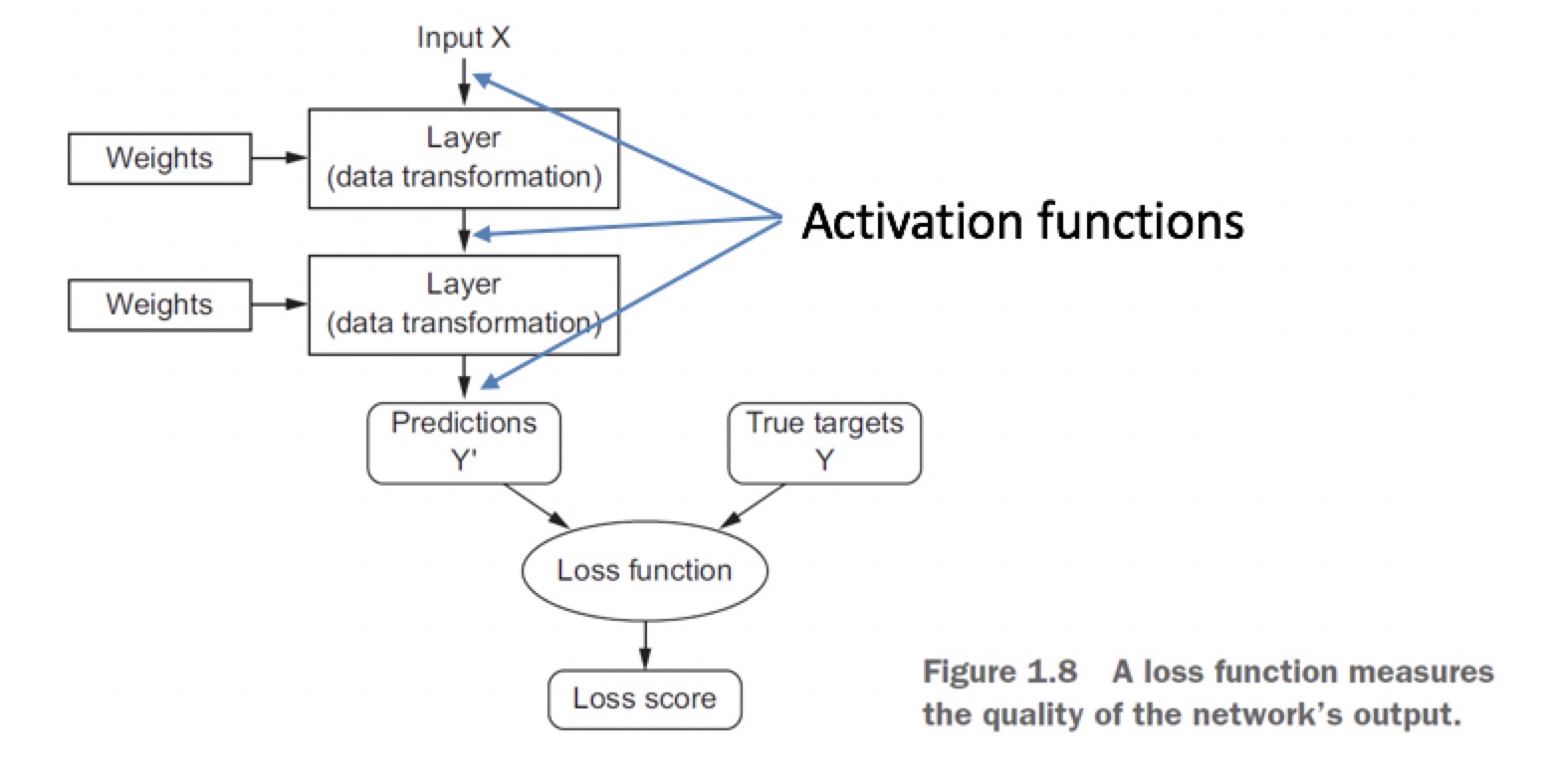


1. How do neural network Works?
   1. Each hidden layer combines different specifically features
      1. With different activation function
         1. Ex: Age (5th hidden neural) can use the Rectifier
   2. Increase the flexibility

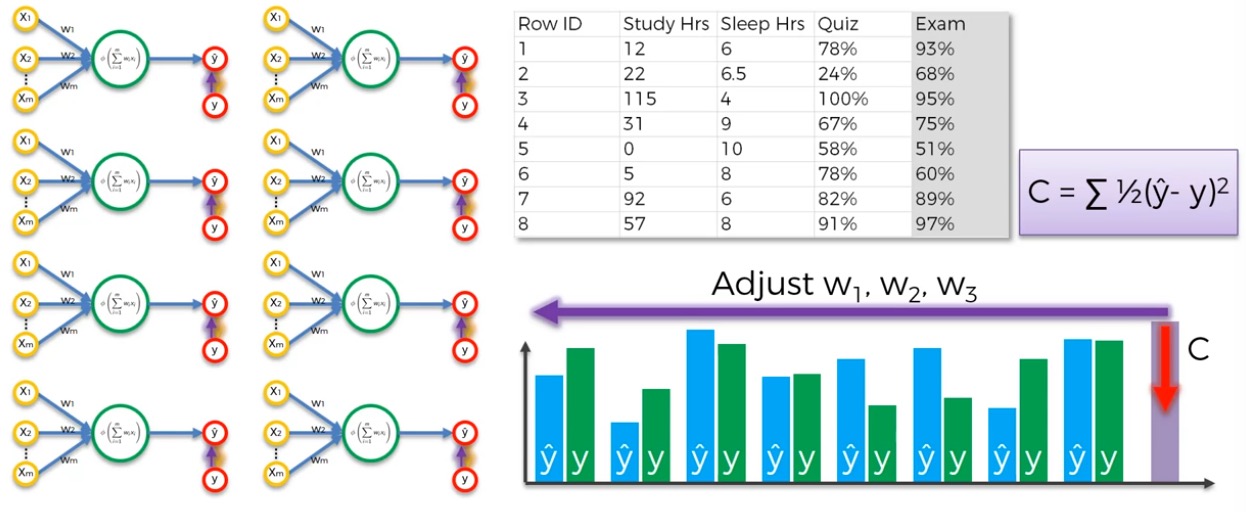


* 1. Logical functions in activation
     1. AND/OR/NAND/XOR
        1. NAND
           1. Equal to 0 only when both of A and B are 1
        2. XOR
           1. Equal to 0 only when A equal to B
     2. For NAND and XOR, we have to use more than one layer. Because the data points can’t separate using one layer.
        1. 🡪 we need to use a multilayer perceptron (MLP) by adding a hidden layer

1. Anatomy of Neural Network
   1. The goal is to compare the predicted Y with the target Y and choose weights and biases to minimize the loss function



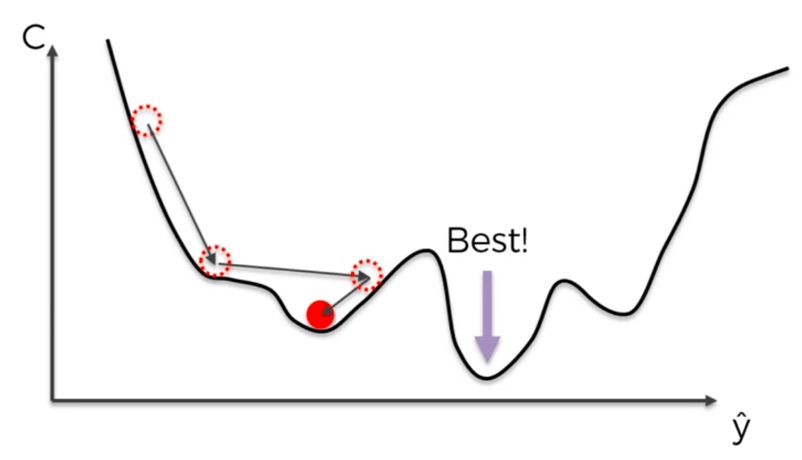
1. Cost Function (loss function)
   1. Calculate the output from row1 for cost function 🡪 get the feedback (sum of the Y difference) and adjust the weight 🡪 repeat for row2, row3, …
      1. The whole process called back propagation



* 1. Squared error loss (L2 loss)

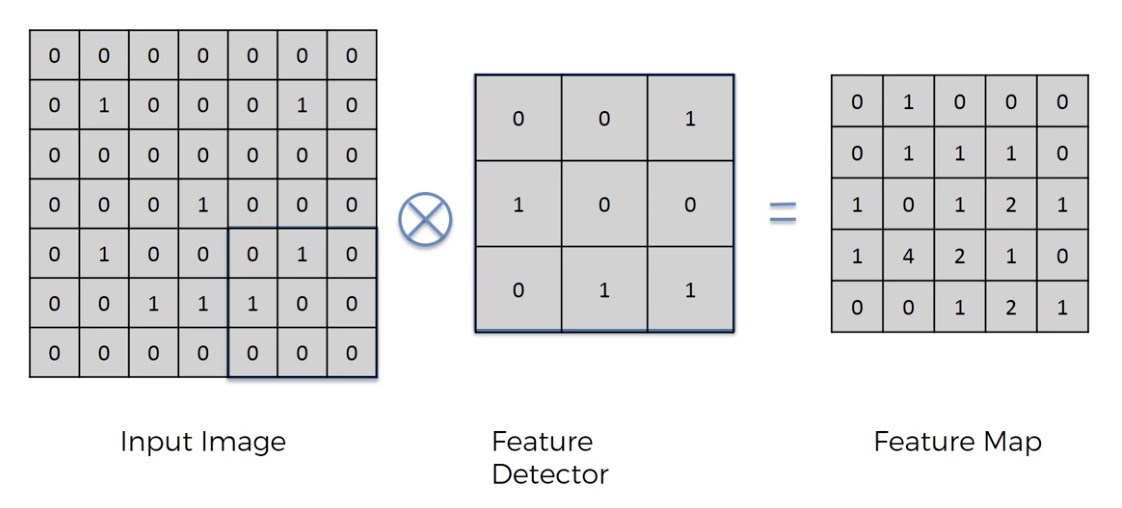
* + 1. For linear regression
  1. Absolute error loss (L1 loss)

     2. Better for outliers
  2. Logistic loss
     1. When the Y is either positive or negative
  3. Cross entropy
     1. When the Y is 0/1 (cat or dog)
     2. Useful for CNN
     3. At beginning, we have a tiny output value. The MSE is not very efficiency to evaluate the model. But using the Cross-entropy which defined with log, we can have a bigger power to evaluate the model.
     4. Also used as the cost function for logistic
  4. Hinge loss
     1. Same case as above, popular for SVM

1. Gradient descent algorithm
   1. To answer how to minimize the loss function C(W, b) with respect to W and b
   2. Why not try out a thousand different parameters/inputs to get the minimize the loss function?
      1. Curse of dimensionality
         1. We will get input ^ (number of weights) combinations
   3. Example: cost function
      1. Calculate derivative of f(x)
      2. Derivative =
         1. The 1st component of the gradient is positive when X1>5, negative when X1<5, and zero when X1 = 5
         2. The 2nd component of the gradient …… and zero when X2 = 3
         3. The cost is the lowest when the gradient (derivative f(x)) is equal to zero
         4. => When x = (5, 3)
2. Stochastic gradient descent (SGD)
   1. Adjust the weights after every single row – very computed
   2. For more complicated model, it is hard to compute gradient
   3. Some loss functions are not convex. We will get the local minimum instead of global minimum
   4. Further methods: mini-batch gradient decent
      * 1. Separate the full-batch to mini-batch 🡪 adjust the weights by mini-batch
        2. Avoid finding local minimums
        3. We can also do a batch gradient descent with only one batch
      1. Epoch – Going through each training example exactly once
         1. Ex:
            1. Training examples: 2000
            2. Mini-batch size: 500
            3. It will take 2000/500 = 4 iterations to complete 1 epoch
         2. We need to use more than one epoch 🡪 updating weights more time
      2. SGD: reinforcement learning, BGD: batch learning
3. Forward Prorogation VS Back Prorogation
   1. Randomly initialize the weights to small numbers close to 0 (not 0)
   2. Input the first observation of the dataset in the input layer, each feature in one input node
   3. Forward Prorogation 🡪 from left to right 🡪 the neurons are activated in a way that the impact of each neuron’s activation is limited by the weights 🡪 Propagate the activations until getting the predicted result y
   4. Compare the predicted result to the actual result. Measure the generated error
   5. Back Prorogation 🡪 from right to left 🡪 the error is back-propagated 🡪 update the weights according to how much they are responsible for the error 🡪 the learning rate decides by how much we update the weights
   6. Repeat step a to e
   7. Regularization
      1. Overfitting 🡪 we want to use smaller variables to interpretable model
      2. Add penalty
         1. Lasso and ridge
      3. Dropout
         1. Randomly drop units (both hidden and visible, along with their connections)
      4. Batch normalization (standardization)

## Convolutional neural networks (CNN)

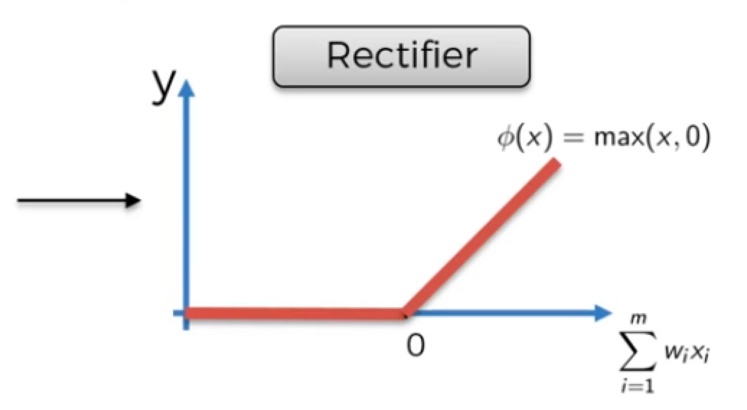
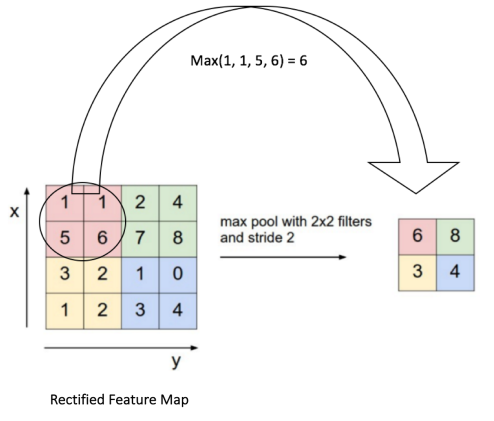
1. Application
   1. Image data and computer vision tasks
   2. Inspired by architecture of the biological neural network in the visual cortex
2. Why not DNN?
   1. DNN can suffer from the curse of dimensionality and may not scale well to higher resolution images
      1. Like weight 256 \* height 256 \* depth 3 = 196, 608 dimensional vectors
   2. CNN can dramatically and efficiently reduce the dimensionality by exploiting the spatial structure of data – through several distinct network features
3. Architecture of CNN
   1. Input Image 🡪 CNN 🡪 Output Label (Image Class)
   2. Image 2\*2 🡪 4 pixels
      1. Black and White 🡪 2d array 🡪 0 ~ 255-pixel value
      2. Colored 🡪 3d array 🡪 red, green, blue 🡪 each one has 0 ~ 255-pixel value
   3. The neurons in a layer are connected to only a small region of the previous layer
4. Step 1 - Convolutional Layer
   1. Steps:
      1. Filter – represented by a weighted (along with a bias)
         1. Also called feature detector
         2. Feature is what we use to identify the input (dog’s ear).
      2. We slide the filter over the input image pixel by pixel
      3. Convolution - For each position, we calculate the sum product between the underlying image patch and the filter
         1. First convolved feature = 1(1) + 1(0) + 1(1) + 0(0) + 1(1) + 1(0) + 0(1) + 0(0) + 1(1) = 4
      4. Repeat the convolution process until the entire image is covered

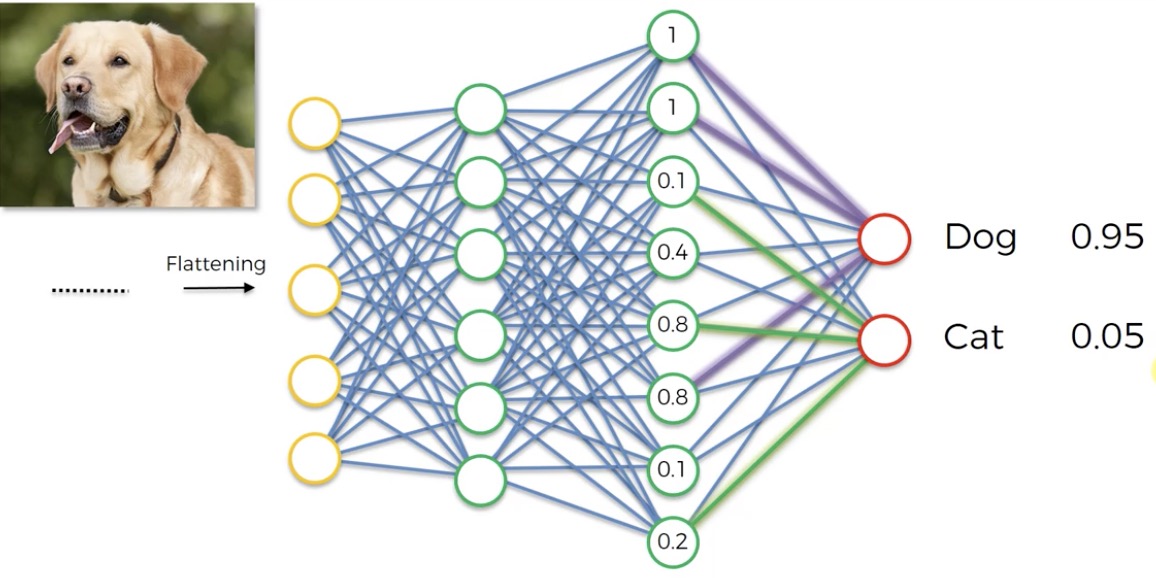


* + 1. Then we obtain the convolved feature matrix (feature map) corresponding to the particular filter
    2. It is common to apply different filters (feature detectors) to extract various/specific features and patterns from the input image
       1. The number of filters used for convolution 🡪 DEPTH of convolution layer



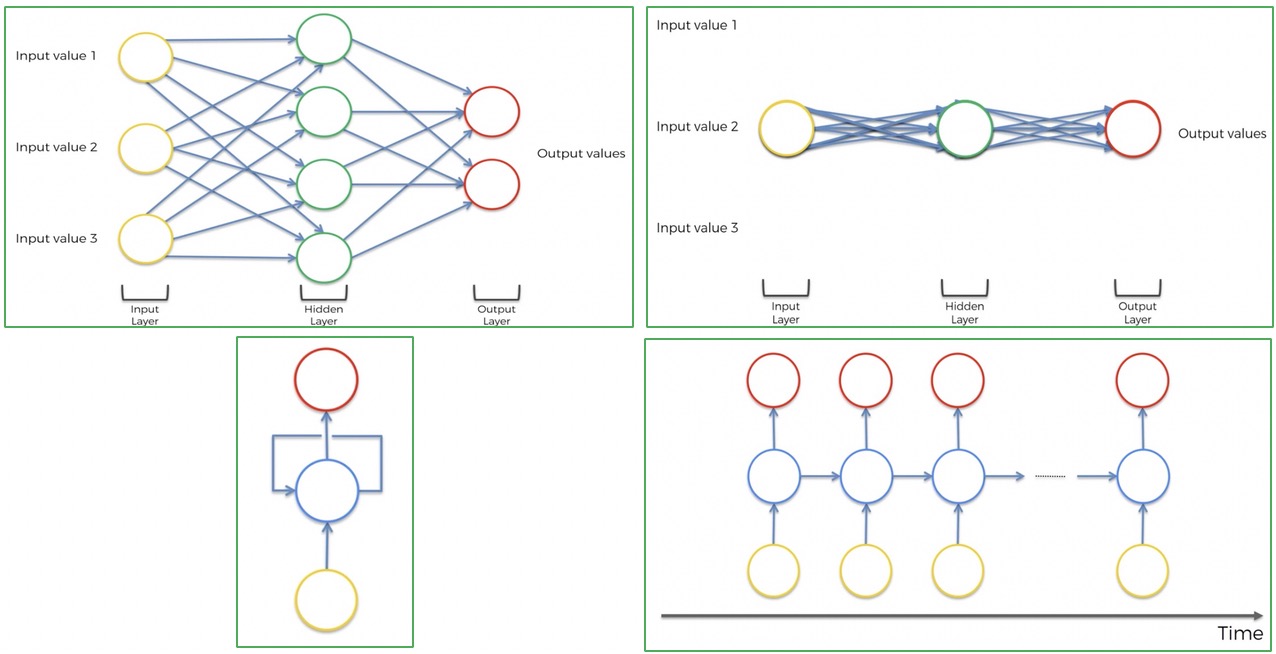
* 1. Zero-padding
     1. Padding the input matrix with zeros around the border
     2. Often used to control the size of the feature maps
     3. If P=2 🡪 add two zeros around the border
  2. Stride
     1. Slide the filter by more than one pixel at a time
     2. A larger stride leads to smaller feature maps
  3. How many connections/weights?
     1. For input size [16\*16\*20] and filter size [3\*3]
     2. We have 3\*3\*20 = 180 connections/weights
  4. Output size
     1. (Volume Size (W) – Filter Size (F) + 2 \* Zero Padding (P))/Stride(S) + 1
     2. For 7\*7 input and a 3\*3 filter with stride 2 and pad 0
     3. Output = (7 – 3 + 2 \* 0) / 2 +1 = 3 🡪 3\*3 output

1. Step 1(b) - ReLU Layer (Activation function)
   1. The ReLU activation function is the rectified linear unit function f(x) = max(0, x)
      1. Why? 🡪 we want to increase non-linearity in our image/network because image itself is very nonlinear (convolutional is linear)
         1. EX: “Black, white, and black” (linear can’t detect the white between two blacks)
   2. Apply to the feature map matrices from the convolution layer entrywise (per pixel)
   3. Output feature map 🡪 “Rectified Feature Map”
2. Step 2 - Pooling Layer
   1. Why use pooling layer?
      1. It helps NN can have ability to still find out the feature no matter the feature’s texture, location, size is different (Or image tilt, twist) 🡪 spatial invariance property for NN
      2. Reducing the number of parameters and size 🡪 prevent overfitting, but also preserve the main features
   2. Pooling is a form of nonlinear downsampling (subsampling)
   3. Reduces the dimensionality of feature map while retaining the most import information (max, mean, sum, …)
   4. The sequence of convolution layer, ReLU layer, and pooling layer can be repeated to form a deep neural net
3. Step 3 – Flattening
   1. Pooled Feature Map 🡪 Flattening 🡪 one column of data (one vector of inputs)
   2. [1 2 3 4 5 6 7 8 9]
4. Step 4 - Fully Connected Layer
   1. Adding a whole artificial neural network to the CNN
      1. Input layer from flattening 🡪 fully connected layer 🡪 output layer
5. Output and Loss Layer
   1. Each category uses different feature to predict
   2. After the fully connected layers, CNN ends with the output and loss layer containing the softmax neuron for multi-class classification tasks (k>2)
      1. Softmax 🡪 bring value to 0~1 and Pdog+Pcat = 1 🡪 Which class has a highest proability
      2. Cross-Entropy 🡪 loss function for softmax

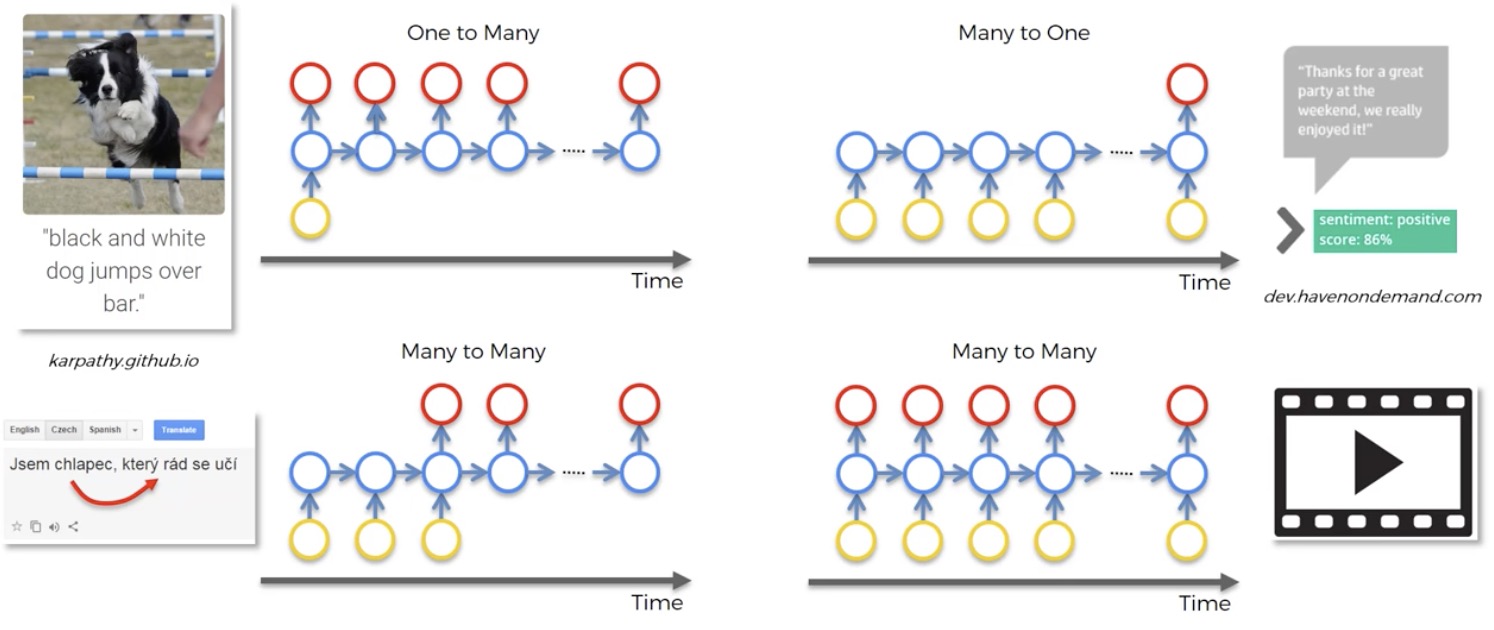


## Recurrent Neural Networks (RNN)

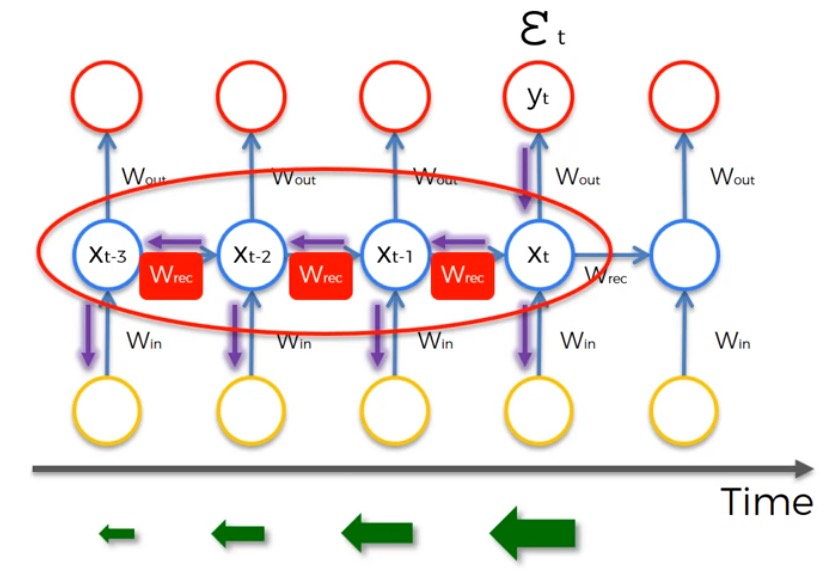
1. Application
   1. Sequence data and machine translation
   2. 1 🡪 Simple artificial neural network
   3. 1-2 🡪 Squash it all together 🡪 add a new dimension
   4. 2-3 🡪 Simplify & twist to vertical 🡪 add a line (the hidden layer not only gives an output but also feeds back into itself)
      1. Every single node here is not just a node but a representation of a whole layer of knowledge
   5. 4 🡪 You get inputs coming into the neurons then you get outputs but also the neurons connecting to themselves through time 🡪 “short term memory”



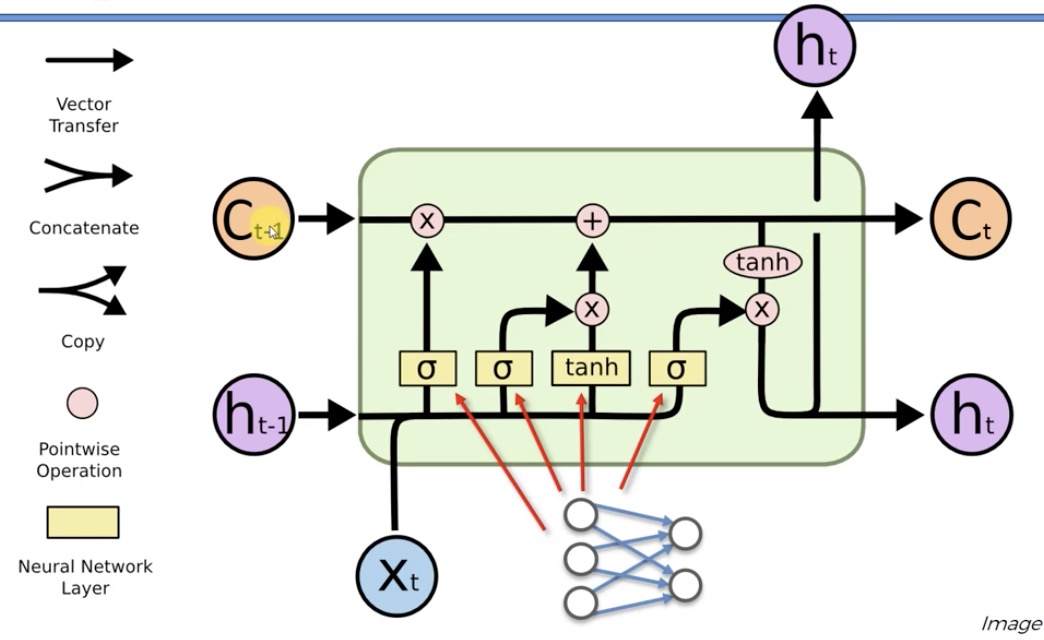
1. Example:



1. Vanishing Gradient
   1. Calculate each cost function 🡪 propagate all the way back through time to these neurons (and update weight recurring)
   2. Vanishing gradient problem
      1. The lower the gradient, the slower/harder update the weights
         1. Wrec ~ small 🡪 Vanishing
         2. Wrec ~ large 🡪 Exploding
   3. Solutions:
      1. Vanishing Gradient
         1. Weight Initialization
         2. Echo State Networks
         3. Long Short-Term Memory Networks (LSTMs)
      2. Exploding Gradient
         1. Truncated Backpropagation
         2. Penalties
         3. Gradient Clipping

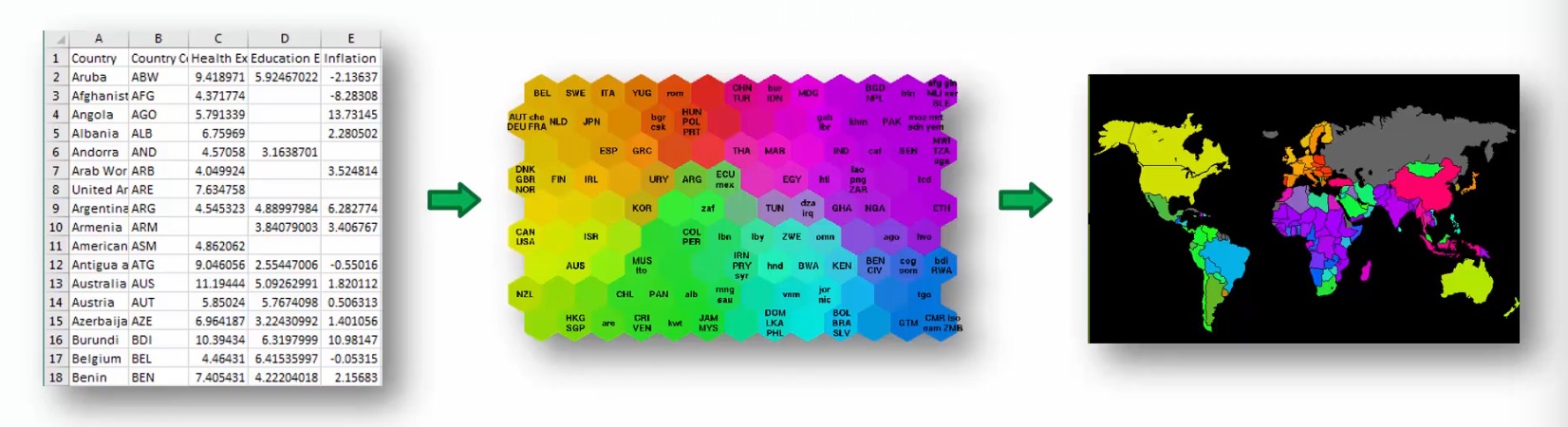


1. LSTMs
   1. Structure
      1. Everything here 🡪 One Vector
      2. C 🡪 Memory
      3. H 🡪 Output
      4. X 🡪 Input
   2. Concatenate 🡪 Two lines are running in parallel to each other (not combine)
   3. Copy 🡪 simple copy into multiple lines
   4. Pointwise Operations
      1. “X” (Valves)
         1. Forget Valve (F) 🡪 Memory Valve (V) 🡪 Output Valve (O)
         2. Controlled by activation function to decide whether memory flows through the vector transfer (closed / open)
            1. Sigmoid activation function / tanh activation function
      2. “+” (T-shaped joint)
         1. Have memory going through and then you can add addition memory as well depending on the valve
   5. Network Network Layer
      1. Layer of sigmoid coming out controlling the valve for each elements in the vector of memory

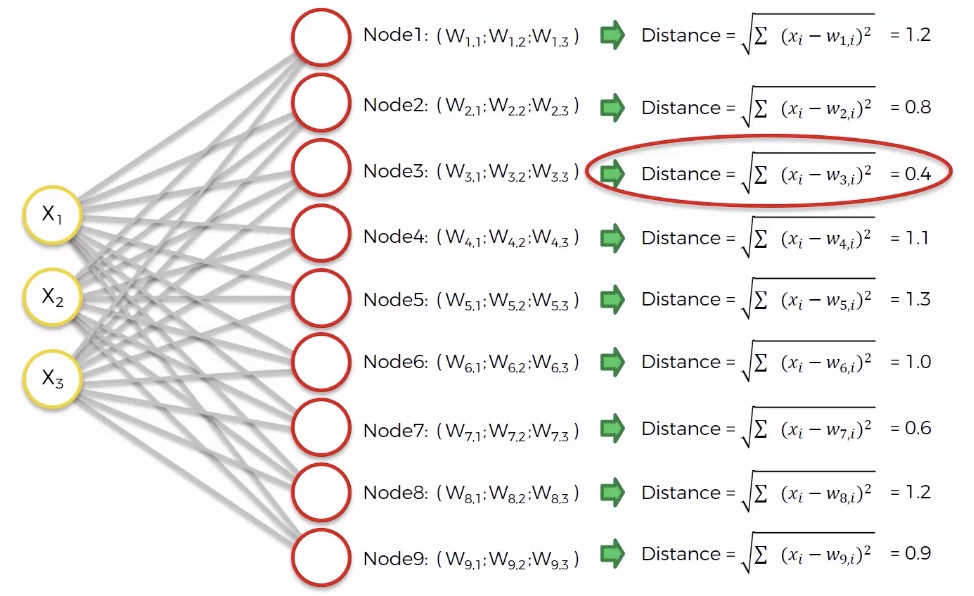


## Self Organizing Maps

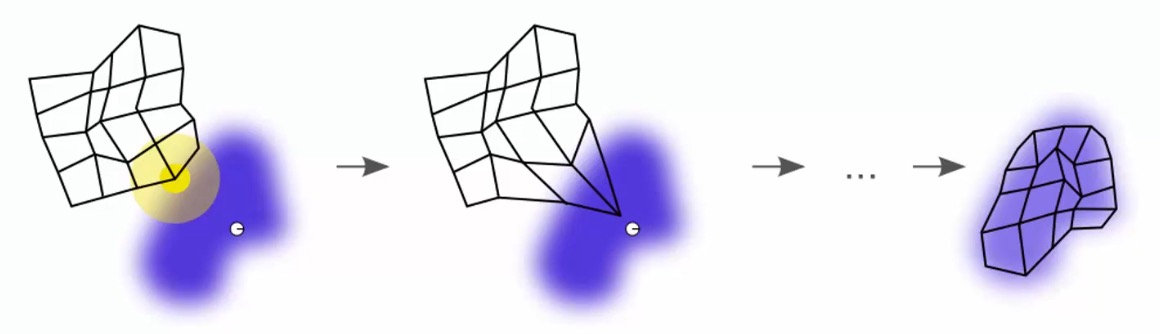
1. How do SOMs Work?



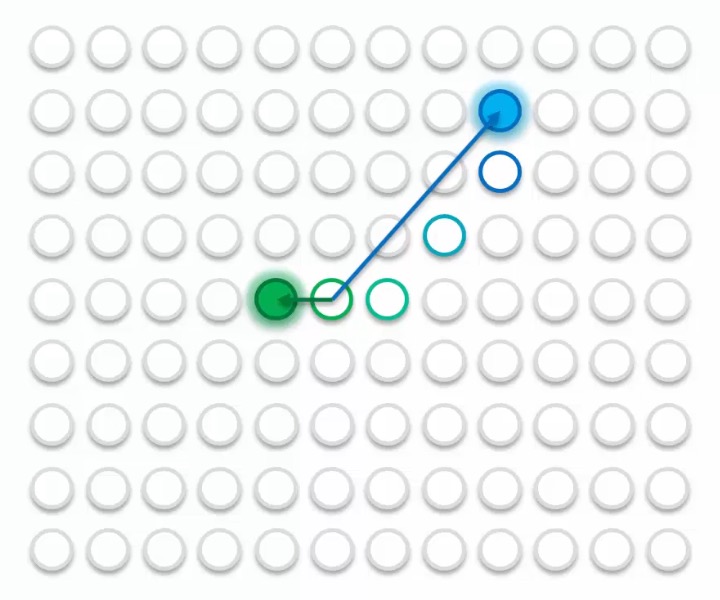
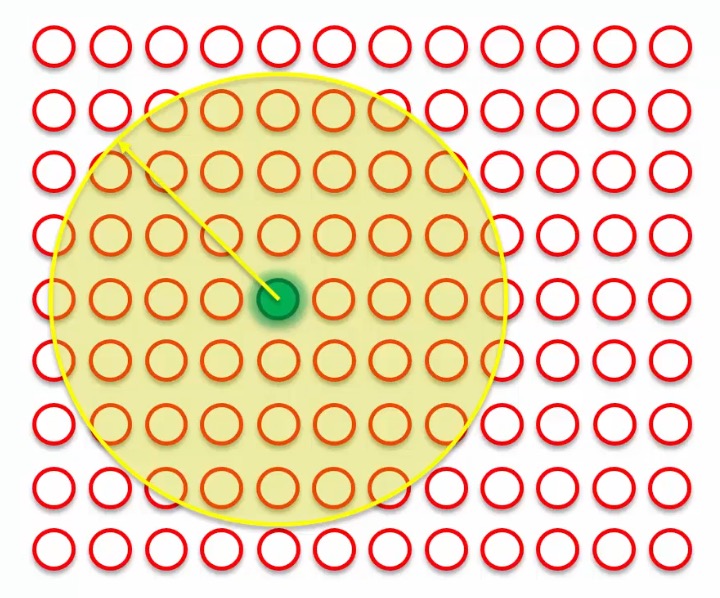
* 1. Step One:
     1. Change the visual representation between the visible input nodes and output nodes
     2. Each node has 3 weight assigned as coordinates in the input space
        1. The weights are different meaning from artificial neural networks
     3. Find the node with the shortest distance 🡪 best matching unit



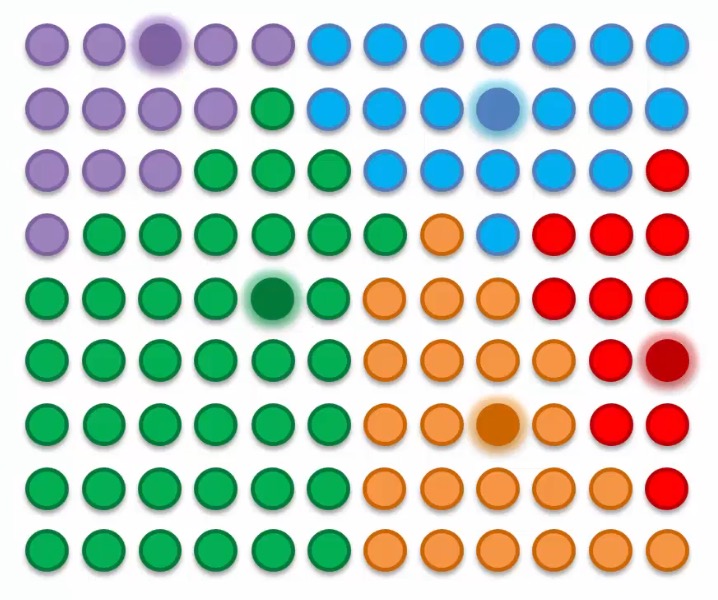
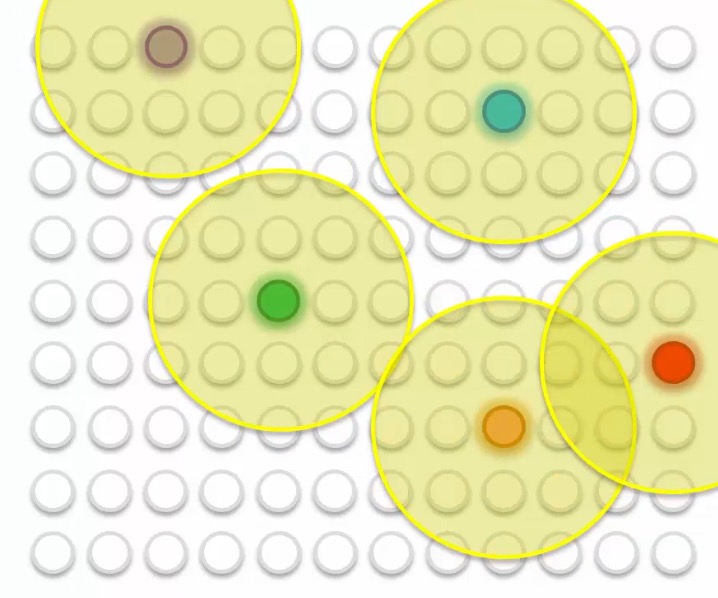
* 1. Step Two:
     1. The best matching unit in row 1 shows as the green point
     2. Updated the weight 🡪 the self-organizing maps going to drag closer to this point



* + 1. Every single point/node that falls inside the whole radius around the best matching unit is going to have its weights updated
       1. The closer to the best matching unit 🡪 the more weight is updated



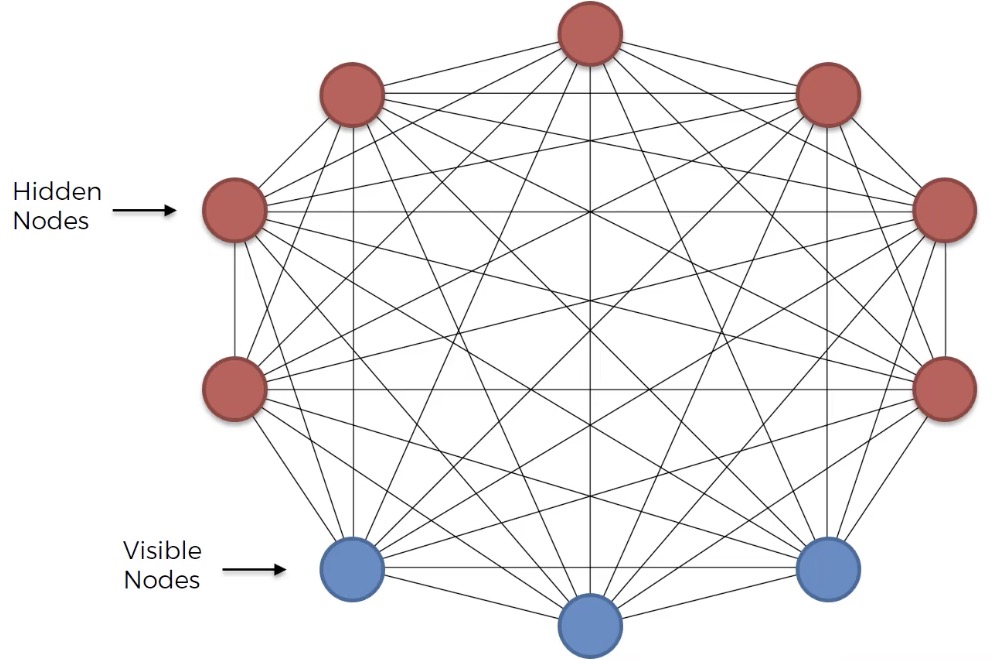
* + 1. The best matching unit in row 2 shows as the blue point
    2. For the new points, the color depends on how far closed to the green & blue points
       1. If closer to the blue point 🡪 the color will more look like blue
       2. Similar with K-means
       3. After updating the weight, they will all drag closer and closer 🡪 the radius is shrinking
  1. Results



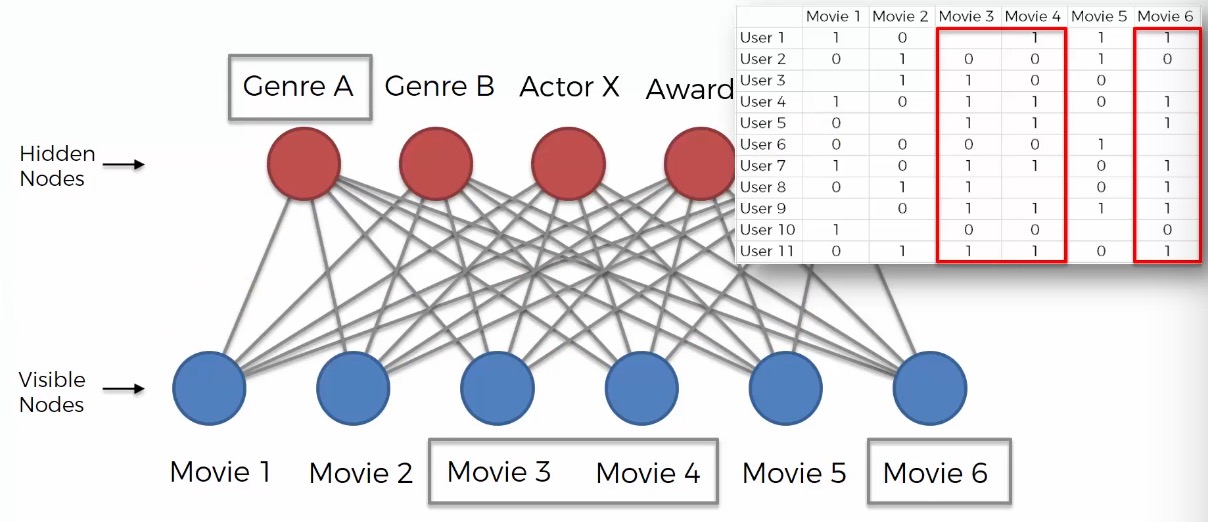
* 1. Important to know
     1. Reduce the dimensionality
     2. SOMs retain topology of the input set
     3. SOMs reveal correlations that are not easily identified
     4. SOMs classify data without supervision
     5. No target vector 🡪 no backpropagation
     6. No lateral connections between output nodes

## Boltzmann machines

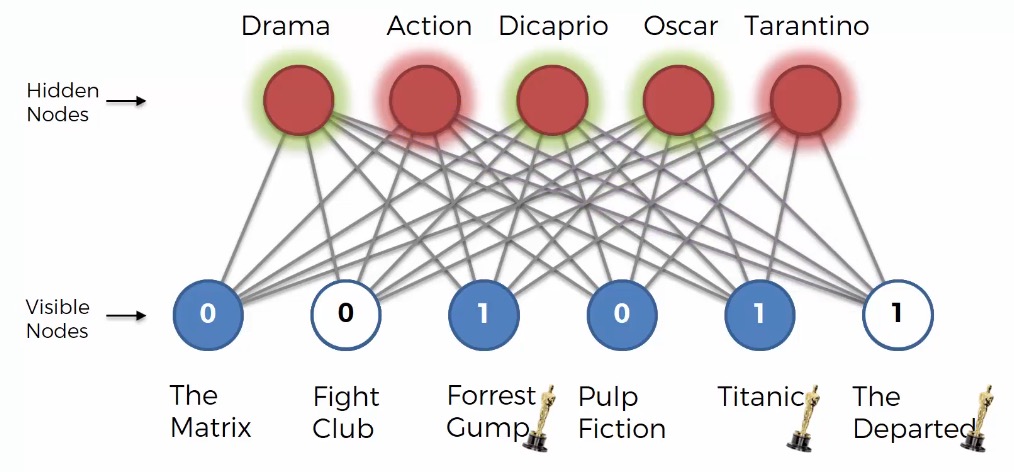
1. Features
   1. Doesn’t have an output layer
   2. Everything connects each other
   3. No direction



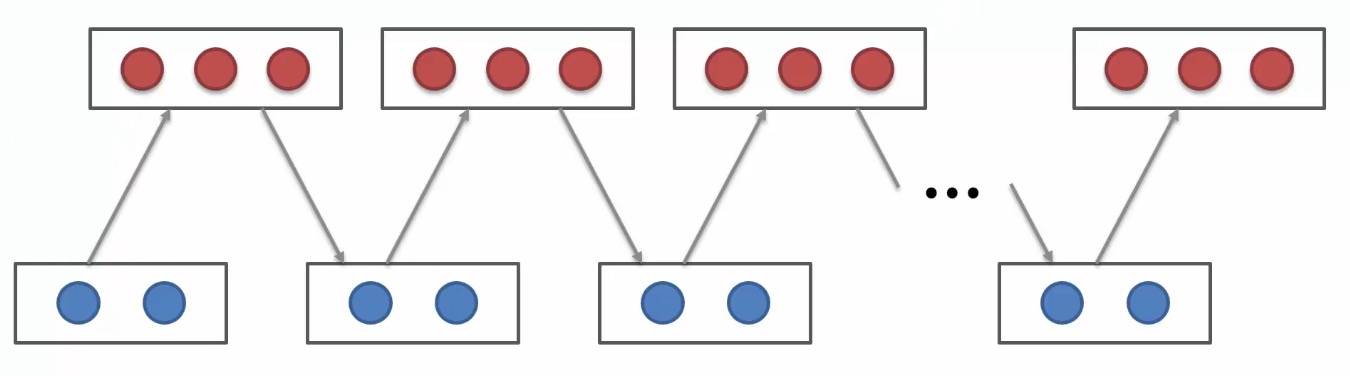
1. Energy-based models
   1. The weight will dictate and constantly try to get to the lowest energy state possible
      1. I 🡪 state
      2. εI 🡪 energy of that system
      3. K 🡪 constant
      4. T 🡪 temperature of the system
2. Restricted Boltzmann Machine
   1. Training the data
      1. Hidden nodes are some features
      2. Dataset
         1. 1🡪 Like this movie
         2. 0 🡪 Doesn’t like this movie
         3. Blank 🡪 Haven’t watch this movie
      3. We can find some relationships from the rating (similarities) 🡪 assign the nodes to the common feature
         1. A way to understand what is the features that these movies might have in common
         2. Will keep adjusting the weights



* 1. Testing the data
     1. Each hidden node connects with some specific movies
        1. Drama 🡪 Forrest Gump & Titanic = 1 🡪 Like Drama
        2. Action 🡪 The Matrix & Pulp Fiction = 0 🡪 Doesn’t Action
        3. Dicaprio 🡪 Titanic = 1 🡪 Like Dicaprio
        4. Oscar 🡪 Forrest Gump & Titanic = 1 🡪 Like Oscar
        5. Tarantino 🡪 Pulp fiction = 0 🡪 Doesn’t like Tarantino
     2. Reconstruct the input and predict the Fight Club & The Departed
        1. From the training, the Fight Club connect with Action 🡪 Since the person doesn’t like Action 🡪 he won’t like Fight Club either 🡪 Fight Club = 0
        2. With the same logic, The Departed is connected with Drama, Action, Dicaprio and Oscar 🡪 since there are more “LIKE” nodes 🡪 The person will like The Departed 🡪 The Departed = 1



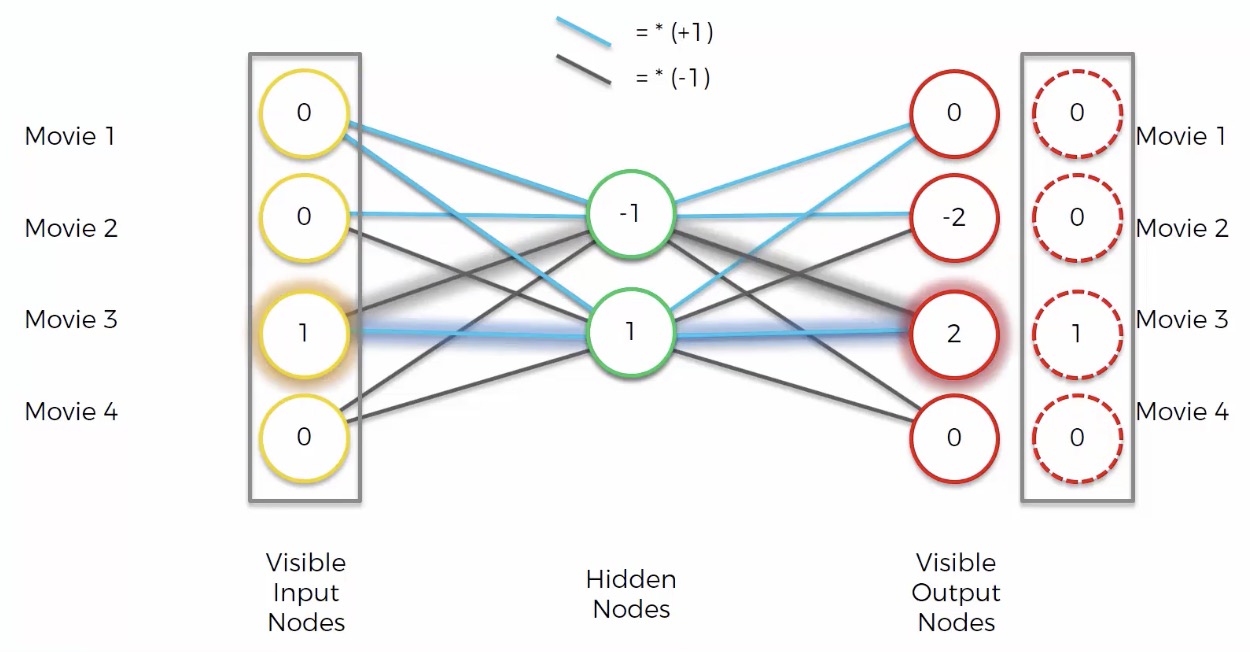
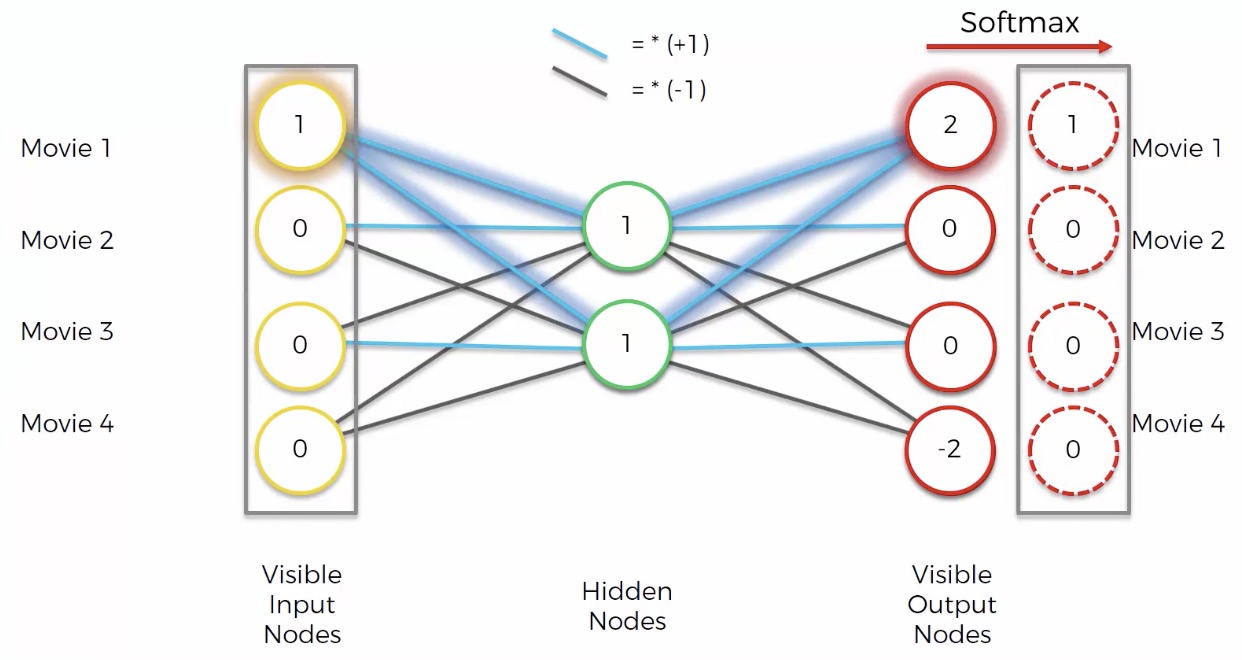
1. Contrastive Divergence
   1. If there is no back propagation (gradient descent process), how does the research machine adjust its weights?
      1. Randomly select the weight for hidden nodes 🡪 every single hidden node is constructed from all three visible nodes 🡪 All the hidden nodes will together reconstruct the visible nodes (not equal to previous visible nodes) 🡪 repeat until we get the same visible nodes back



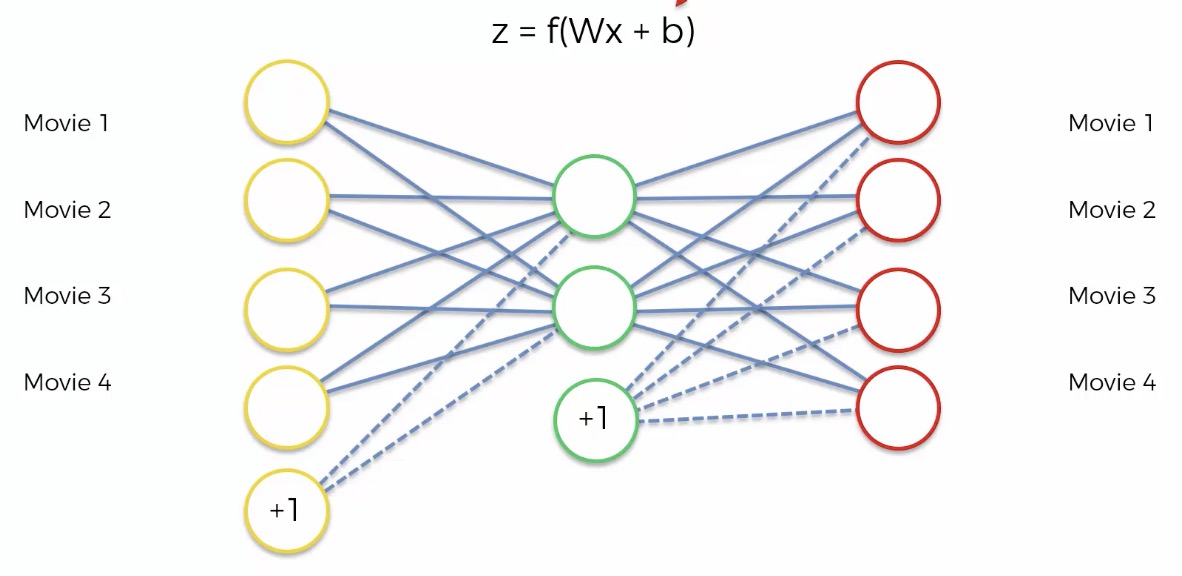
1. Deep Belief Networks (Advanced)
   1. More hidden layers above
2. Deep Boltzmann Machines (Advanced)

## autoencoders

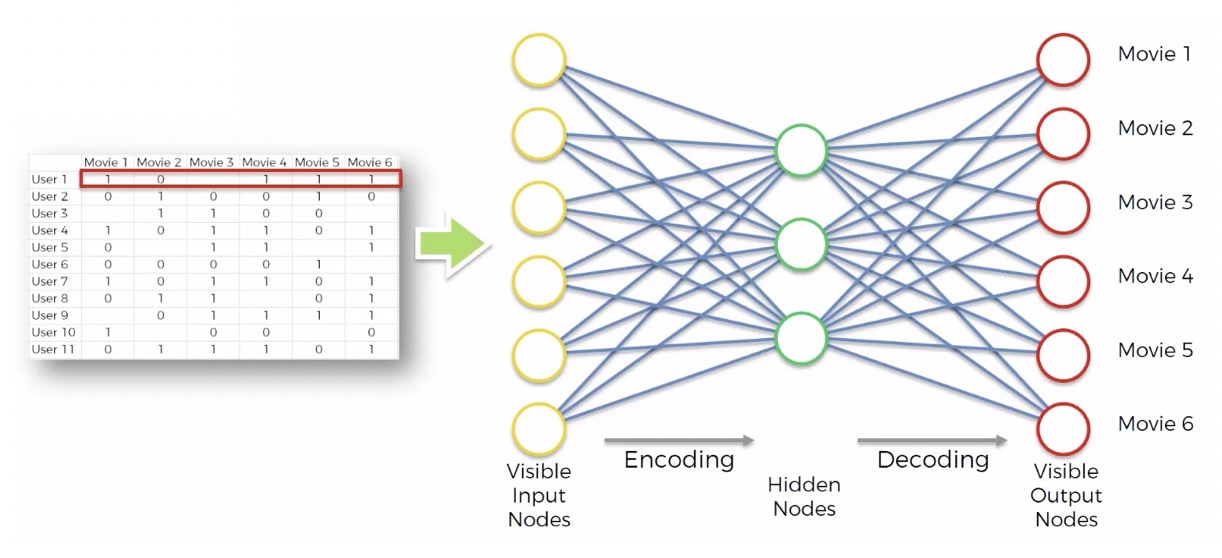
1. Use case
   1. Feature detector (hidden layers)
   2. Recommender systems
   3. Encoding data to save space
2. Example
   1. We want to prove that it is possible to take four values and encode them into two values with certain weights
   2. 1: Like the movie, 0: Unlike the movie
   3. Define the synopses (weights plus/minus 1) 🡪 Blue = \*(+1); Black = \*(-1)
   4. Step:
      1. Visible Input Nodes: whether the person like this movie or not
      2. Hidden Nodes: Calculate it by synopses weight (+1 or -1)
      3. Visible Output Nodes: Calculate the product sum of hidden nodes and its synopses weight (+1 or -1)
      4. Last step: Use softmax to further modifying the data

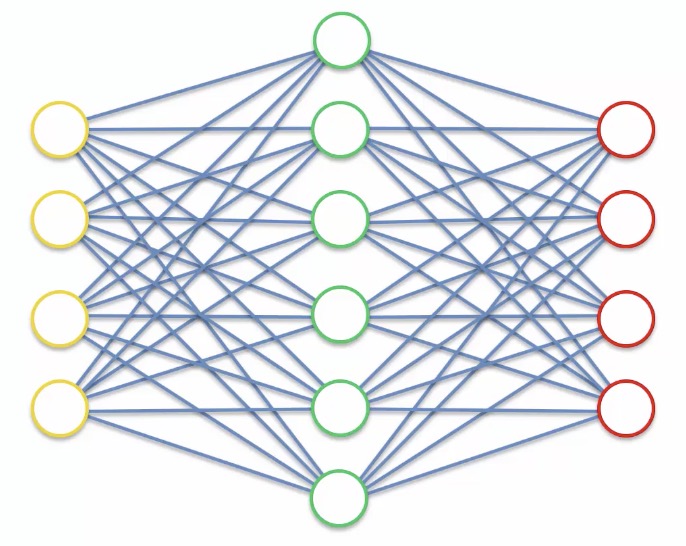


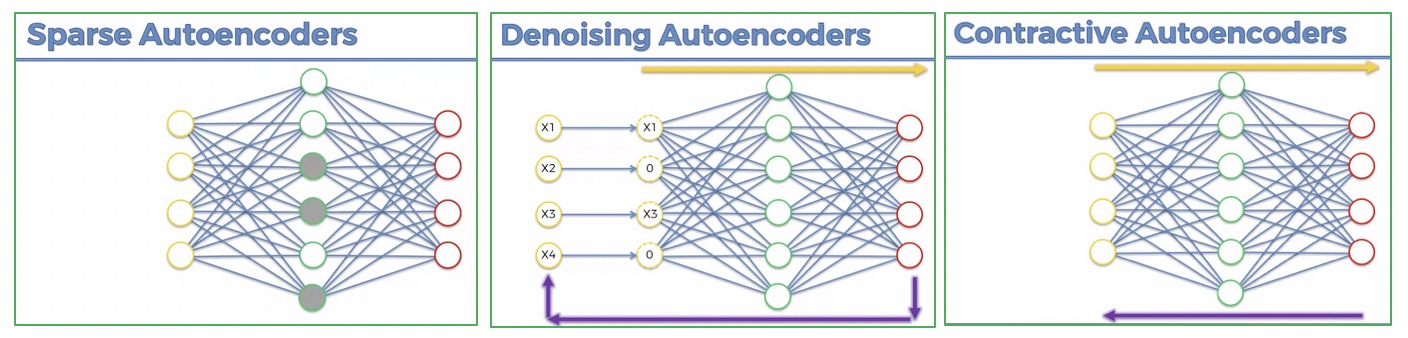
1. A Note on Biases
   1. Sometime people will consider about the biases (a constant) in the activation function



1. Training an Auto Encoder
   1. Step One 🡪 Table
      1. Start with the observations correspond to the users and the columns (the features) correspond to the movies
      2. Each cell contains the rating (from 1 to 5, 0 if no rating)
   2. Step Two 🡪 Yellow Nodes
      1. Take the first row and put into the network
   3. Step Three 🡪 Green Nodes
      1. Calculate the hidden nodes (encoded into a vector z of lower dimensions by a mapping function f)
      2. Z = f(Wx+b) 🡪 w: weight, b: bias
   4. Step Four 🡪 Red Nodes
      1. Calculate all visible output by decoding the Z 🡪 aiming to replicate the input vector x
   5. Step Five 🡪 Forward
      1. Compare the results to the actual rating by minimizing the reconstruction error
   6. Step Six 🡪 Backward
      1. Back-propagated and adjust the weights accordingly
   7. Step Seven
      1. Repeat 1-6 🡪 Take the next row in data set
      2. By using reinforcement learning or batch learning
   8. Step Eight
      1. When the whole training set passed through the ANN 🡪 makes an epoch
      2. Redo more epochs



1. Problem: overcomplete hidden layers
   1. As soon as the hidden layer which is same or greater than the input layer 🡪 able to cheat 🡪 each hidden layer can just stand for each input nodes 🡪 useless for encoding
   2. How to solve? 🡪 regularization
      1. Sparse Autoencoders
         1. Use constraint/penalty on the loss function which doesn’t allow the autoencoders to use all of its hidden layer any time
      2. Denoising Autoencoders
         1. Move input value to the left and replace with modified version of input values
            1. Randomly turn some of them into zeros
         2. Compare the output with the original values
      3. Contractive Autoencoders
         1. Add the penalty into the loss function during the back-propagated step



1. Other Advanced Autoencoders
   1. Stacked Autoencoders
      1. Have two stages of encoding (add another hidden layer)
   2. Deep Autoencoders
      1. Different with stacked autoencoders
      2. Pre-trained layer by layer

# Others

1. Data understanding & Cleaning
   1. Numerical variables:
      1. Mean, Extreme Value (Outliers)
      2. Draw visualization to see the trend and outliers/high-leverage points.
   2. Categorical variables: Number of Unique Value
   3. Missing Value – Percentage populated
      1. Replace with “NA” or “NA” with a unique number
      2. Average/Median value of subgroup
2. KS for each variable
   1. Determining if two datasets differ significantly (the maximum distance). The higher the separation, the higher the measure of goodness and importance of the variable. – Descending order and choose the top 30