Machine Learning

What is Machine Learning

# AI

* Machine Learning
  + Is a collection of algorithms designed to help AI make better decisions.
    - Image recognition: computer Vision
    - Sound recognition. Natural language programing
* Other

Robot -> make me cup of coffee.

1-Hear and turn the sounds into text and then understand the text and act upon it: Speech Recognition

2-To be able to see and understand what it sees. Computer Vision

3-move around and make the coffee: Reinforcement learning.

AI

Logic: all men die, I’m a man 🡪 I will die.

Machine Learning is a collection of (mathematical, programs) algorithms that helps AI in areas that it can’t solve by itself by pure programming.

Machine Learning

* Africa 🡪 Perceptron
  + Summation function
  + Activation function
    - Sigmoid
    - Leaky ReLU
    - Tanh
    - ReLU
    - Maxout
    - ELU
  + Weights
  + Inputs
* America 🡪 DATA: feature engineering
  + Structured Data
    - Numpy
    - CVS file from internet
    - Excel spreadsheet
    - Database (msql, orcl, Mysql, postgre)
  + Unstructured Data
    - Mongo DB
    - BiqQuery
    - S3 PDF Files
    - Old newspapers
* Europe 🡪 Math
  + Algebra
  + Linear Algebra
  + Calculus and differential equations
  + Other
* Asia -> Algorithms
  + Supervised (we already have the answers -lables)
  + Unsupervised
  + Reinforcement learning
  + Multilayer perceptron
  + Regression
  + Classification
  + Clustering
  + KNN
  + Baye
  + Markov
  + etc
* Australia -> Programming
  + - Which programming language
      * Python Uptown Mall
      * Java JCPenny
      * C/C++ Fashion Valley (mall)
      * Assembly MissionValley (Mall)
      * JavaScript
      * Others
    - Modules
      * Numpy
      * Pandas
      * PySpark (has built in ML modules)
      * PyTorch
      * Keras
      * Tensorflow
      * HuggingFace
      * Others
* Other

Marital Status

Married

Single

Widowed

Divorced

Annulled

Married

1, 0 (Yes, No)

Single

1, 0 (Yes, No)

Widowed

1, 0 (Yes, No)

2 + 2 = 4

2 + 3 = ?

Perceptron

Should I go out this weekend?

1. Do I have extra $200 to spend
2. Is the weather nice

X1 (amount of money I have) > 100$ 100% 🡪 Yes 1

X2 (Nice weather ) %100 Yes 🡪 1

If X1 + X2 = > 2 🡪 output is 1

W1 x X1 + W2 x X2 = 1, 0 (Summation Function) -> 1 + 1

If 0 stay home (0) , If 1 go out (1) 🡪 Activation Function

If I have over $10,000

And the bad weather then I can travel

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Perceptron 2

Machine Learning (continents):

1. Perceptron
2. Math
3. Data
4. Algorithms
5. Coding(programming)

Exercises:

Turn the following to perceptrons:

Y = mx + b

Y = mx + m2x2 + b

Y = m1x1 + m2x2 + m3x3 + b

Y = m1x1 + m2x2 + m3x3 + b

Y = m1x + m2z + m3h + b

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Math

* Algebra
  + Cartesian coordinate system
  + Y= mx +b (graph) (algorithm) Y = W0X + b 🡪 y = 3x + 2 (model)
* Linear Algebra
  + Matrix
  + Dot product
* Calculus
  + Differential Equations
  + Basic Calculus
* Probability & Statistics

3x + 3y + 2z = 8

1x – y + z = 5

8x + 5y + 7z = 90

W = [[ 3 3 2 ]

[1 -1 1]

[8 5 7 ] ]

Inputs = [x y z ]

Result = [8 5 90]

WxInputs = Results

Y = 5x + 2h + 3z

Sum notation

A = [5, 2, 3] , b = [x, h, z]

1x3 1x3

1-3 X 1-3

1-3 (T) = 3X1

A (T) . B = A1xB1 + A2XB2 + A3XB3 = 5x + 2h + 3z

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Data Continent

* Africa 🡪 Perceptron
* America 🡪 DATA
* Europe 🡪 Math
* **Asia 🡪 Algorithms**
* Australia 🡪 Programming $$

Data:

* Structured Data
  + Composed of tables (rows, columns) Primary key, secondary Key 🡪 Database
    - Microsoft SQL Server
    - MySql
    - DB2
    - Oracle
    - PostgreSQL
    - Access (msft)
    - EXCEL
* Unstructured Data
  + { “saad”: 1, “purchase”:”book” , “transactions”: { “transaction1”: “complete”, {“address”: {“streetname”: “A ST”, “number”: 123}}}
  + DynamoDB
  + MongoDB
  + BigQuery
* Data Formats
  + CSV files
  + Parque Files
  + S3 full of PDF Files
* Other

Feature Engineering

Customer list

Marital Status

* Married
* Single
* Widow
* Annulled
* Divorced
* Other

Medallion Model

Raw 🡪 Silver Bronze Gold

Data Warehouse

Data Lake

Data Mart

Data Lakehouse

Data Mesh

Data Fabric

Pipelines

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Programming

* Programming language (Shopping center/ big mall Mall of Americas)
  + Python (very easy to learn but slow)
  + Java/Scala
  + C/C++ (too many {} ) (Medium difficulty and fast or medium fast)
  + JavaScript
  + Assembly (very fast but difficult to learn)
  + Pascal
  + Fortan
  + R
  + Julia
  + Matlab
  + other
* Modules (stores, ANF, AEO, TH, TR, NKE, Addidas, )
  + NumPy
  + Pandas
  + PySpark
  + Keras
  + Tensorflow
  + PyTorch
  + Hugging Face
  + Scikit-learn
  + XGBoost
  + LightGBM
  + CatBoost
  + OpenCV (computer vision)
  + Matplotlib / Seaborn / Plotly
  + NLTK / spaCy

Start in Python 🡪 convert to C++

First select the programming language 🡪 pick the modules you need for programming

Numpy

Pandas or pyspark

Pytroch or tensorflow keras or hugging face

(visualization modules) multiplot MATLAB etc

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Algorithms

* Africa 🡪 Perceptron
* America 🡪 DATA
* Europe 🡪 Math
* **Asia 🡪 Algorithms**
* Australia 🡪 Programming $$

CAR: A big blue car

A small red boat

A medium green shirt

Blue and Yellow 🡪 green

|  |  |  |
| --- | --- | --- |
| Color | Size | object |
| Supervised | Deep vs ML | Regression |
| Unsupervised | One perceptron or multiple | Classification |
| Reinforcement | One layer vs Multiple layers of perceptrons | Clustering |
| Semi-Supervised |  | Decision Trees |
|  |  | Random Forests (many Trees) |
|  |  | k-NN: k-Nearest Neighbor |
|  |  | Naïve Bayes |
|  |  | K-Means Clustering (unsupervised) |
|  |  | Gradient Boosting Machines (XGBoost, LightGBM, CatBoost) for tabular data |
|  |  | Markov model |

Y= mx + b

Y = mx2 + m1x +b hyperbola

# Regression:

X value what is the y value?

2 bed house in zip code 92922 ?price?

X = 2 bed y= price

X = 3 bed y = price

How many inches of rain we will have tomorrow?

What is the temp tomorrow?

What is humidity tomorrow?

How many inches of snow we will have tomorrow?

# Classification?

Is there a pic of a cat in my image?

Is it going to rain or now tomorrow?

Is it going to snow or not tomorrow.

Algorithms:

* Regression (supervised) (the sweetheart of Machine Learning DL)
  + Linear
  + Logistic Regression
  + Ordinal Regression
  + Hyperbolic Regression
  + Other
* Classification
  + Bi-classification
  + Multi-classification

Naïve Bayes :

200 7th st

5 floors

Third floor 100 employees kbg export

500 people

1/5 = %20 kbg export

100/500

If every fake dollar has no wrinkles

Dollar has no wrinkles therefore there is a chance that this is a fake dollar

Markov model:

The weather tomorrow is going to be similar to the weather today +- %5.

%90

Supervised:

When data is labeled.

1000’s patients all of which I

Un-supervised I don’t have labeled data.

Reinforcement learning

ML Algorithms:

* Supervised Learning
  + Regression
    - Linear
    - Logistic
    - Ordinal
    - other
  + Decision Tree
    - Random Forest
  + SVM: Support Vector Machines
  + Naïve Bayes
  + KNN: K-nearest neighbor
  + GBM: Gradient Boosting Machines
  + Neural networks
  + AdaBoost
  + Classification
    - classification
    - Bi-classification
    - Multi-classification
* Un-supervised Learning Algorithms:
  + Clustering
  + K-Means Clustering
  + Hierarchical Clustering
  + PCA: Principal Component Analysis
  + Apriori Algo
  + Autoencoders
* Reinforcement Learning Algorithms:
  + Q-Learning
  + Deep Q-Networks DQN
  + SARSA: State Action Reward State Action
  + Policy Gradient Methods
  + Actor-Critic methods
  + Proximal Policy Optimization (PPO)
  + TRPO: Trust region Policy Optimization
  + DDPG: Deep Deterministic Policy Gradient
  + A3C: Asynch Advantage Actor-critic
  + SAC: Soft Actor critic

Programming in Machine Learning

A – Programming Language

* Python easy to learn relatively slow (unless using pyspark)
* Java relatively hard to learn med speed
* JavaScript medium med speed
* C/C++ ?? hard to learn relatively fast
* R ?
* Go ???
* Matlab ???
* C# med to learn med speed
* Julia ???
* Scala medium to learn faster
* Assembly ??? hard to learn but fastest

B – modules related to AI/Machine Learning/Deep Learning

* Numpy
* Pandas
* Pyspark (option for machine Learning)
* Tensorflow regression pants clustering jackets
* Kears
* Pytorch
* Scikit-learn
* XGBoost
* LightGBM
* CatBoost
* Hugging Face Transformers
* Keras
* FastAI
* Detectron2
* SpeechBrain
* Monai

Pants -> A- where mall, Walmart, Macy’s shopping center, B-which exact store AWF, AEO, TH, Gap, BR

Great American Mall -> GAP , Boss style, price

Visual Opencv …

LLM try to learni

Speech recognition NLTK/NLP

Computer programs

* Java
  + Numpy
  + Pyspark (very powerfull)
  + Pandas
  + Tensorflow (tensors ml) American Eagle Oufitters
  + Keras
  + PyTorch ANF Abercrombie and Fitch
  + OpenCV (vision)
  + NLTK
  + Other modules
  + Hugging face
  + Scikit-learn
* **Python**
  + Numpy
  + **Pyspark (very powerfull)**
  + Pandas
  + Tensorflow (tensors ml) American Eagle Oufitters
  + Keras
  + PyTorch ANF Abercrombie and Fitch
  + OpenCV (vision)
  + NLTK
  + Other modules
  + Hugging face
  + Scikit-learn
* C#
  + Numpy
  + Spark (very powerfull)
  + Pandas
  + Tensorflow (tensors ml) American Eagle Oufitters
  + Keras
  + Torch ANF Abercrombie and Fitch
  + OpenCV (vision)
  + NLTK
  + Other modules
  + Hugging face
  + Scikit-learn
* C/C++
  + Numpy
  + Pyspark (very powerfull)
  + Pandas
  + Tensorflow (tensors ml) American Eagle Oufitters
  + Keras
  + PyTorch ANF Abercrombie and Fitch
  + OpenCV (vision)
  + NLTK
  + Other modules
  + Hugging face
  + Scikit-learn
* R
* Matlab
* JavaScript
* Scala
* Others

What functions/methods are there in pyspark for machine learning

give me a list of pyspark machine learning methods and functions

**ChatGPT said:**

Excellent — here’s a **clean, structured list** of the most important **PySpark Machine Learning methods and functions**, organized by category so you can use it as a reference 👇

**⚙️ 1. Setup and Utilities**

| **Function / Class** | **Purpose** |
| --- | --- |
| from pyspark.ml import Pipeline | Build ML pipelines that chain data transformations and estimators. |
| from pyspark.ml.feature import \* | Import feature transformers (for preprocessing). |
| from pyspark.ml.classification import \* | Import classification algorithms. |
| from pyspark.ml.regression import \* | Import regression algorithms. |
| from pyspark.ml.clustering import \* | Import clustering algorithms. |
| from pyspark.ml.evaluation import \* | Import evaluators for performance metrics. |
| from pyspark.ml.tuning import \* | Hyperparameter tuning (CrossValidator, ParamGridBuilder). |

**📊 2. Data Preprocessing & Feature Engineering**

| **Function / Class** | **Purpose** |
| --- | --- |
| StringIndexer | Convert categorical strings to numeric indices. |
| OneHotEncoder | Encode categorical variables as binary vectors. |
| VectorAssembler | Combine multiple feature columns into a single feature vector. |
| StandardScaler | Scale features to have zero mean and unit variance. |
| MinMaxScaler | Scale features to a specific range (e.g. 0–1). |
| Binarizer | Convert continuous features to binary (0/1) based on a threshold. |
| PolynomialExpansion | Generate polynomial combinations of features. |
| PCA | Dimensionality reduction (Principal Component Analysis). |
| Imputer | Handle missing values by replacing them with mean/median. |
| Tokenizer / RegexTokenizer | Split text into words or tokens. |
| StopWordsRemover | Remove stop words from text. |
| CountVectorizer / HashingTF / IDF | Text feature extraction (TF, TF-IDF). |
| Word2Vec | Generate word embeddings. |

**🤖 3. Classification Algorithms**

| **Estimator** | **Description** |
| --- | --- |
| LogisticRegression | Binary or multinomial logistic regression. |
| DecisionTreeClassifier | Decision tree for classification. |
| RandomForestClassifier | Ensemble of decision trees. |
| GBTClassifier | Gradient-boosted trees. |
| NaiveBayes | Probabilistic classifier for text or categorical data. |
| MultilayerPerceptronClassifier | Feedforward neural network. |
| LinearSVC | Support Vector Machine classifier. |

**📈 4. Regression Algorithms**

| **Estimator** | **Description** |
| --- | --- |
| LinearRegression | Predict continuous values. |
| DecisionTreeRegressor | Regression tree model. |
| RandomForestRegressor | Ensemble regression. |
| GBTRegressor | Gradient-boosted regression trees. |
| GeneralizedLinearRegression | Generalized linear models (GLMs). |
| AFTSurvivalRegression | Time-to-event prediction (survival analysis). |
| IsotonicRegression | Monotonic regression (non-decreasing relationship). |

**🔵 5. Clustering Algorithms**

| **Estimator** | **Description** |
| --- | --- |
| KMeans | Partition data into *k* clusters. |
| BisectingKMeans | Hierarchical variant of K-Means. |
| GaussianMixture | Soft clustering using Gaussian Mixture Models. |
| LDA | Latent Dirichlet Allocation (topic modeling). |

**📉 6. Dimensionality Reduction & Feature Selection**

| **Method** | **Description** |
| --- | --- |
| PCA | Reduce features to key principal components. |
| ChiSqSelector | Feature selection based on Chi-Squared test. |
| VectorSlicer | Select subsets of feature vectors by index. |

**🧪 7. Evaluation Metrics**

| **Evaluator** | **Description** |
| --- | --- |
| BinaryClassificationEvaluator | For binary classification (AUC, accuracy). |
| MulticlassClassificationEvaluator | For multi-class classification. |
| RegressionEvaluator | For regression metrics (RMSE, MAE, R²). |
| ClusteringEvaluator | For clustering (e.g., silhouette score). |

**🔁 8. Model Tuning & Validation**

| **Class** | **Description** |
| --- | --- |
| ParamGridBuilder | Build grid of hyperparameters. |
| CrossValidator | Perform k-fold cross-validation. |
| TrainValidationSplit | Train/validation split tuning. |

**📂 9. Persistence (Save / Load Models)**

| **Function** | **Description** |
| --- | --- |
| model.save(path) | Save a trained model. |
| Model.load(path) | Load a saved model. |
| PipelineModel.save(path) | Save a full pipeline |