**Week 1 – Introduction and Framing**

Type of ML learning

* Supervised: inputs + outputs (labels)
  + Goal: generalization
  + Types: regression (i.e. predict temperature tomorrow) and classification (i.e. predict hot or cold tomorrow)
* Unsupervised: inputs only
  + Goal: find hidden patterns
  + Clustering: technique to find subgroups in data
  + Principal Component Analysis: for high dimensionality, combined with clustering
* Reinforcement: inputs with rewards
  + Goal: learn series of actions

Functions

* Logical: deterministic outputs, logical testing that checks extreme cases
* Statistical: learned by ML from data, outputs are predictions, statistical testing that checks average case
* Models: Learned functions
* MSE: good loss function for real-valued data

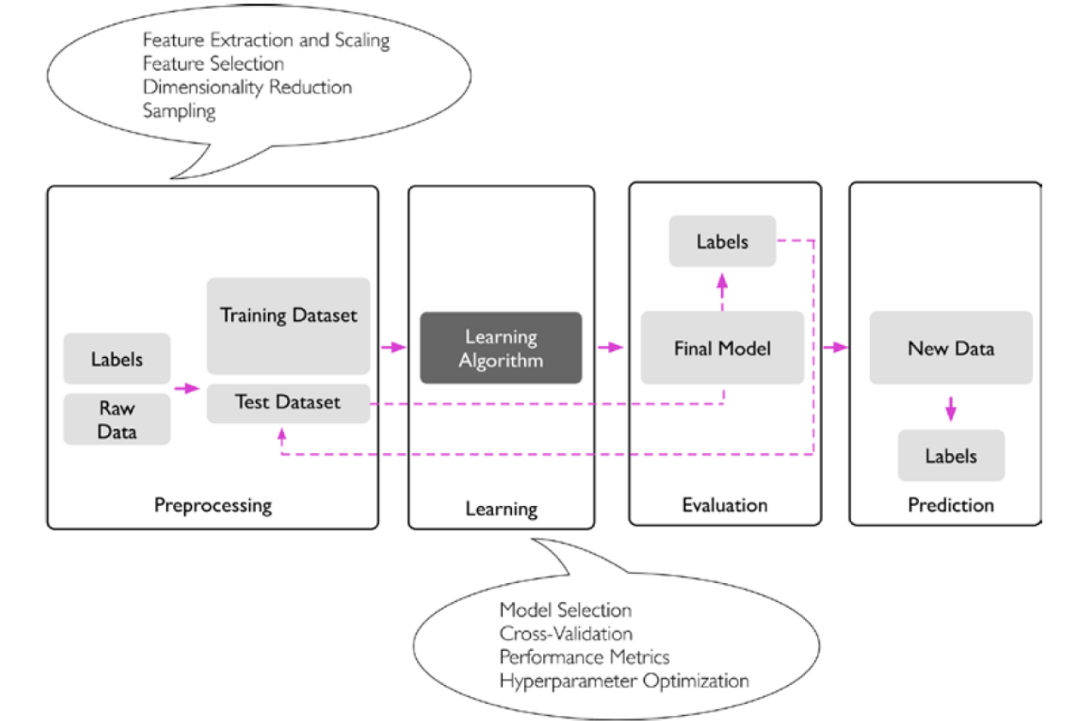
A mathematical symbols and signs

Description automatically generated with medium confidence

Generalization

* Occam’s razor: all else equal, simpler is better (avoid overfitting)

Train/test split: Don’t include info from testing into design of model



*Scale data to avoid issues with gradient descent*

**Week 2 – Linear regression / Gradient descent**

Model

* Solve LR by matrix computation or algorithm
* Parameters: weights and bias (minimize MSE to decide which is better)

Loss function



A black and white math symbol

Description automatically generated with medium confidence

Gradient descent

* Choose initial value for parameter -> adjust based on gradient -> keep updating until convergence
  + Partial derivative of each parameter stacked into vector: gradient points in direction of steepest descent
  + Parameter updates:

A black and white math equation

Description automatically generated

* Types:
  + GD: all in one batch, inefficient for large data sets, too precise (overfitting)
  + SGD: one example at a time (too much noise)
  + Mini-batch SGD: batches chosen at random, batch size is a hyper-parameter
* Epoch: single pass through all of data
  + More epochs: more overfitting
  + Number needed to reach convergence determined by learning rate
* If not convex – may get stuck in local minimum or plateau
  + Use momentum, strategic selection of starting point

**Week 3 – Features**

Multivariate linear regression

A math equations on a white background

Description automatically generated

Features

* Must be common across all types for generalization
* Feature vector: numeric even if raw data was not, row of X matrix

Feature engineering: continuous during model development/experiments

* Clean data
  + Handle missing values, outliers
* Produce numeric features (e.g. convert text to numeric)
* Combine to form new features
* Keep only useful features (lack of relationship, redundant features via correlation)
  + Bottom-up: add features one at a time
  + Top-down: remove feature one at a time
* Scale features
  + Log scaling: make nonlinear relationship linear
  + Z-score scaling: Make mean=0, variance=1 (subtract mean, divide by std) -> all features on same range
    - Can use same learning rate
  + Bucketing: complex, multimodal relationships, numeric feature -> buckets (quantiles)
* Dense vs sparse representations
  + Dense: multi-hot vector
  + Sparse: which positions are not zero
* Course of dimensionality
  + Keep number of dimensions as small as possible
  + Higher dimensions -> data spaced further apart (grows exponentially) -> more difficult to fit model

**Week 4 – Logistics regression (binary classification)**

Logistic (sigmoid) function

A math equation with numbers and symbols

Description automatically generatedA mathematical equation with a plus and minus

Description automatically generated

* z = XWT + b
* W: steepness
* b = midpoint
* Range for z: unbounded
* Range for y: 0 to 1

Accuracy metric: Correct/Total

Logistic loss (binary cross-entropy)

A black text on a white background

Description automatically generated

* Undefined when abs(y-y’) = 1
* Guaranteed to get to global minimum -> log loss is convex and differentiable
* Gradient descent same as linear regression

A group of math equations

Description automatically generated

Decision boundary (for P = ½)

A number of letters and numbers

Description automatically generated with medium confidence

**Week 5 – Logistic regression (multiclass classification)**

Confusion matrix (binary)

A white background with black text

Description automatically generated

A screenshot of a formula

Description automatically generated A white sheet with black text and red border

Description automatically generated A diagram of a curve

Description automatically generated

Precision/recall curve: tradeoff between precision/recall when using different classification thresholds

* AUC used to summarize overall performance of classifier

Multiclass Classification

* K binary classifiers: one vs. rest
* Each classifier has own parameters
* Softmax normalization for label-exclusive multiclass classifiers: scales set of scores to a multinomial probability distribution (sum of values = 1)
  + Generalization of sigmoid function

A mathematical equation with black text

Description automatically generated with medium confidence

Logistic loss (multiclass cross-entropy)

A black and white logo

Description automatically generated

* Keeps pushing model (returning gradient) until loss = 0

Linear model limitations:

A white background with black text

Description automatically generated

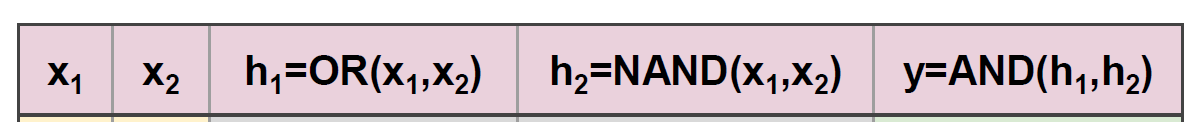
Convolution

* Approximates spatial gradient (sharp change in pixel intensity) -> edge detection

**Week 6 – Neural networks (NN)**

Logical operators: OR, AND, NAND (not and), and XOR (X1 and X2 are different)

* XOR problem: no linear model solution
* Constructing XOR:



* Training model to learn XOR: different solutions from above (local minima) due to different parameter initializations but still works

Backpropagation

* Updating layers of parameters (weights + bias) in gradient descent (1 layer for linear regression, 2+ for neural networks)
* Forward pass: making predictions

Small changes hypothesis

* Learning in ML with gradient descent only works if making small changes in parameters = small changes in output
  + Chaotic systems (i.e. weather) violate this -> butterfly effect
* Ensuring non-chaotic behavior:
  + Limited nonlinear functions (sigmoid, tanh)
  + Differentiable (smooth) loss
  + Each unit = function of units in previous layer (builds on itself)

Chain rule

A close up of a sign

Description automatically generated

Activation functions (non-linear)

A graph of function and function

Description automatically generated with medium confidence

* ReLU derivatives: 0 and then 1

Optimizers

* Examples: SGD, learning rate schedules, momentum, Adagrad, AdamA diagram of a loss

  Description automatically generated

Hierarchical organization similar between NNs and brains:

* More neurons for lower-level layers (simpler, diagonal lines)
* Less neurons for higher-level layers (complex)

Validation set

* 20-30% of training data (80% of entire set)
* Used for selecting best hyperparameters during training
* Merge with rest of training data to train final model before testing
* K-fold cross-validation on small datasets