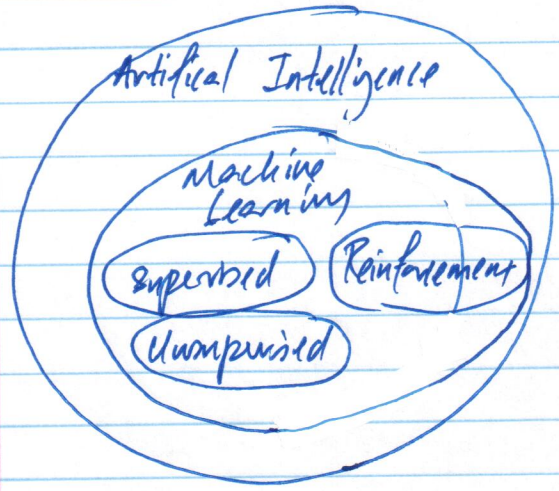
AWS Machine Learning Foundations - Scholarship

Lesson 2 : Introduction to Machine Learning



AI - computers automatically learn without explicitly programmed to do so.

ML - type of AI software development technique.

- Supervised learning
  - corresponding label ~~or~~ output
  - learn to predict output

- unsupervised learning
  - no labels or output
  - learns the underlying patterns or distributions

- reinforcement learning
  - learns which action to take in a ~~this~~ situation to maximize reward on way to reach specific goal.

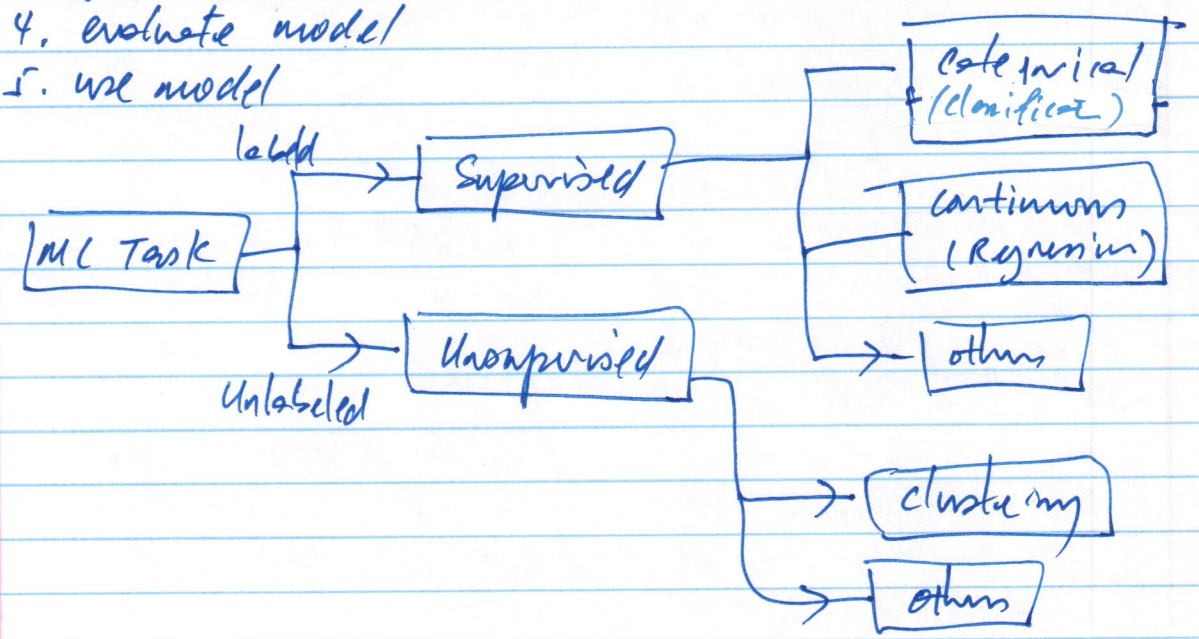
Traditional programming
- explicit programming
- covers all cases + edge cases.

ML
- adjusts model to real-world data.
- predict outcomes not part of data used in training.
- finds patterns in data that can be used to solve problem.

statistics
↓
applied math → ML ← computer science

ML components:
1. ML model
2. model training algorithm - iterative process to fit a generic model to specific data.
   - iteratively change weights in model to meet goals.
3. model inference algorithm - make predictions from trained model.

Steps:
1. define problem, specific task
2. build data set
3. train model
4. evaluate model
5. use model



Data Collection → Data Inspection → Summary Statistics → Data Visualization

Data Inspection:
- outliers
- missing values
- transformations proposed

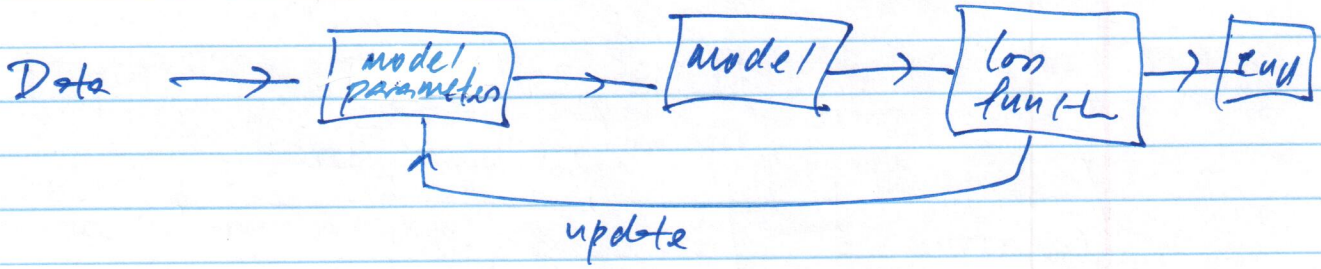
Summary Statistics:
- mean
- IQR
- stdev
- scope, scale shape

Data Visualization:
- see outliers, trends

split dataset
- training - ~80%
- test - withheld during training, test how well model performs to new data.

model training
- iteratively update model parameters to minimize loss function
  - model parameters -
    - settings or configurations that can be changed / updated to alter how model behaves.
    - weights, biases

  - loss function
    - quantify model's distance from goal.



Hyperparameters - settings of model that are not changed during training, but affects model performance + convergence.

Models:
1. Linear models
   - $y = mx + b$ linear function
   - classification - probability of being in target class

2. tree-based models
   - learn to segment / categorize by building an extremely large structure of nested if/else blocks, splitting at each block. Training determines criteria for blocks to split.

3. Deep learning models
   - collection of neurons connected together by weights. Training involves finding the values for the weights.

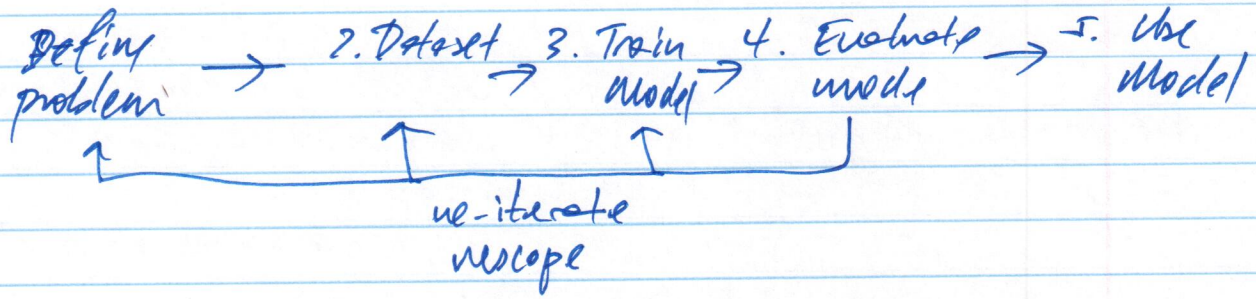
- FFNN - feed forward neural network structures neurons in series of layers, each layer containing weights to all neurons in previous layer.

- CNN - convolutional neural network
  - uses filters over grid-organized data (ie. images) video

- RNN/LSTM - recurrent neural network
  - long short term memory
  - structured to efficiently represent for loops
  - used for processing sequence data.

- Transformer - training over large datasets of data sequences.

Model evaluation metric
- accuracy - how many times got it right
- log loss - how uncertain model prediction is. or how confident model believes prediction is accurate.
- more....



Model Inference:
- monitor results
- update model / data / training

- Bag of words - counts number of times word appears in text.
- Data vectorization - converts non-numerical data into numbers.

• silhouette cluster - -1 to 1

0 = overlapping clusters

< 0 = incorrect data point assignment

> 0 = successful distinct clustering.

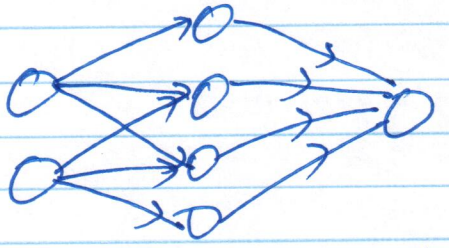
• stop words - list of words removed by NLP when building dataset.

Lesson 3: ML with AWS

Introduction to Computer Vision - AWS DeepLens.

- uses machines to detect patterns + gain high-level understanding of images + videos.

- uses neural networks



input layer | hidden layer | output layer

input: receives input data.

hidden: finds important features that have predictive power

output: generates output

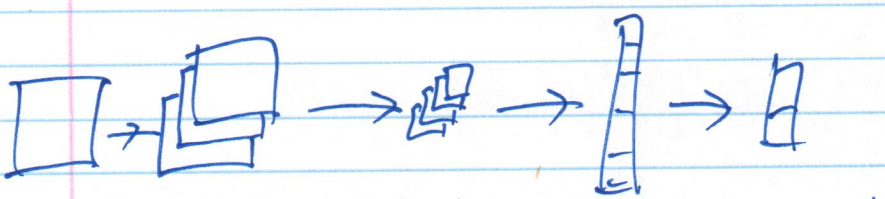


image | CNN | CNN | output layer

e.g. edge detection | feature/texture detection

feature extraction

Examples of CV:

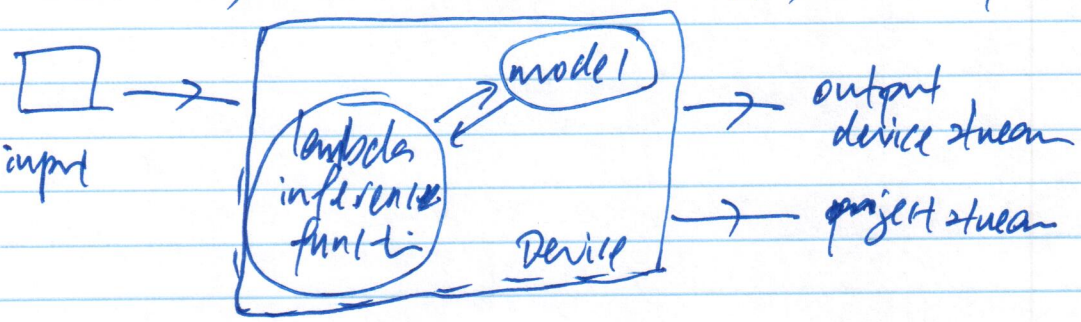
1. image classificat
   - images } content filtering
   - videos }
   - text or optical character recogniti (OCR)
   - sorting

2. object detecti
   - where in image is object of interest.
   - e.g. FaceID, autonomous driving, etc, count, different classes

3. semantic segmentati
   - pixel by pixel classificat.

4. activity recogniti
   - videos with added time dimension.
   - detect changes over time.

AWS DeepLens
- create + deploy end-to-end CV-based applicat.
- deep learning enabled camera to deploy directly on device.



Reinforcement Learning (RL)
- agent is trained to achieve a goal based on feedback it receives as it interacts with environment. Collects reward for each action, incentived. Unhelpful actions are low/no reward.

- learning objective of maximizing total cumulative reward, agent learns through trial + error one time, mapping guiding actions to situations.
  better trained the agent, more efficiently it chooses actions that accomplishes goal.

Examples of RL:
1. games
2. video game
3. real-time strategy game
4. help determine how difficult game levels should be.
5. wind turbine optimization.

Agent for
- software, training
- makes decision to reach goal

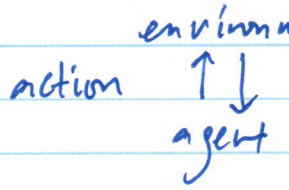
Environment
- surrounding area agent interacts in

State
- current position within environment known to agent.
- e.g. start, terminal, obstacle, etc.

Action
- for every state, agent needs to take an action toward achieving goal

Reward or feedback
- numerical reward, given an agent for each action it takes in a given state.
- reward function - incentive plan that assigns scores as rewards. based on variables.

Episode
- period of trial + error when agent makes decision + gets feedback from environment.



- Training Algorithm - model's learning objective.
  - maximize ~~reward~~ cumulative reward.
  - soft actor critic (SAC)
    - exploration
    - data efficient
    - lack stability
  - proximal policy optimization (PPO)
    - stable
    - data hungry

- Action space - set of all valid actions or choices
  - discrete action space
    - finite
  - continuous action space
    - range of values

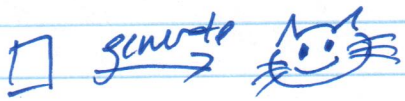
- Hyperparameters - control performance of agent during training.

- Reward Function - encourage agent to reach its goal.

Exploration - randomly wandering to explore while learning
Exploitation - use information from previous experiences to help reach its goals.

Introduction to Generative AI
- unsupervised

Discriminative:  classify → is cat? determine if cat or not.

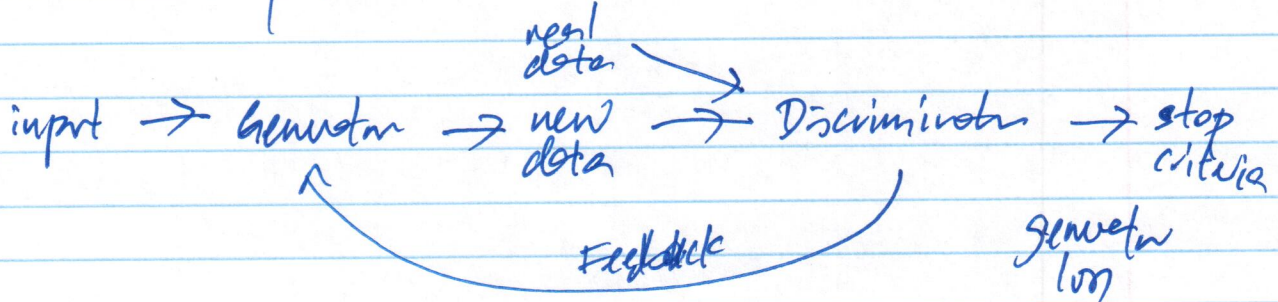
Generative:  generate → create new image of cat.

1. Generative adversarial networks (GANs)
   - pits 2 networks against each other to generate new content.
   - swaps back + forth between training a generative network (produce new data) and a discriminator network (measures how close predicted data represents training data set).

2. General autoregressive models (AR-CNN)
   - study systems that evolve over time
   - assumes likelihood of some data depends on what has happened in the past.
   - e.g. weather, stocks, ...

3. Transformer-based models
   - study data with sequential structure (e.g. words in sentence).

GAN Training Process:



generator loss
- how much deviate real data from generated data.

Lesson 4: SWE Practices, Part 1

- write clean, modular, simple code.
- Refactor when needed
- always optimize code
- always document code
- version control

Lesson 5: SWE Practices, Part 2

- unit tests
  - Pytest
    - name test file must be test_run_me.py

- logging

- code review
  - catch errors
  - ensure readability
  - standards are met

  - questions to ask
    - is code clean + modular + efficient
    - documentation ?
    - tested ?
    - logging ?

Lesson 6: Introduction to Object Oriented Programming

class - blue print consisting of attributes + methods

OOP - objects - an instance of class
- characteristics / attributes
- actions / methods

Class
- magic methods
  - __add__
  - __repr__

- inheritance
  - overide

Making a package
- distributions/
  - Gaussiandistribute.py
  - Generaldistribute.py → from .Generaldistribute import Distribute
  - __init__.py → from .Gaussiandistribute import Gaussian
- setup.py → from setuptools import setup

```
setup (name = 'distributions',
       version = '0.1',
       descript = 'Gaussian distributions',
       packages = ['distributions'],
       zip_safe = False)
```

Upload to PyPi
- license.txt
- Readme.md
- setup.cfg

} additional files

python setup.py sdist
twine upload dist/*
pip install <package-name>