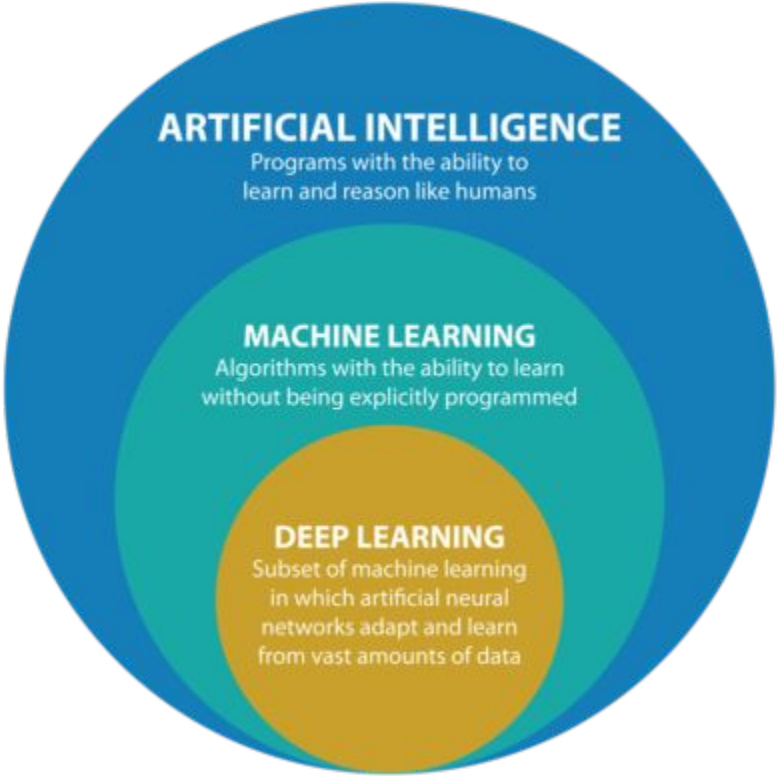
Machine Learning with Python

Week 2 - 2019 Summer

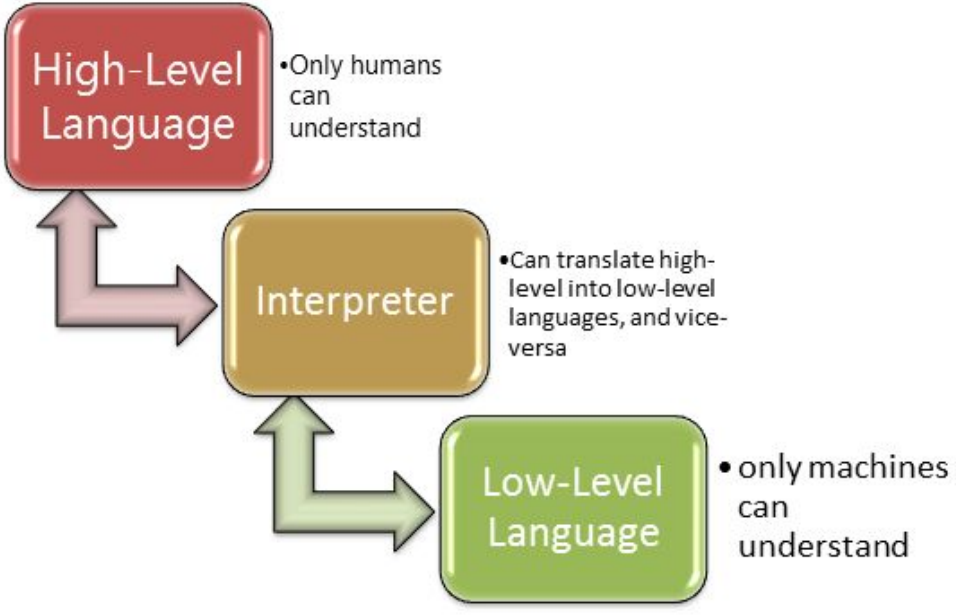
Difference between ML, AI, and Deep Learning?



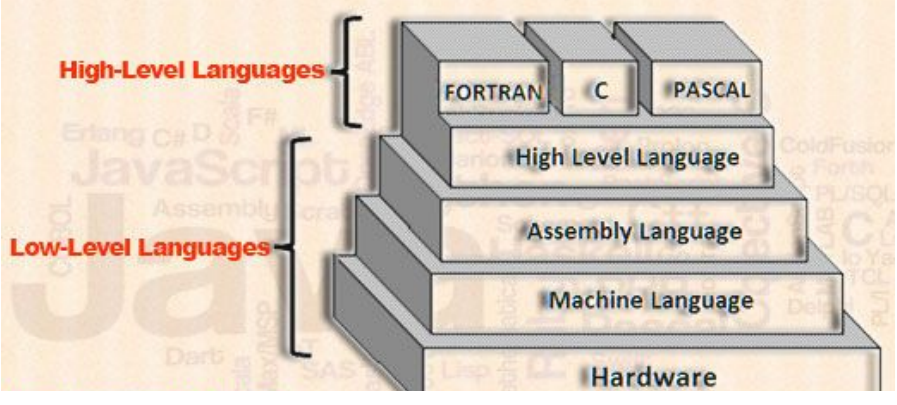

Python
programming language
type
reference
instance
design
release
Java
division
method
operator
development
function
feature
use
expressi
tement
lang
object
software
implementation
used
also
version
code
syntax
including
support
program
module
class
string
philosophy
large
tuple
About Python

Python - What It Is

- High level programming language
  - Closer to human languages, instead of machine language
- Statistical computations
- Data science, Machine Learning
- Very easy to learn quickly
  - Great syntax

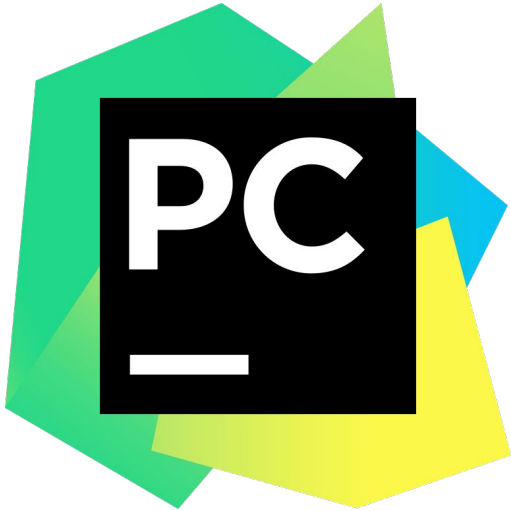


High Level → Low Level



Pycharm - IDE

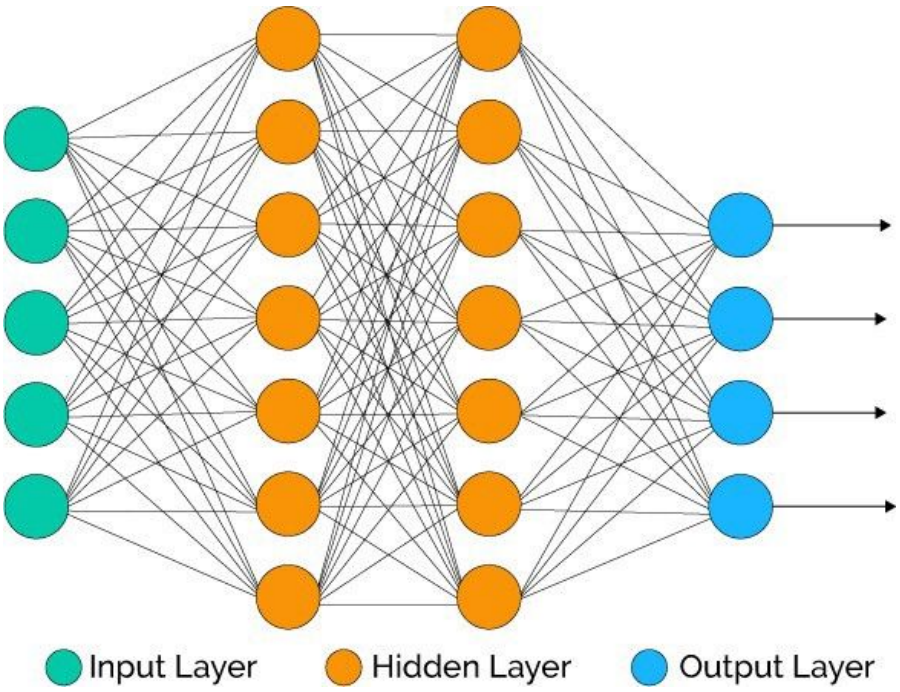
- Pycharm - IDE
  - Integrated Development Environment
- Runs python mainly
- Developed by JetBrains
  - Have multitude of IDEs such as Webstorm and IntelliJ also
- Many different IDEs - Jupyter, IDLE, PyDev, Spyder, etc.

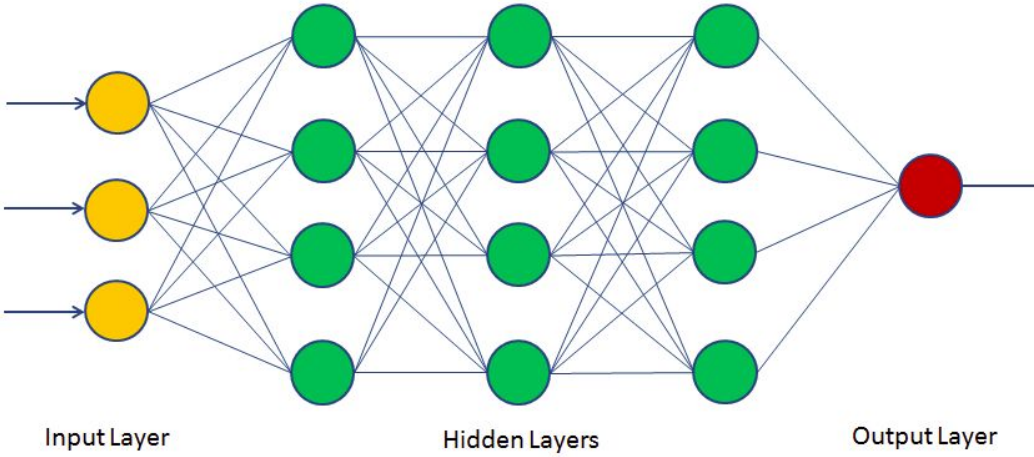


Libraries in Python

- Library
  - Collection of method/functions → incorporate in code by just calling them after importing library
- Numpy - arrays & matrices
- Scipy - scientific computing
- Scikit-learn - ML
- Theano - evaluates math expressions such as matrices
- TensorFlow - used for neural nets in ML
- PyTorch - deep learning + NLP
- Pandas - data manipulation + analysis
- Matplotlib - math extension, plotting

Building First Neural Net from Scratch

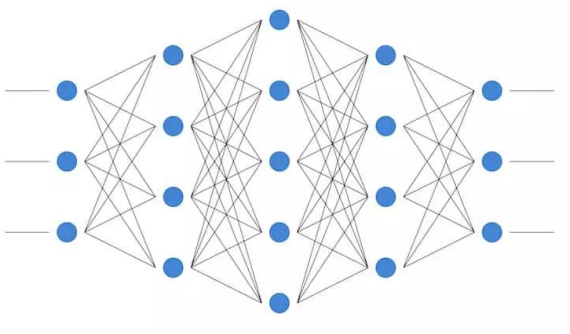

Input Layer
Hidden Layer
Output Layer


Input Layer
Hidden Layers
Output Layer

Neural Nets

What is a Neural Net?

- Models human brain
- **Set of algorithms**
- Learns to perform task through training
  - Similar to baby understanding human language, gets trained over time
- Computer analyzes training examples
- Classification, clustering, predictive analytics - regression
  - Ex - group new unlabeled data acc to previous labeled training data
  - Supervised, semi-supervised, and unsupervised forms

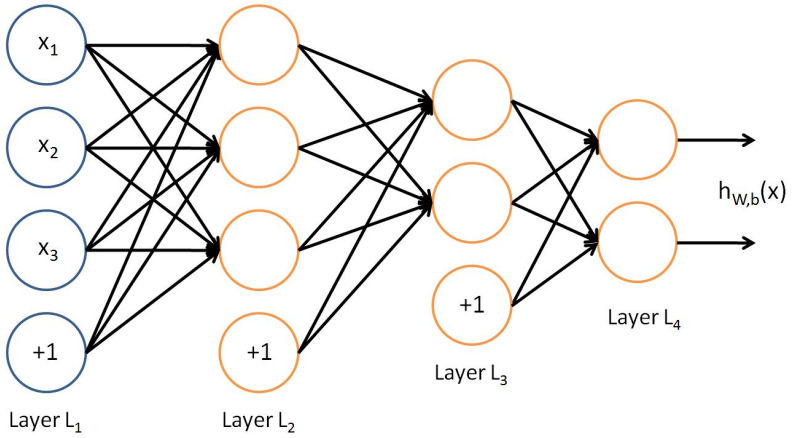


Structure of Neural Net

- Each layer → consists of nodes
  - Nodes → where computation occurs
  - Modeled off of neuron in brain
    - Neuron = nerve cell, reacting to stimuli
- Node combines input from data with coefficients/weights
  - Resemble significance of inputs (makes significance stronger or weaker)
    - Ex - how impo is this piece of input towards helping me classify this data with least poss error?
- Node's activation function
  - Summation of input-weight products
  - Decides if neuron should be activated, determines output behavior of node
- One node's output = input for consecutive layer
- Going further into neural net (into diff layers) ⇒ nodes can distinguish more complex features

Why do some Neural Nets have multiple layers?

- Single layer - linearly separable problems
- Ex - 2 entities can be classified so that we only need to draw a line between them to separate them into different classes
- HOWEVER
  - Real world → lot of problems are not linearly separable
- Need to use multi layers instead



Procedure for Building Neural Net in Python

- Teaching the neuron to reach correct answer
  - Give each input a weight
  - Larger magnitude of weight, whether positive or negative, will have larger effect on neuron's output
- STEP 1 - Apply weights on input and neuron will calc output
  - Weight = how much each feature/input value matters to the neuron, initialized randomly in beginning
- STEP 2 - Compute error (actual answer - neuron's answer)
- STEP 3 - Adjust weights depending on error (positive or negative error)
- STEP 4 - Iterate several times

$$\text{Output of neuron} = \frac{1}{1 + e^{-(\sum weight_i input_i)}}$$

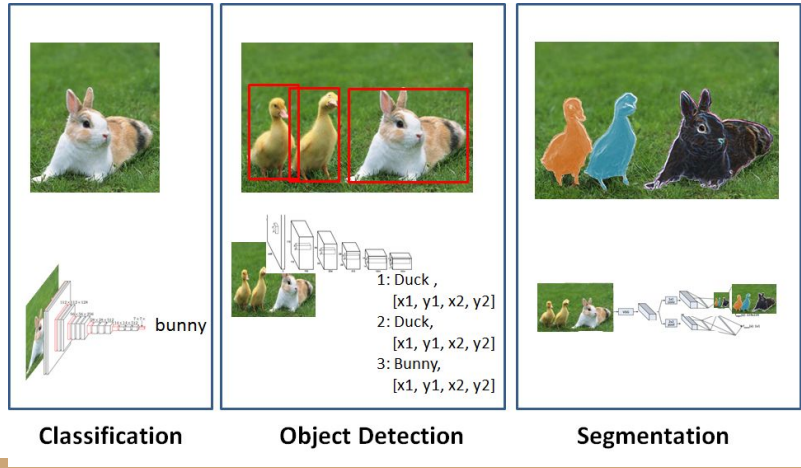
Procedure for Building Neural Net in Python

Error Weighted Derivative Formula -

$$\text{Adjust weights by} = error \cdot input \cdot SigmoidCurveGradient(output)$$

Some popular types of Neural Nets

- CNN - Convolutional neural network
  - Visual images
  - Trying to differentiate between images
  - Arranges neurons in 3 dimensions
- RNN - Recurrent Neural Network
  - Connections between nodes → direction graph
  - Looks at previous state for next states
  - Handwriting recognition, speech recognition



Unlabeled vs. Labeled Data

Unlabeled Data Example	*Example Judgment for Labeling*	*Possible Labels*	*Possible Supervisor*
Tweet	Sentiment of the tweet	*Positive/ negative*	Human/ machine
Photo	Contains *house* and *car*	*Yes/No*	Human/ machine
Audio recording	The word *football* is uttered	*Yes/No*	Human/ machine
Video	Are weapons used in the video?	*Violent/ nonviolent*	Human/ machine

But does all ML use some type of neural net?

- Supervised ML algos - classification & regression
  - Direct supervision during training
  - Developer selects type of samples and type of desired results (results are already known, just need to sort through data)
  - Based on labeled data → make predictions of future, unforseen data
- Semi-supervised ML algos
  - Limited set of labeled sample data
  - Task to label unlabeled data
  - In between supervised and unsupervised

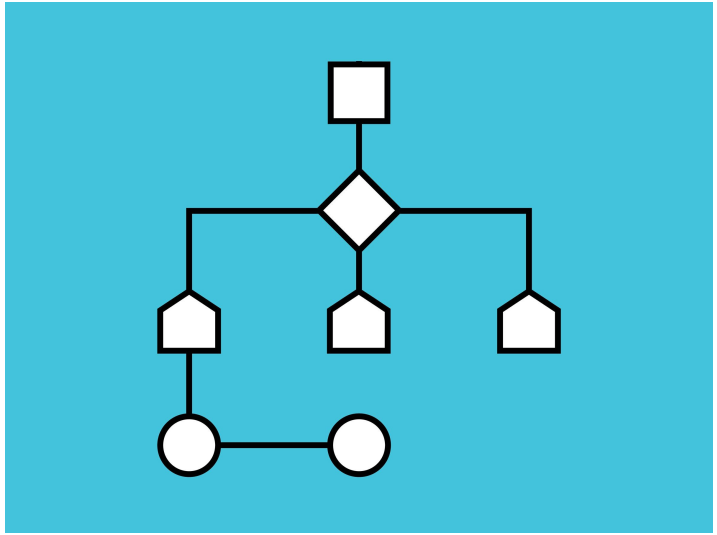
But does all ML use some type of neural net?

- Unsupervised ML algos
  - No supervision during training
  - Results are unknown
  - Uses unlabeled data (no labeled)
  - Clustering based on patterns, dimensionality reduction
- Reinforcement ML algos
  - Find best possible course of action to maximize reward

Supervised Learning with Python

Popular Algorithms

- 2 main branches - Classification & Regression
- Types of regressions - Linear, Logistic, Polynomial, etc.
- Neural Nets
- Random Forest
- Support Vector Machine
- Naive Bayes
- Decision Trees
- Gradient Boosted Trees



1. Linear Regression in Python

- Predict some value (Y) over given set of features (X)
- Train and test phases
  - Train = machine fits function from labeled training sets
  - Test = machine predicts Y for given unlabeled training set
- Goal - reduce error → improves accuracy of model, decrease value of loss function
  - Find model parameters so cost function
    - METHOD = GRADIENT DESCENT

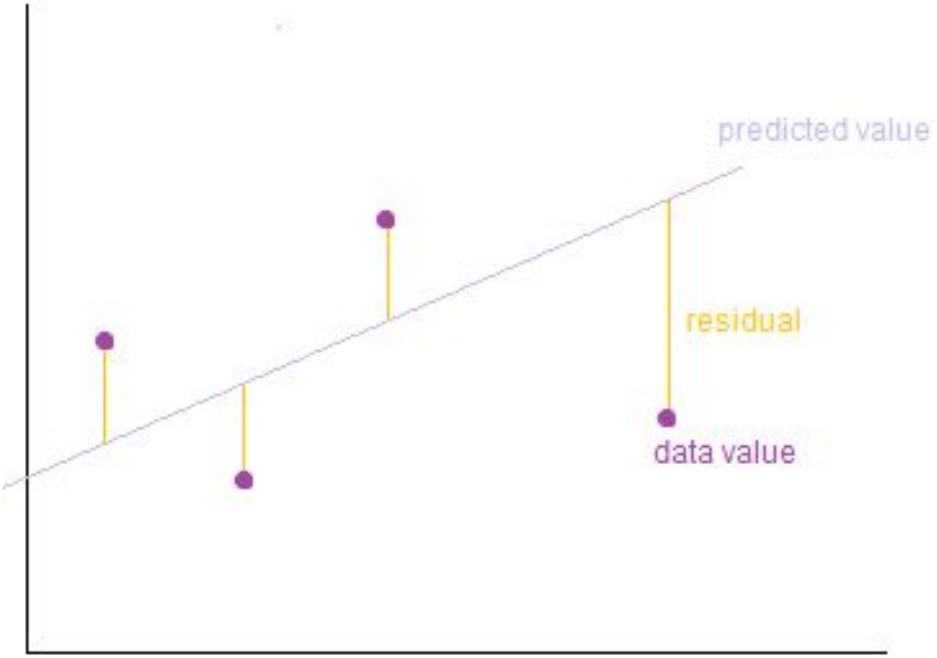
$$Y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n$$

- Y is the predicted value
- θ_0 is the bias term.
- $\theta_1, \ldots, \theta_n$ are the model parameters
- $x_1, x_2, \ldots, x_n$ are the feature values.

1. Linear Regression in Python

$$\widehat{Y} = f(X) + \epsilon$$

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$



1. Linear Regression in Python - Mean Squared Error Function

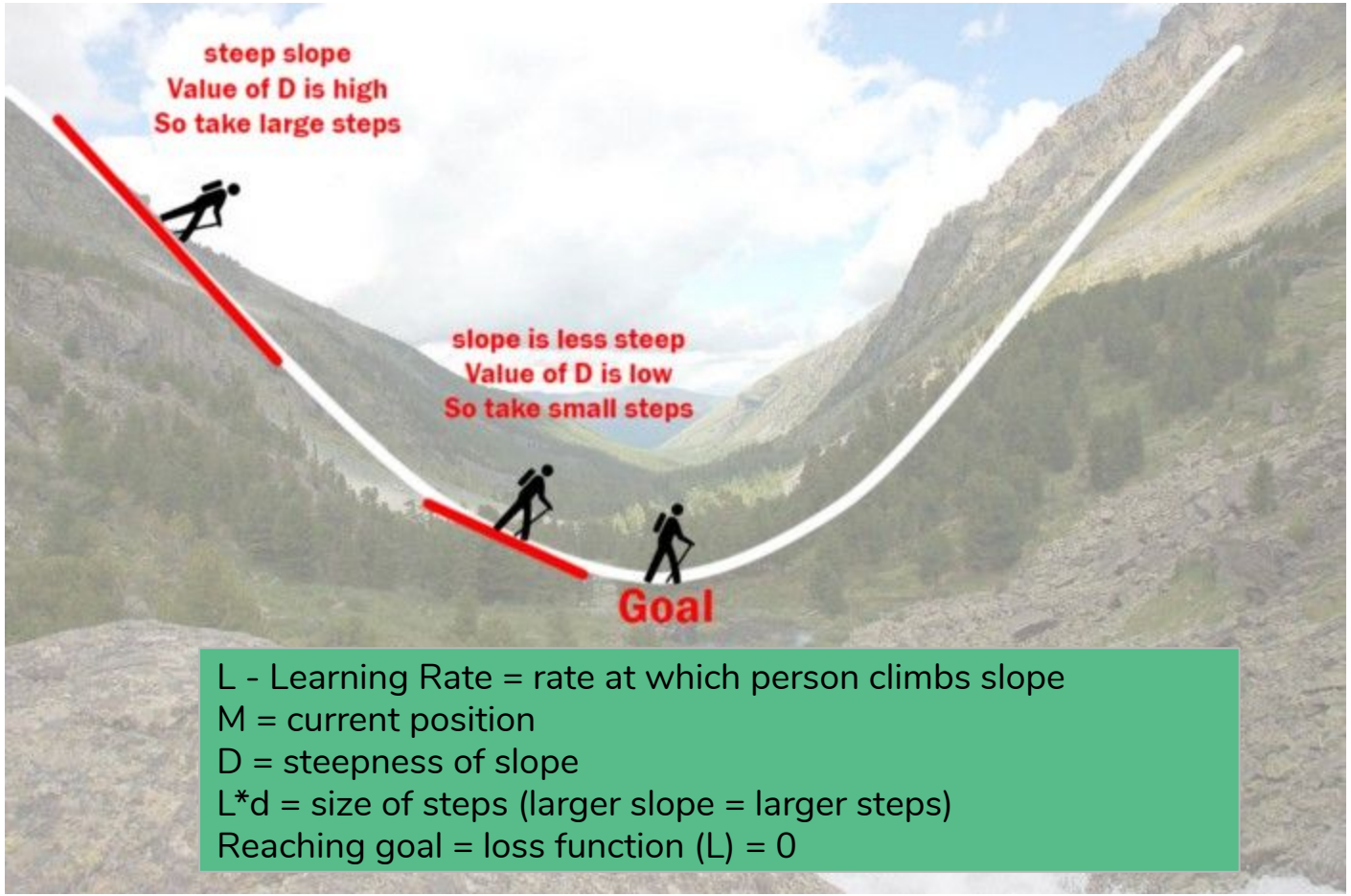
- Obtain residual (y - yhat), where yhat = predicted
- Square this residual
- Find the mean of these squares

Find the mean of the squares

$$\frac{1}{n}\sum_{i=0}^{n}(y_i - \bar{y}_i)^2$$

1. Linear Regression in Python - Gradient Descent

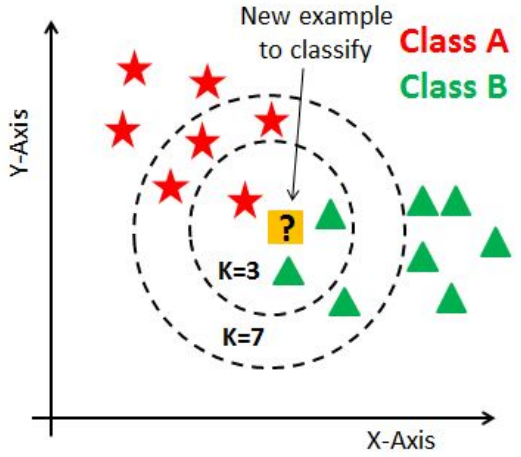
- Optimization algorithm, iterative, finds min value of loss/error function
- STEP 1 = random initialization
- STEP 2 = see how cost function changes when model parameters change
  - Get partial derivatives of cost function considering the parameters
- STEP 3 = update parameters after computing partial derivatives
- Steps 2 & 3 repeated
  - Each iteration of gradient descent → cost function decreases
  - Till cost function converges to a particular min value



L - Learning Rate = rate at which person climbs slope
M = current position
D = steepness of slope
L*d = size of steps (larger slope = larger steps)
Reaching goal = loss function (L) = 0

2. K Nearest Neighbors Algorithm

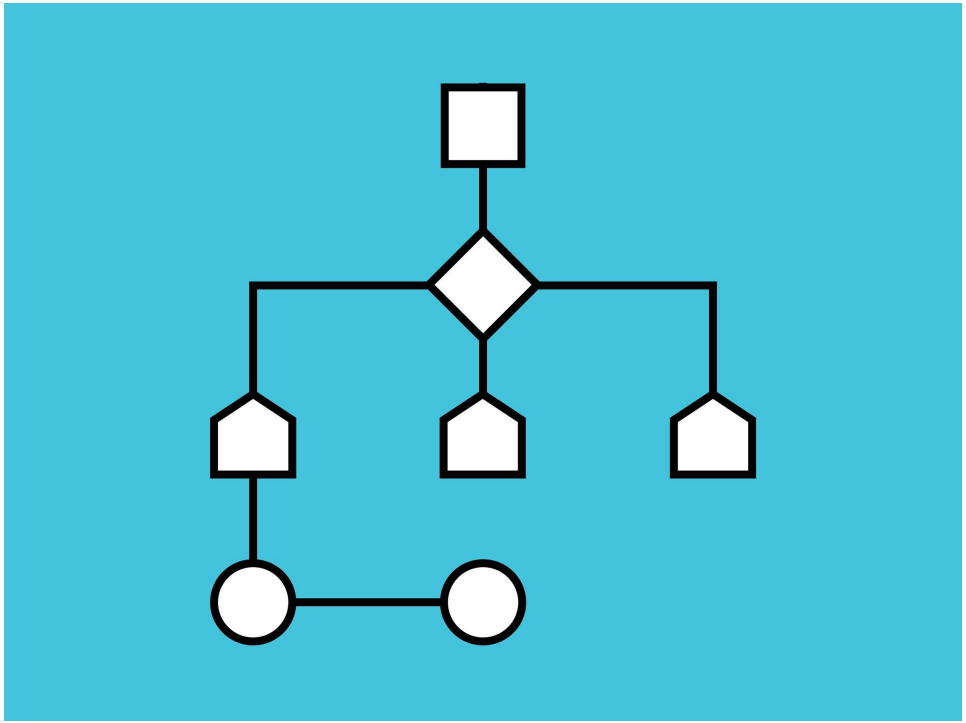
- Classification algorithm
- Classifies new cases based on distance
- K is number of neighbors
- Ex - Let new point = x, find the k closest neighbors to it using any distance function
  - Calculate distance from each object to this point
  - Obtain k closest neighbors
  - Look at vote for each label
  - Look at majority
  - Able to classify that point based on majority rule



Unsupervised Learning with Python

Popular Algorithms

- Clustering algos
  - K means
  - Hierarchical
- Finding patterns
- Dimensionality reduction

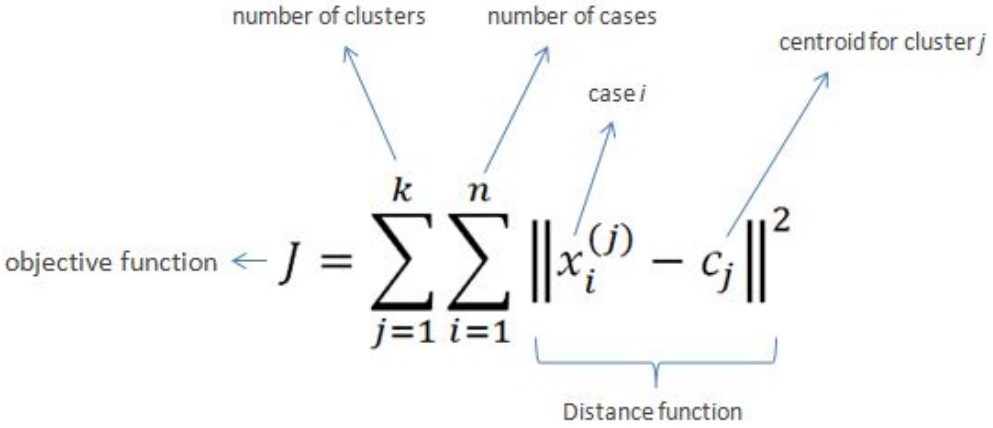


1. K-means Clustering Algorithm

- Group similar data points together in clusters
- Looks for k number of clusters
- Able to deduce patterns from clusters
- K = number centroids
  - Centroid = center of each cluster
- Each data point in 1 cluster
- Why k-**means**?
  - Means = averaging data → centroid

1. K-means Clustering Algorithm - Procedure

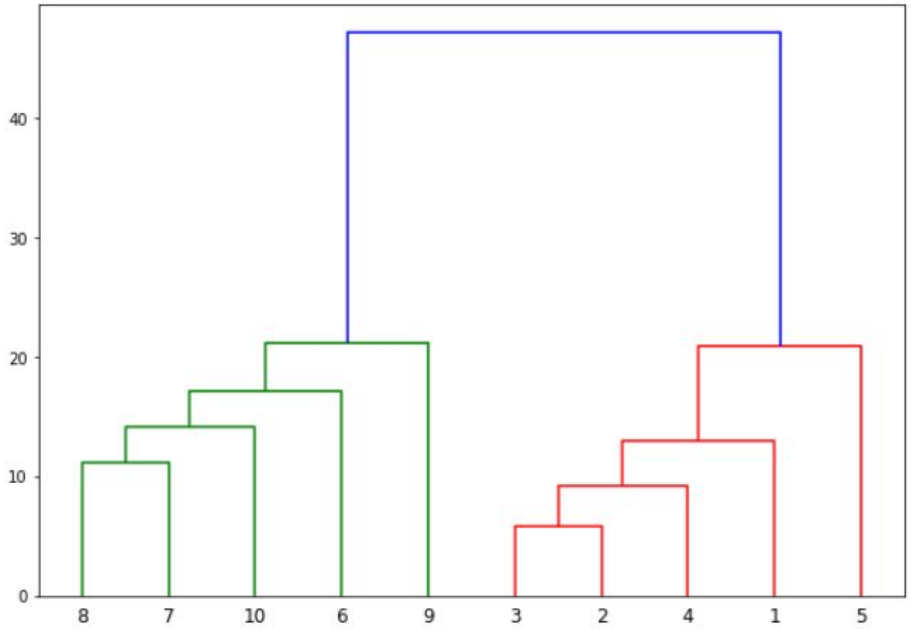
- STEP 1 - Randomly select centroids
- STEP 2 - Iterates till finds optimal position of centroids
  - Either centroid values don't change and stay constant OR
  - Number of iterations previously defined in beginning has been reached
- https://www.saedsayad.com/clustering_kmeans.htm



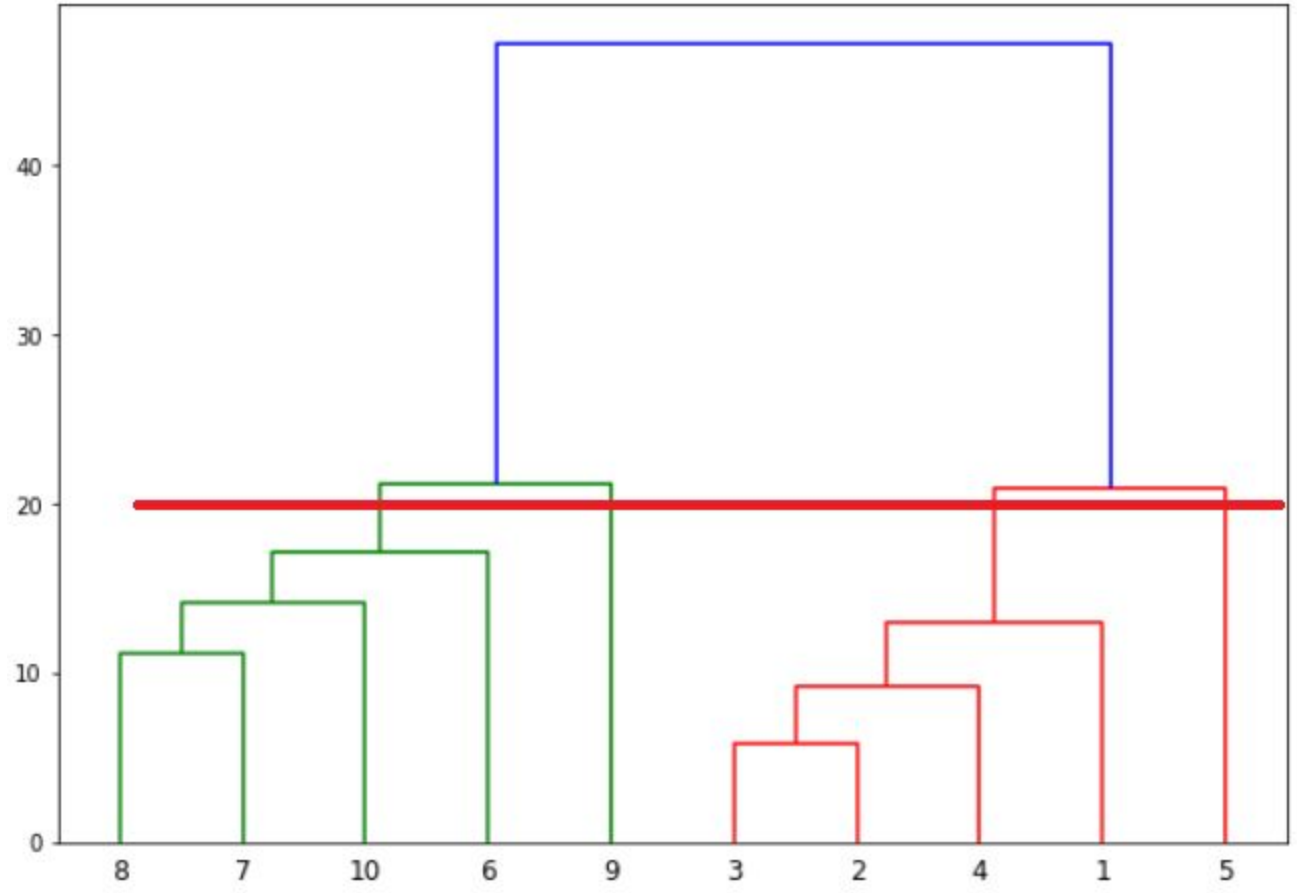
2. Hierarchical Clustering (Agglomerative) Algorithm

- 2 types of hierarchical clustering
  - Agglomerative and divisive
    - Bottom up vs top down approach to clustering
    - Bottom up = start with individual data points and cluster, and top down = start with one big cluster and split into diff smaller clusters
- STEP 1 - each data point is 1 cluster, # clusters = k
- STEP 2 - Form cluster with 2 closest points = k-1 clusters
- STEP 3 - Continue to form more clusters → k - 2 clusters (join 2 closest together)
- STEP 4 - Repeat till 1 big cluster
  - Dendograms used to divide into multiple clusters
- Key step - finding distance between clusters (Ex : Euclidean distance)

2. Hierarchical Clustering (Agglomerative) Algorithm



- Records sequences of cluster merges
- Vertical height = euclidean distance
- Finds 2 points closest to each other first
  - Dendrograms formed with 2 and 3 and 7 and 8
  - Then 4 is joined with 2-3 cluster, and so on till 1 big cluster formed
- Draw horizontal line → min threshold for new clusters to form
  - Number of vertical lines it crosses = number clusters


40
30
20
10
0
8
7
10
6
9
3
2
4
1
5

How Ward's Clustering Method Works

https://www.statisticshowto.datasciencecentral.com/wards-method/

https://www.youtube.com/watch?v=ukzF19rgwfU

https://www.youtube.com/watch?v=f_uwKZIAeM0