**Artificial Intelligence –** Two Approaches:

**Symbolic AI**: Uses mathematical symbols to represent objects and the relationships between them

**Connectionist AI**: Precursor to Deep Learning

**Types of ML Models:**

**Classification:**

* ***Binary Classification***: Predictions is a yes or no and where output is in the form of categories

Ex algorithm: Logistic Regression

* ***Multi-Class Classification***: One output for more than two possible outcomes.

Ex algorithm: Multinomial Logistic Regression. Is a movie a comedy, documentary, or Thriller?

**Regression** (Linear or Non Linear)

* ***Simple Regression***: (1 input; 1 output) - Predict a numeric value where outcome is based on average of what has occurred. Example algorithm: Linear Regression. e.g. Home Value, Salary
* ***Multivariant Linear Regression*:** (Multiple input; 1 output)– Regression using various data features. i.e. how much will it rain today based on 1. How much it rained yesterday, 2. Is it summer or winter

Linear Regression Models:

Models alleviates collinearity amongst variables which leads to overfitting.

* **Ridge Regression** - Adds a small squared Bias factor
* **Lasso Regression –** Adds an absolute value Bias

**ML Learning Models:**

* Supervised Learning Model
* Unsupervised Learning Model
* Semi-supervised Learning Model
* Reinforcement Learning Model
* Nature inspired Learning Model
* Bio-inspired Learning Model
* Statistical Learning Model

**ML Learning**

**Supervised Learning** – Feedback is given through every step

Use labeled training data to learn mapping function of input X to output Y: y = f(x)

(usually regression tasks) Answer is known, training exposes relationships or weights.

**Classification Algorithms (Linear Classifiers)** (e.g. Speech, Handwriting recognition, bio-metrics)

* **Logistic Regression** – Predicts outcomes when you have at least 2 variables. E.g. Predict lung cancer using variables of height, weight, age and smoking history
* **Linear Regression** – Y = ax + b (e.g. Rainfall in CM)
* **Support Vector Machines (SVM)** (*See* *Below*) – Creates a separation of classes and finds a dividing line or hyperplane that separates classes
* **Boosted Trees** – Weighted voting converting weak learners to strong
* **Neural Networks** – (*See Below*)
* **Naïve Bayes** – Classification: uses Bayes theorem; probability an event occurs given a preceding event occurs. Called Naïve because it assumes all variables are independent.

P(h|d) = (P(d|h) P(h)) / P(d)

**Classification & Regression** Algorithms

* **Decision Trees** –Choose best feature to split upon

**Note**: **CART** = **C**lassification **A**nd **R**egression **T**rees (i.e. ID3 or C4.5) –

Terminal Nodes are Leaf Nodes – Output Variable (Y)

Non-terminal nodes are everything else, i.e.

Non-terminal nodes are root node, input variable (X) and internal nodes.

* **Random Forest** – Random selection of feature to split.
* **K Nearest Neighbor** (**KNN)** – Find K nearest instances similar to new instance.

Similarities are calculated using measures such as Euclidean or Hamming distance

Output most frequent class or mode (**Classification**)

Output mean of outcomes (**Regression**)

**Ensemble Learning**: A type of supervised learning that combines predictions from multiple models. Voting used for **Classification,** Average used for **Regression**

Example ensemble algorithms

* **Bagging** – Bootstrap Aggregation. Parallel learning – Simple Voting.

Example: Bagging with Random Forests.

* **Bagged decision tree** – 1 learner
* **Bagged Random Forest** – Multiple Listeners

Boot Strap Sampling

1. Data generated from random subset of original data
2. Dataset uses repetition to keep dataset size the same as original
3. Original dataset used as a test dataset.

Train multiple modes on each data subset using same model

* **Boosting** – Sequential Learning – Weighted Voting. Convert weak learners to strong learners. Assign greater weights to misclassified instances of previous models

Example algorithms: **ADABoost** (Adaptative Boost) or **CATBoost** (Categorial Gradient Boosting)

* **Gradient Boosting** – **XGBoost** (extreme Gradient Boosting)
* **Gradient Boosted decision tree** –
* **R gbm** (gbm = gradient boosting machine) (old)
* **Stacking** – Combines multiple classifications or regression models via meta-classifiers or meta-regressors (models).

Base models trained on complete dataset. Meta Models trained on Base Model as features.

**UnSupervised Learning** – Never receives feedback.

Finds relationships using known input variables (X) but no output variables. Training exposes the

prediction and outliers. e.g. “Women age 25 from the west coast prefer blue jeans”

1. **Association** – Finds probability of co-occurrence i.e. Someone who buys bread is 80% likely to buy eggs.

* **Apriori** (from earlier) – Generate associations between frequent items in dataset X->Y

Principle is that if an itemset is frequent then all of its subsets are frequent.

/🡪 Support = frq(x,y) / N

Rule X->Y --🡪 Confidence = frq(x,y) /frq(x)

\🡪 Lift = Support/(supp(x) + supp(y))

**Eclat** (**E**quivalence **CLA**ss **T**ransformation) – link([video](https://www.youtube.com/watch?v=oBiq8cMkTCU))

1. **Clustering** – Finds similar Data Points and groups objects together

* **K-Means** – Groups similar data into clusters. Assigns a data point to the clusters having least distance between its centroid and the data point.
* **HCA** (Hierarchical Cluster Analysis)
* **Expectation Maximization**

1. **Dimensionality Reduction** – Reduce data set variables while keeping important information.

**Visualization** **algorithms** – Input data is output into a 2D or 3D representation that can be easily plotted

1. **Feature Selection** (aka variable, attribute or variable subset selection) – Select subset of features in training for model construction.
2. **Feature Extraction** – Data transformation: Moves high dimensional space to low dimensional space.

**Algorithms**:

* **PCA** – Principle Component Analysis – Reduces the number of variables (called “Principle Components”) in the data (Dimensionality).
* **Kernel PCA** – Uses Kernal Methods
* **LLE** - (Locally Linear Embedding)
* **t-SNE** – (t-distributed **S**tochastic **N**eighbor **E**mbedding)

1. **Anomaly Detection -** Finds outliers in a collection of data. Ex: Banks look for fraudulent transactions

**Reinforcement Learning** – Feedback given only when learning achieves its goal.

Interacts with its environment learning through trial and error.

**Deep Learning –**

**Artificial Neural Network (ANN)** – Computer system with lots of connected nodes (Neurons) arranged in layers passing information from one layer to another.

* **Neural Networks** (see below)
* **Recursive Neural Networks** – Hierarchical Network – Input processed hierarchically in a tree fashion
* **Radical Basis Function Neural Network** – Used in Cybersecurity, healthcare and power restoration
* **Kohonen Self-Organizing Network** –Classification used in healthcare, Image & Speech recognition
* **Modular Neural Network** – Useful when you have several independent NNs working on same task.

**Sequence to Sequence (Seq2Seq) Learning** – Training exposes relationships between sequences. Examples:

Language Translations, AI Chatbots

**Adversarial Training** – Include bad data to provide a more robust model

* **Deep Convolutional Generative Adversarial Network (DCGAN)** – To generate physical models you must have an understanding of real-world components and relationships.

**Adversarial Network** – Two networks that fight each other which makes both stronger. 2 types:

1. Discriminator: Tries to make correct prediction from input data
2. Generator: Creates data to fool the Discriminator

**More information:** Algorithms:

**Neural Networks**

Concepts:

* **Perceptron** - Single layer feedforward (hidden) Neural Network

Consists of 4 parts: 1. Input Layer, 2. Weights and Bias, 3. Net Sum, 4. Activation Function

* **Feature Scaling** – Normalize data during pre-processing
* **Activation Function (**aka **Transfer Function)** – Defines or scales or normalizes the output to fit requirements for input node. e.g. (0,1) or (-1,1). Two types:

1. Linear - Equation: f(x) = x, Range: (-α to α)
2. Non Linear – (Many) for example:
3. **Sigmoid** or Logistic – Range: (0 to 1)
4. **Tanh** or Hyperbolic Tangent function – Range: (-1 to 1)
5. **ReLU** (**R**ectified **L**inear **U**nit) – Range: (0 to α)
6. Leaky ReLU or Randomized ReLU (if leak is not 0.01 then it is random) – Range: (-α to α)

* **Back Propagation** – (see Gradient Descent) Walk back error through hidden layers to adjust weights
* **Drop Outs** – Solution to overfitting – Simplify (thin) the network by “dropping” nodes forcing new paths.
* **SoftMax Function** – Range (0 – 1) to create percentages that add up to one for use in output predictions for neural networks.
* **Convolutions** – example (input: image, output: edges). Two convolutions second output: lines

Each convolution network looks for a specific pattern

Problems:

* **Vulnerable to Spoofing** – Adversarial attack
* **Opacity** – Not known how it makes decisions
* **Lack of common sense** – e.g. recognizes chairs & sofas but doesn’t know they are for sitting.

Algorithms: (see Deep Learning)

* **Feed Forward Neural Network (FFNN)** – First & simplest NN. Data only flows in one direction.

Example: Image & Speech recognition

* **Recurrent Neural Network (RNN)** – Add state information into Neural Network. Make stateful

Example: Text to speech, text predictions

* **LSTM** – Long Short-Term Memory network – A special type of Recurrent Neural Network. Aspects:
  1. Has control to decide when to let the input enter a neuron
  2. Has control to decide when to remember what was computed in a previous step
  3. Has control to decide when to let output pass to the next time stamp

Simulates how the human brain can handle sudden contact switches based on input

Uses relevant information which may be far from where it is needed

* Ex: I grew up in France… I speak fluent *French,* fluent; Label video frames. i.e. are characters **Deep Neural Network** – More hidden layers
* **Convolutional Neural Network** – Used in Computer Vision, image classification, signal processing
  1. Convolutional layer: Use a model to make predictions on an N x N matrix (Usually 2 x 2)
  2. Layer that figures out the most important features from a dataset
  3. Max Pooling - Down sample algorithm. Keep the most interesting prediction of the N x N matrix
     1. Down sample an input representation & reduce dimensionality
     2. Apply a max filter to usually nonoverlapping subregions of the initial representation
  4. Fully Connected – Layer that compares new images to existing data to find make final prediction
* **Graph Convolutional Network**
* **Support Vector Machines** (**SVM**) – Creates separation of classes and finds a line/hyperplane that separates classes.

1. Two-step process
2. Create transformation such as W=X2 + Y2 to add a Z-axis where W=Z
3. Transform back into original plane
4. Tuning Parameters

* **Support Vector Regression (SVR) –** SVM used for classification

**Kernel** **Methods**– **Kernel Functions** (Similarity function): Given two objects a kernel function will output similarity scores

Linear Kernel – Prediction using the dot product between input X and each support vector xi. I.e. f(X) = B(0) + Sum (ai \* (x,xi))

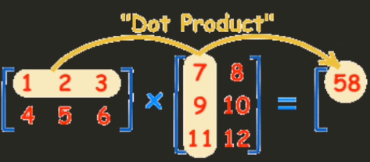
* + Linear: (X1, X2) = X1\* X2
  + Polynomial: k(X1, X2) = (λ X1\* X2 + c)d
  + Gaussian or Radical Basis: k(X1, X2) = exp(λ ІІX1\* X2ІІ2)
  + Sigmoid: : k(X1, X2) = tanh(λ X1\* X2 + c)
* **Gamma –** How far the influence of a single training example reaches (Low meaning Far, High meaning Close)
* **Margin –** Separation of Line to the closest class points

**Definitions**

* **Gradient** – An increase or decrease in the magnitude of a property – e.g. the slope of a line
* **A Priori -** "from before." If you know how many red, white, and blue gum balls are in the gum ball machine, this a priori knowledge can help you predict the color of the next ones to be dispensed.
* **Stochastic -** having a random probability distribution or pattern that may be analyzed statistically but may not be predicted precisely.
* **Euclidean Algorithm or Distance** - An efficient method for computing the greatest common divisor of two numbers. i.e. the "ordinary" straight-line distance between two points in Euclidean space
* **Hamming Distance** - Distance between two strings or numbers of equal length is the number of positions at which the corresponding symbols are different

**Concepts**

* **Adversarial Images -** Images whose class category looks obvious to a human, but causes **massive** **failures** in a deep network.
* [Attention Mechanism](https://skymind.ai/wiki/attention-mechanism-memory-network) –
* **Batch Normalization** – Eliminate or reduce the effect of outliers. Two Problems:
  1. Weights Problem – Initial weights may contain many outliers
  2. Learning Weights Problem – Gradients from outliers affect learning weights
* **BIAS** – Shifts activation function up or down
* **Bucketing** and **Padding** – For Seq2Seq networks –
  + Split sequence into fixed length buckets,
  + Pad smaller sequences to fit fixed size buckets
* **Chain Rule** - is a [formula](https://en.wikipedia.org/wiki/Formula) for computing the [derivative](https://en.wikipedia.org/wiki/Derivative) of the [composition](https://en.wikipedia.org/wiki/Function_composition) of two or more [functions](https://en.wikipedia.org/wiki/Function_(mathematics)).
* **Collinearity** - A condition in which some of the independent variables are highly correlated.
* **Defensive Distillation** – Create second model to mimic first model to defeat adversarial attacks
* **Distance Function (aka Distance Metric) -** 
* **Dot Product – Multiply rows by columns to achieve result**

****

* [Eigenvalue](https://medium.com/fintechexplained/what-are-eigenvalues-and-eigenvectors-a-must-know-concept-for-machine-learning-80d0fd330e47) **-** The scalar that is used to transform (stretch) an Eigenvector.
* [Eigenvector](https://medium.com/fintechexplained/what-are-eigenvalues-and-eigenvectors-a-must-know-concept-for-machine-learning-80d0fd330e47) - Vectors that, when a linear transformation (i.e. multiplying by a scalar) is performed, does not change its direction. Every vector (list of numbers) has a direction when it is plotted on a XY chart.
* **Encodings** (aka **Embeddings**) – Transform an object into a numerical representation
* **Fourier Transform** – Break apart complex sound waves into component frequencies
* **Gated Recurrent Unit Neural Network** (GRU) -
* **GNB** (Gaussian Naïve Bayes)
* **Gradient Descent** (called with Neural Networks **Back propagation**) – A method to calculate a partial derivative (or gradient) of a function to determine its slope. Two common ways to compute:
  1. **Analytic differentiation** – computes derivatives using the chain rule
  2. **Approximate differentiation** – using finite difference
* **Image processing**:
* **HOG** – Histogram of Oriented Graphics – Face detection using black and white images looking at pixel brightness to create vectors (arrows called gradients) with slope from brighter to darker
* **Face Landmark Estimation** – Measure distances to a number of landmarks on a face
* **Affine Transformation** – Transform rotation and scale preserving parallel lines
* **Loss Function** – Error - How far off is the prediction
* **Markov Model** - It is assumed that future states depend only on the current state, not on the events that occurred before it (that is, it assumes the [Markov property](https://en.wikipedia.org/wiki/Markov_property))
* **MobileNets** - an architecture for running deep networks on mobile devices. They use a different style of convolutions to reduce both memory consumption and inference time.
* **Model Compression** – Typically accomplished using Teacher-Student Training since student models can have near teacher-level performance, even while using 1-2 orders of magnitude fewer parameters
* **Multi-task Cascaded Convolutional Network** (MTCNN) –
* **Multiple Linear Regression (MLR) -** a statistical technique that looks at how multiple independent variables are related to a dependent variable by modeling the linear relationship between independent variables and a dependent variable using several variables to predict the outcome.
* **One-Shot Learning -** aims to learn information about object categories from one, or only a few, training images. Often done using Siamese Network.
* **Overfitting** – 100% prediction on the training data, poor prediction on real data
  + **Regularization** – Penalize Loss function by adding more data
    - Lasso – Sum the absolute values of weights
    - Ridge – Sum the squared values of weights
  + **Cross-Validation** – Train on subset of data to validate with rest of data
    - Holdout – Verify predictions using untrained data
    - K Fold – Train multiple models using subsets and perform holdout on each model
  + **Dropout** – Disable nodes in a Neural Network to force different paths during training
* **Performance Metrics**. Consists of:
  + ***Accuracy*** – Percentage of time guess was true: (TP+TN)/(TP+TN+FN+FP)
  + ***Precision*** = True positives / All positive guesses: TP/(TP+FP)
  + ***Recall*** = True positives / Total positives: TP/(TP+FN)

**TP** = True Positive**, TN** = True Negative**, FP** = False Positive**, FN** = False Negative

* **Pipeline** – A machine learning workflow. Can be automated
* **Siamese Network –** An ANN that uses the same weights while working in tandem on two different input vectors to compute comparable output vectors. Often one of the output vectors are precomputed, thus forming a baseline the other output vector are compared against using a distance function.
* [**Sigmoid Function**](https://en.wikipedia.org/wiki/Sigmoid_function) **–** Converts a sum to a probability between 0 and 1 using a mathematical function having a characteristic "S"-shaped or **sigmoid curve**.

[Logistic Function](https://en.wikipedia.org/wiki/Logistic_function): S(x) = 1 / (1 + e-x) = ex / (ex + 1)

[Hyberbolic tangent](https://en.wikipedia.org/wiki/Hyperbolic_function): S(x) = tanh x = (ex - e-x) / (ex + e-x) (shifted & scaled version of the logistic function)

* **Teacher-Student Training -** Teacher and student are trained on dataset. Student it trained using *output of the teacher* (using “soft labels”rather than “hard labels”). Teacher is outputting *class probabilities.*
* **Tensor**: N-Dimensional matrix
  + 0-Dimensional – Scalar
  + 1-Dimensional – Vector
  + 2-Dimenstional - Matrix
* **Transfer Learning** – a method where a model developed for a task is reused as the starting point for a model on a second task.
* **Weights** - How much importance or strength to apply to a particular node.

fighting?, eating?, talking?

* **Bayesien Network** (BaysNet) –
* **Learning Rate Decay** – (aka **Adaptive Learning Rate** or **Learning Rate Annealing**) –

Make large changes at the beginning when large learning rates are used and decrease learning rates later for smaller updates. i.e. training vs tuning. Two popular learning rate decay options:

* 1. Decrease learning rate gradually based on the epoch
  2. Decrease learning rate using punctuated large drops at specific epochs
* **Epoch –** One training cycle through dataset. One complete presentation of the dataset to be learned
* **Graph Learning** –
* **SGD** - (Stochastic Gradient Descent) **–** Pits are called local minima
* **Residual Network** (ResNet) Introduces Residual Learning using shortcut connections Note: ResNet-34 means a 34 layer deep neural Residual Network
* **Residual Learning:** Create learning layers using the residual of previous layers. i.e. use the subtraction of features learned from input of that layer.
* **Shortcut connections:** Directly connecting input of nth layer to some (n+x)th layer.
* **Performance Measure** – for regression problems

**RMSE** – (Root Mean Square Error) -

**MAE** – (Mean Absolute Error) -

* **Generative Query Network Architecture –** Uses two different networks to learn its way around complex virtual environments. Two Networks:

1. Representation Network: Use standard image recognition to identify what is visible
2. Generative Network: Uses Network 1 output to produce a 3D model of the environment to predict objects it can’t see

Example: AI sees a table with one leg occluded and generates a 3D representation of table with missing leg.

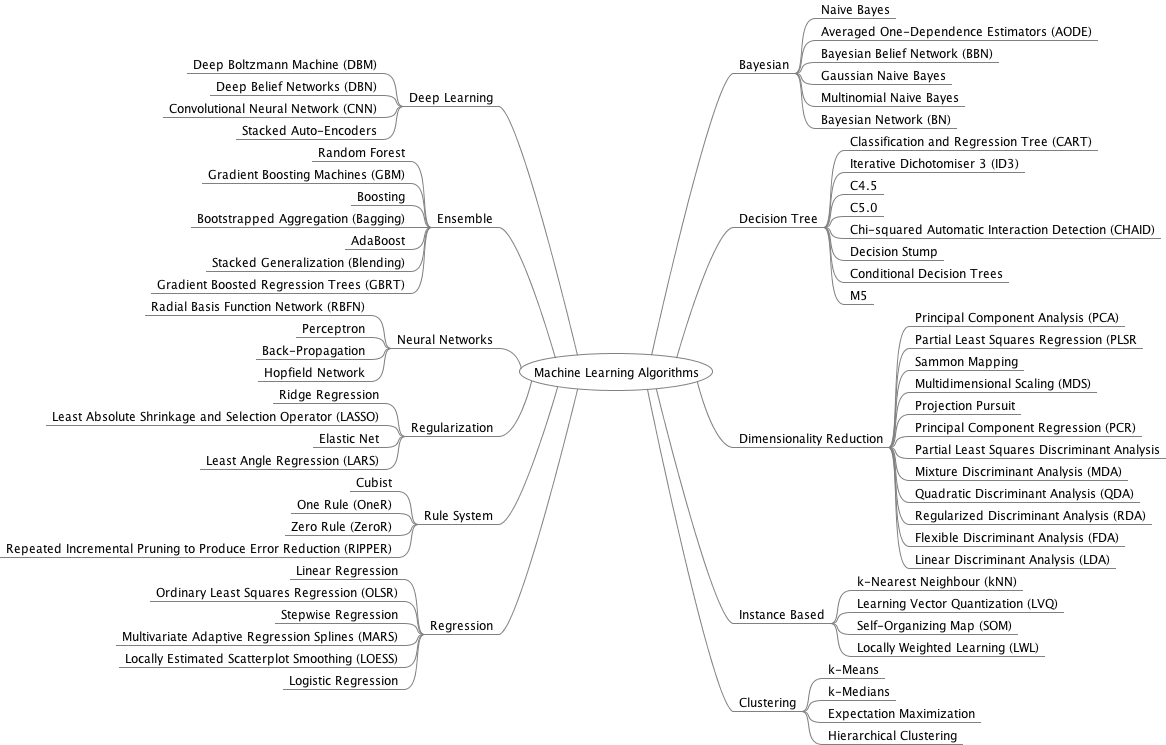
* **Graph Network training** – A Neural network that takes Graph Relational Database information as input for training to introduce inductive biasing into the neural network.

Induces a BIAS toward representing things as objects and relationships

Example: Paws, tails, whiskers *“is a part of”* A cat

Ball A & Block B *“is next to”* One another

The earth *“in in orbit around”* The Sun

**Machine Learning Algorithms:**

|  |  |
| --- | --- |
| 1. **Regression Algorithms**  * Ordinary Least Squares Regression (OLSR) * Linear Regression * Logistic Regression * Stepwise Regression * Multivariate Adaptive Regression Splines (MARS) * Locally Estimated Scatterplot Smoothing (LOESS) | 1. **Instance-based Algorithms**  * k-Nearest Neighbour (kNN) * Learning Vector Quantization (LVQ) * Self-Organizing Map (SOM) * Locally Weighted Learning (LWL) |
| 1. **Regularization Algorithms**  * Ridge Regression * Least Absolute Shrinkage and Selection Operator (LASSO) * Elastic Net * Least-Angle Regression (LARS) | 1. **Decision Tree Algorithms**  * Classification and Regression Tree (CART) * Iterative Dichotomiser 3 (ID3) * C4.5 and C5.0 (different versions of a powerful approach) * Chi-squared Automatic Interaction Detection (CHAID) * Decision Stump * M5 * Conditional Decision Trees |
| 1. **Bayesian Algorithms**  * Naive Bayes * Gaussian Naive Bayes * Multinomial Naive Bayes * Averaged One-Dependence Estimators (AODE) * Bayesian Belief Network (BBN) * Bayesian Network (BN) | 1. **Clustering Algorithms**  * k-Means * k-Medians * Expectation Maximisation (EM) * Hierarchical Clustering |
| 1. **Association Rule Learning Algorithms**  * Apriori algorithm * Eclat algorithm | 1. **Artificial Neural Network Algorithms**  * Perceptron * Back-Propagation * Hopfield Network * Radial Basis Function Network (RBFN) |
| 1. **Deep Learning Algorithms**  * Deep Boltzmann Machine (DBM) * Deep Belief Networks (DBN) * Convolutional Neural Network (CNN) * Stacked Auto-Encoders | 1. **Dimensionality Reduction Algorithms**  * Principal Component Analysis (PCA) * Principal Component Regression (PCR) * Partial Least Squares Regression (PLSR) * Sammon Mapping * Multidimensional Scaling (MDS) * Projection Pursuit * Linear Discriminant Analysis (LDA) * Mixture Discriminant Analysis (MDA) * Quadratic Discriminant Analysis (QDA) * Flexible Discriminant Analysis (FDA) |
| 1. **Ensemble Algorithms**  * Boosting * Bootstrapped Aggregation (Bagging) * AdaBoost * Stacked Generalization (blending) * Gradient Boosting Machines (GBM) * Gradient Boosted Regression Trees (GBRT) * Random Forest | 1. **Other Algorithms**  * Computational intelligence (evolutionary algorithms, etc.) * Computer Vision (CV) * Natural Language Processing (NLP) * Recommender Systems * Reinforcement Learning * Graphical Models |

**Flash Cards**

|  |  |
| --- | --- |
| AI Approaches   1. Symbolic AI 2. Connectionist AI | Unsupervised Learning   1. Associations – comes after 2. Clustering 3. Dimensionality Reduction |
| Adversarial Network   1. Generator   Discriminator | Unsupervised Learning: Associations   1. Apriori – Comes after 2. Eclat – Equivalence Class transformations |
| Ensemble Learning Algorithms   1. Bagging – Parallel Learning   Simple Voting   1. Boosting – Sequential Learning   Weighted Voting   1. Stacking – Uses Meta Classifiers or Meta regressors | Unsupervised Learning: Dimensionality Reduction   1. PCA Reduce number of variables (Principle Components) in the data 2. Kernal PCA 3. LLA – Locally Linear Reduction 4. t-SNE – t-distributed Stochastic Neighbor Embedded |
| Perceptron: Parts   1. Input Layer 2. Weights and Bias 3. NetSum 4. Activation Function | Unsupervised Learning: Clustering   1. K-Means – Group similar data in clusters 2. HCA – Hierarchical Cluster Analysis 3. Expectation Maximizations |
| Activation Function:   1. Sigmoid – Range (0 to 1) 2. Tanh – Hyperbolic Tangent.   Range (-1 to 1)   1. ReLU – Rectified Linear Unit   Range (0 to 1) Neg values = 0  or Range (0 to α)   1. Leakly ReLU – Range (-α to α) | LSTM - Long Short Term Memory: decide when to   1. Let input enter a neuron 2. Remember what was computed in previous step 3. Pass output to next time stamp |
| Overfitting   1. L2 Regularization 2. Dropout 3. Early Stopping 4. Cross Validation | Convolutional Neural Network Layers:   1. Convolution Layer 2. Max Pool Layer 3. Fully Connected Layer 4. SoftMax Layer |

**ML Algorithmic Models:**

* Linear Regression
* Logistic Regression
* Decision Tree
* SVM
* Naive Bayes
* kNN
* K-Means
* Random Forest

**Deep Learning pre-trained image recognitions Models:**

* AlexNet
* DenseNet169
* Inception\_v3
* ResNet34
* Squeezenet1\_1
* VGG13

Facial Recognition

[OpenFace](https://cmusatyalab.github.io/openface/) (system)

[FaceNet](https://arxiv.org/pdf/1503.03832.pdf) (system)

[OpenCV](https://opencv.org/) (library)

Deep Learning 10 methods:

1. Back Propagation
2. Stochastic Gradient Descent

Stochastic meaning: randomly determined; having a random probability distribution or pattern that may be analyzed statistically but may not be predicted precisely.

1. Learning Rate Decay
2. Dropout
3. Max Pooling
4. Batch Normalization
5. Long Short-Term Memory
6. Skip-gram
7. Continuous Bag Of Words
8. Transfer Learning

Machine Learning Concepts – Considerations

1. Iterate, iterate, iterate – build prototypes to test algorithms, ideas and concepts
2. Use a single evaluation metrics – makes comparisons easier between iterations
3. Error Analysis is critical – find out why you algorithm failed
4. Define an optimal error rate – (Bayes error rate) – Know when you are done

Detect when an algorithm has a high bias or variance

1. Work on problems humans can do well – e.g. speech or image recognition, classification or object detection
2. How to split dataset:
   1. Training Dataset
   2. Developers Dataset – Perform hyperparameter tuning, select & create proper features and do error analysis
   3. Test Dataset – evaluate performance of the system

Datasets:

* ImageNet
* AlexNet
* [Kaggle datasets](https://www.kaggle.com/datasets)
* [MNIST DATABASE](http://yann.lecun.com/exdb/mnist/) of handwritten digits
* [CIFAR-10](https://www.cs.toronto.edu/~kriz/cifar.html) dataset
* [Caltech-UCSD Birds-200-2011](http://www.vision.caltech.edu/visipedia/CUB-200-2011.html)
* [UC Irvine Machine Learning Repository](http://archive.ics.uci.edu/ml/index.php)
* [Amazon AWS Datasets](https://registry.opendata.aws/)
* [Caltech 101](http://www.vision.caltech.edu/Image_Datasets/Caltech101/): Pictures of objects belonging to 101 categories
* [Caltech 256](http://www.vision.caltech.edu/Image_Datasets/Caltech256/): Collection of all 30607 images 256 categories
* [DAGS](http://dags.stanford.edu/projects/scenedataset.html): stanford background database
* [PlantVillage Dataset](https://drive.google.com/file/d/0B_voCy5O5sXMTFByemhpZllYREU/view)
* [ProPublica](https://www.propublica.org/datastore/) ([link](https://github.com/propublica/compas-analysis/blob/master/Compas%20Analysis.ipynb))

Data Portals

* <http://dataportals.org/>
* <https://opendatamonitor.eu/frontend/web/index.php?r=dashboard%2Findex>
* <https://www.quandl.com/>

Dataset index:

* [Wikipedia List of datasets for machine learning research](https://en.wikipedia.org/wiki/List_of_datasets_for_machine_learning_research)
* [Quora List of public datasets](https://www.quora.com/Where-can-I-find-large-datasets-open-to-the-public)
* [Subreddit - datasets](https://www.reddit.com/r/datasets)
* [Kaggle](https://www.kaggle.com/datasets)

Machine Learning Model Image: Username: deeplearning PWD: deeplearning

* Python 3.5
* Open CV 3.2 w/ Python Bindings
* dlib 19.9
* TensorFlow 1.5 for Python 3
* Keras 2 for Python 3
* Face Recognition for Python 3
* PyCharm community edition

Additional development environment

* Python 3.7.2
* Atom
* Jupyter Notebook
* Matplotlib - is the library you’ll want if you’re going to make charts.
* Numpy - is the library you’ll need for all things mathematical.
* Pandas - is the best tool available for importing and managing datasets.
* Scipy
* Scikit-learn

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

Deep Learning Frameworks:

1. Tensor Flow w/ TF Learn API – Python – Google
2. Pytorch – Python - Facebook
3. Caffe – C++ - Facebook
4. Sonnet – Google + Tensor Flow
5. Keras – Python (for beginners) + Tensor Flow or CNTK
6. MxNet w/ Gluon– Amazon
7. CNTK (Cognitive Tool Kit) – Microsoft
8. Chainer – Open Source
9. DL4J (Deep Learning 4 Java) – Java
10. CoreML – Apple
11. ONNX – Facebook & Microsoft

Beginner – Keres

Production – Tensor Flow

Research – Pytoarch or Sonnet

Interoperability – ONNX

IOS - CoreML

AWS – MxNET

Azure – CNTK

Java – DL4J