

1 Introduction

Artificial Intelligence

- broad concept
- different interpretations
- we do not have a definition of intelligence

Statistical machine learning

- Algorithms and applications where computer learn from data
- AGI
- Artificial General Intelligence
  - Hypothetical computer program that can perform intellectual tasks as well as, or better than a human.

Turing Test

- Also called imitation game
- Tests of a machine's ability to exhibit intelligent behaviour equivalent to, or indistinguishable from that of a human
- Has some philosophical problems (Complex problems, humans cant solve / AI must learn to lie)

Examples of application (today)

- Personalization of news feeds
- Product searching and recommendation s on eCommerce platforms
- Voice-to-text
- Predictive maintenance

Bias

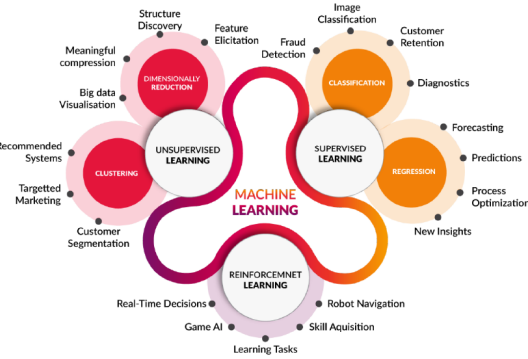
- Results that are systematically prejudiced due to faulty assumptions
- The inability for a machine learning method (like linear regression) to capture the true relationship, eg. Straight Line can't be curved like the true relationship

Variance

- Difference of fits between data sets.
- The difference in fits between training and testing set (different data sets in general)
- Low variance Sum of Squares are very similar for different datasets

The ideal ML algorithm has low bias and can accurately model the true relationship and it has low variability by producing consistent predictions across different datasets.

1.1 Tasks and Algorithms of Machine Learning



1.2 Dialogflow

**Intents** An intent categorizes an end-user's intention for one conversation turn.

- Recognizes the need of a user
- Require training to match to user inputs
- Follow up Intents (on Success)
- Fallback Intents (on Failure)

**Entities** Each intent parameter has a type, called the entity type, which dictates exactly how data from an end-user expression is extracted.

- Extract information from user inputs
- Help to identify required intent
- **System Entities** Date and time / Numbers / Amounts / Units / etc.
- **Developer Entities** defined by list of words (@pizza-type / @drink / etc.)
- **User Entities** transient, temporary Information based on Conversation (@previous-orders)

Dialog

- **Linear** Gather a list of information
- **Non Linear** Using Contexts

Context

- Each Intent can have Input & Output Context
- Intents are active based on active Context
- Expire automatically

Fulfillment

- Action triggered on fulfilled Intents
- e.g. Webhook

**Predictive modeling** Train model for predictions

Feature Engineering

- The process of identifying useful, additional input from the data
- A typical preprocessing step before the actual learning process starts

**Deep neural network** ANNs with multiple hidden layers

**Feature** e.g. Years of working experience, school grades

1.3 7 Steps of Machine Learning

1. **Gathering data** Collect quantity/quality data for training/testing
2. **Preparing that data** Cleanup data (remove duplicates, correct errors, deal with missing values, normalize data, convert data types)
3. **Choosing a model** Select the right algorithm(s)

4. **Training** Train the model, each iteration of process is a training step
5. **Evaluation** Use metrics to measure objective performance of the model, test model against previously unseen data, good train/e-val split is 80/20, 70/30
6. **Hyperparameter tuning** Try to improve upon the positive results achieved during the evaluation through gamble with stepNumber of training steps, learning rate, initialization values and distribution
7. **Prediction** Model should be ready for practical applications

2 Natural Language Processing (NLP)

- Automated processing of human language (written & spoken)
- Aims to understand and generate human (natural) language
- Understanding spoken text is still difficult
- Understanding written text became BIG business (search-engines)
- Generating human-like conversations is still very hard

2.1 4 Ingredients of Machine Learning

1. Data

- Dataset
- Pre-Processing Pipeline including cleansing, feature-engineering, data augmentation etc.

2. Cost-Function (Loss)

- Formal mathematical expression for good / bad
- Commonly **Mean Squared Error (MSE)**

3. Model

- From linear model:  $\hat{y}_i = ax_i + b$
- To complicated million parameter neural networks
- Different tasks require different models (regression / decision tree)

4. Optimization Procedure

- Algorithm that changes the parameters of the model that the cost-function is minimized.
- E.g. Stochastic Gradient Descent (SGD), ADAM, RMSProp...

For successful ML, there are many more ingredients ...

5. Performance optimization

- Building of efficient pipelines
- Following tool specific recommendations

6. Visualization and evaluation of the learning Process

- Learning curves
- Performance measures
- Tensorboard

7. Cross-Validation & Regularization

- Train models that generalize well to unseen data
- Estimate the generalization error

2.2 Representation of Words

Vectors can be used to represent words based on their meaning.

2.2.1 One-hot representation

- Vector with a single 1-Value
- All other Values are set to 0
- Count the Number of different Words, Define one unique vector per word

Dini Mom isch fett.

Dini:	$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	Mom:	$\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$	isch:	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$	fett:	$\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$	':	$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$
-------	--	------	--	-------	--	-------	--	----	--

Disadvantages

- Very high dimensional vector space (1 Dimension / unique Word)
- Sparse Representation: Each vector has a single 1 and N Zeroes. (Memory Inefficient)
- No Generalization: All words are unrelated to each other.
- Does not capture any aspect of the meaning of a word

2.2.2 Indexing

Make a list of words (optionally alphabetically). Use the index to represent each word.

Example:

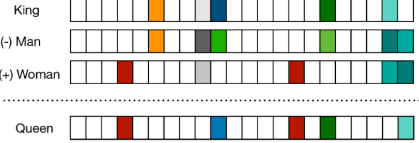
Dini Mom isch fett.

Dini: 0, Mom: 1, isch: 2, fett: 3, ':': 4

- Dense Equivalent of one-hot encoding
- Indexes are not more useful than one-hot vectors
- Often used as preprocessing step
- Indices / One-Hot Vectors are fed into a network which learns more useful representations

2.2.3 Distributed Representation

- vectors that capture (at least partially) the semantics of a word
- Words that occur in similar contexts (neighboring words) tend to have similar meanings
- Similar words share similar representations
- Distributed representations can be learned



Words to Vectors

- Mathematical function maps word to high dimensional Vector
- In neural networks, this function is implemented in the Embedding Layer

Advantage (of vectors)

- Good embedding maps similar/related words to similar regions of the vector space (nearby words have a semantic similarity)
- Dot-Product (Skalarprodukt) is a measure of similarity
- Possible to add/subtract vectors

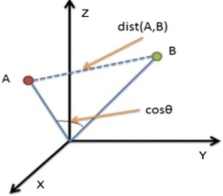
**Calculate Similarities between words** Dot-Product (Skalarprodukt) of 2 Vectors is

- maximal when parallel (0°), both vectors with norm 1 results in max value 1
- zero when orthogonal (90°)
- minimal (negative) when opposite directions (180°) both vectors with norm 1 results in max value -1

Cosine Distance

- Way to calculate how similar two words (vectors) are

cat: A, dog: B



Example  
A: (3, 6, 2, 1), B: (2, 7, 2, 0)  
 $A \cdot B = 6 + 42 + 4 + 0 = 52$

$$||A|| ||B|| = \sqrt{(9 + 36 + 4 + 1)} * \sqrt{(4 + 49 + 4 + 0)} = 53.38539$$

$$A \cdot B / (||A|| ||B||) = 0.9740$$

high value equals high similarity (to be an animal)

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}}$$

3 Probability

3.1 Random Variables

- Values depend on outcomes of a random phenomenon
- Random variable  $X$  is a variable that takes a numerical value  $x$ , which depends on a random experiment
- **Discrete**  $X$  takes any of a finite set of values 1.5, 2.123, 6.2, 10
- **Continuous**  $X$  takes any value of an uncountable range e.g. real numbers from an interval

Best we can know

- All possible values
- Probability of each value

E.g. The discrete random variable  $X$  is the number observed when rolling a fair dice.  
 $P(X = x) / P(x)$ : 1/6 for each possible value

Joint Probability

- Joint Properties of two random variables
- Defined by the Joint Probability Mass Function

E.g. Dice1 = 5 AND Dice2 = 4  
 $P_{XY}(5, 4) = P_X(5) * P_Y(4) = 1/6 * 1/6 = 1/36$

	X=1	X=2	X=3	X=4	X=5	X=6
Y=1	1/36	1/36	1/36	1/36	1/36	1/36
Y=2	1/36	1/36	1/36	1/36	1/36	1/36
Y=3	1/36	1/36	1/36	1/36	1/36	1/36
Y=4	1/36	1/36	1/36	1/36	1/36	1/36
Y=5	1/36	1/36	1/36	1/36	1/36	1/36
Y=6	1/36	1/36	1/36	1/36	1/36	1/36

Independant random Variables

- Joint Probability is the product of the individual probabilities

$P(X, Y) = P(X) * P(Y)$  (only if independant)  
 $P(X, Y, Z) = P(X) * P(Y) * P(Z)$  (only if independant)

Correlated random Variables

- There are events that are not independant
- Such random variables are correlated
- X: observe clouds (0=no, 1=small, 2=big)
- Y: observe rain (0=no, 1=light, 2=moderate, 3=heavy)

Conditional Probability

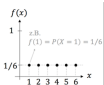
- One variable is no longer random
- X is observed, its value is fixed
- Calculate the probabilities of Y given X:  $P(Y|X)$

$P(X, Y) = P(X|Y) * P(Y)$   
 $P(X, Y) = P(Y|X) * P(X)$   
 $P(Y|X) = \frac{P(X, Y)}{P(X)}$   
**Bayes Rule**  
 $P(X|Y) * P(Y) = P(Y|X) * P(X)$   
Therefore  
 $P(Y|X) = \frac{P(X|Y) * P(Y)}{P(X)}$

3.2 Probability mass function (PMF)

Wahrscheinlichkeitsfunktion, a function  $f(x)$  that provides the probability for each value  $x$  of a discrete random variable  $X$

Graph of a PMF



4 Data Visualization

- See trends, clusters and local patterns in data
- Difficult to see in raw data
- Detect outliers and unusual groups
- Validate Hypothesis/Conjecture/Theory

Important in a Plot

- **X-Axis labels** which data is represented and its units
- **Y-Axis labels** which data is represented and its labels
- **Title**
- **Scale** linear, logarithmic
- **Dimensionality of the data** 2D / 3D

**Dataframe** a two-dimensional labelled data structure with columns of different types

4.0.1 Data Analysis Libraries

NumPy

- Package for scientific computing in Python
- Multidimensional array object
- Routines for fast array operations (sorting, selecting, FFT, linear, ...)

pandas

- Built on top of NumPy
- Routines for accessing tabular data from files (.csv, xls, etc.)
- Supports 2-dimensional data (dataframe and series)
- Dataframes are something like database tables

Matplotlib

- Library for visualizing data
- Provides bargraphs, histograms, piecharts, scatter plots, lines, boxplots, heatmaps, ...

Seaborn

- Extension of Matplotlib, NumPy and pandas
- More user friendly
- Plots are aesthetically better

4.0.2 Line Plots

- Bivariate, Continuous
- Recognizes trend (pattern of change) (over time)

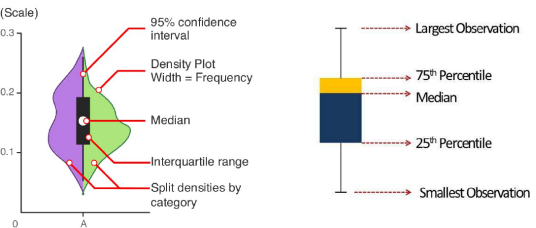
4.0.3 Bar Chart

- Used for categorical data
- Counting based on each category

4.0.4 Histogram

- Represents the empirical distribution of a variable
- Automatically creates bins (interval) along the range of values
- Shows vertical bars to indicate the number of observations per bin

4.0.5 Descriptive Statistics: Box Plots and Violin Plots



4.0.6 Scatter Plot

- Relationship between (two) continuous variables
- Helps to get an idea of the degree of correlation between variables

5 Linear Regression

a simple method to analyse data

- Only considers a linear relationship between input and output
- In the simplest case,  $x$  and  $y$  are scalars and the linear model therefore has only two free parameters
- The goal is to identify  $a$  (slope) and  $b$  (intercept) for which the linear model best explains the data

$$\hat{y}_i = ax_i + b$$

Applications

- **Interpretation** has some input an effect on the output, eg. Is there a relationship between smoking cigarets and the risk of lung cancer?
- **Prediction** Given some sensor data like oil pressure, temperature ..., eg. a model could predict (and thereby hopefully prevent) an engine failure.

5.1 Model

In ML, we use the term **model** for any mathematical function that explains the data

$$y_i \approx f(x_i)$$
$$y_i = f(x_i) + \epsilon_i$$

where  $\epsilon_i$  is unexplained noise. It is often assumed that  $\epsilon_i$  follows a normal distribution.  
Instead of approximating  $y_i$ , we calculate an **estimate**  $\hat{y}_i$  (y hat) of the usually unknown  $y_i$ :

$$\hat{y}_i = f(x)$$

5.2 Mean Squared Error (MSE)

- Loss we want to minimize
- Usually divided by 2

$$\hat{y}_i = ax_i + b$$
$$\epsilon_i = y_i - \hat{y}_i$$

The difference  $\epsilon_i$ , called residual

$$E = \frac{1}{2N} * \sum_{i=1}^N \epsilon_i^2$$
$$E = \frac{1}{2N} * \sum_{i=1}^N (y_i - (a * x_i + b))^2$$

5.3 Correlation and-Causality

- Correlation is not causality
- Correlation refers to the degree to which a pair of variables are **linearly related**
- Linear regression is a tool to detect correlations between two or more variables
- Correlation can be quantified using the Pearson correlation coefficient

6 Optimization

- Training or learning in AI often suggests an algorithm performing some sort of optimization
- It is the problem of finding a set of inputs to an objective function that results in a maximum or minimum function evaluation
- In our examples the objective is to minimize the loss function

6.1 Gradient Descent

At any location [a,b] we look at the error-gradient in the neighbourhood of [a,b] and move a (small) step in the direction where the error shrinks the most. By repeating this procedure, we will eventually arrive at the location where the error is smallest.

- Iterative Method/Procedure
- Each iteration, the model parameters are updated such as that the Loss (MSE) is reduced
- Move along a trajectory which includes fewer points
- At each point of the trajectory we evaluate the gradient of the error function
- At each iteration, we would have to iterate over all  $N = 1'000$  points to calculate the gradient of the loss function.

Calculate Gradient

$$\text{Gradient of } E = \begin{bmatrix} \frac{\partial E}{\partial a} \\ \frac{\partial E}{\partial b} \end{bmatrix}$$

Calculate these two partial derivatives

$$\frac{\partial E}{\partial a} = \frac{1}{N} \sum_{i=1}^N (y_i - (a \cdot x_i + b)) \cdot -x_i$$

$$\frac{\partial E}{\partial b} = \frac{1}{N} \sum_{i=1}^N (y_i - (a \cdot x_i + b)) \cdot -1$$

$$\text{Gradient of } E = \begin{bmatrix} \frac{\partial E}{\partial a} \\ \frac{\partial E}{\partial b} \end{bmatrix} = \begin{bmatrix} \frac{1}{N} \sum_{i=1}^N (y_i - (a \cdot x_i + b))(-x_i) \\ \frac{1}{N} \sum_{i=1}^N (y_i - (a \cdot x_i + b))(-1) \end{bmatrix}$$

6.2 Stochastic Gradient Descent (SGD)

we do not need the exact gradient to find a trajectory toward the minimum. Instead, at each iteration we can randomly pick a few datapoints and use them to calculate an approximation of the gradient.

- At each iteration, the gradient is calculated on a (randomly selected) subset of the data
- For a fixed learning rate, SGD does not converge

Mini-batches

- $1 < n < N$
- Increasing the batch-size will reduce the variance of the gradient estimation
- batch-size  $n = 1$  yields a very noisy gradient
- batch-size  $n = N$  is expensive to calculate
- often mini-batches of size  $n = 32$  or  $n = 64$  are used

Annealed SGD

- The learning rate  $\alpha$  is reduced over time
- This is called (simulated) annealing
- There are different options (called schedules) how to reduce  $\alpha$  over time
- A fixed learning rate  $\alpha$  does not converge. The algorithm keeps fluctuating around the minimum. Annealed SGD solves this apparent contradiction by adapting the learning rate. It starts with a large  $\alpha$  and reduces it over time

6.2.1 General remarks on SGD

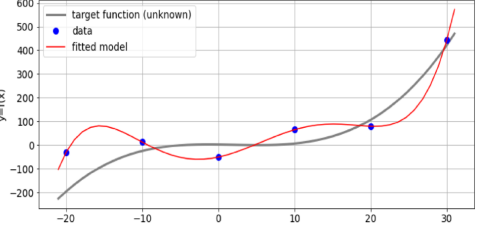
- Gradient-based methods only work if we can express a Loss function as a differentiable function
- SGD is dealing with only a single datum at each iteration. This is very inefficient and rarely used.
- Batch- or mini-batch gradient-descent is usually used

7 Generalization and Regularization

- **Out-of-sample Error** Generalization Error (Test Error) is the MSE of new Data
- A good model has a low Generalization Error

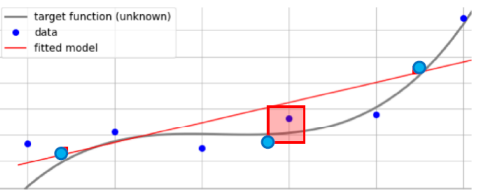
7.1 Overfitting

- A model that perfectly fits the data does not have to be perfect
- In-Sample Error (Training error) was minimized (MSE = 0)
- Overfitting happens if the MSE of Training Error is small thanks to a complex model but the Generalization Error is large (Good with training data, bad with testing data)



7.2 Underfitting

- Using a too simple model
- In-Sample Error is large
- Generalization Error is large



7.3 Training-Set, Test-Set, Model Evaluation

- The Generalization Error can't be calculated, but estimated
- Split the data into 2 sets
  - Training-Set ( 80% of data)
  - Test-Set ( 20% of data)

Training

- Fit the model to the training set
- This minimizes the in-sample error

Evaluating

- Using the Test-Set
- Produces the **Test-Error**
- This is an estimate of the Generalization Error

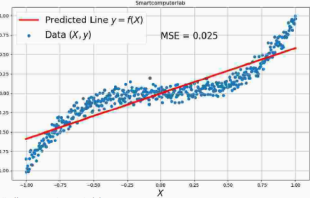
7.4 Bias-Variance Trade-off

High Bias

- A too simple model for the given data

Low Variance

- The model is relatively stable
- Very similar model if trained with new data

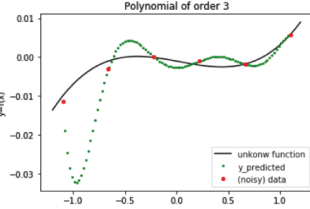


Low Bias

- A more complex model can better explain the data

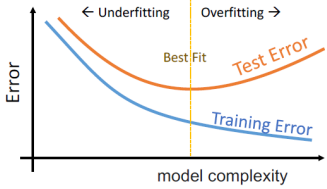
High Variance

- Given a new datapoint, the MSE can be very large
- For a different set with more datapoints, the model may be very different



7.4.1 Trade-off

- Higher bias implies lower variance
- Lower bias implies higher variance
- In practice, all we want is low variance
- The model can only be as complex as the data permits
- You have to find an optimal balance between bias and variance



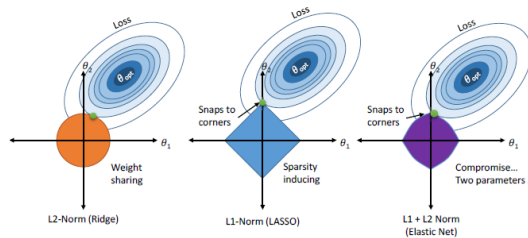
7.5 Regularization

Technique to measure the model complexity

- L1 - Norm  $\sum_{j=1}^p |\beta_j|$
- L2 - Norm  $\sum_{j=1}^p \beta_j^2$

Technique to control the model complexity

- Add a penalty term to the Loss (optimization problem)
- More complex models get a higher penalty
- Add a constrain to the optimization process
- Modified optimization error target  $regularized\ loss = MSE + \lambda \cdot model\ complexity$



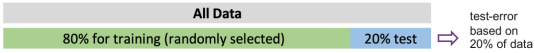
8 Cross-Validation (CV)

- a technique to compare (and select from) different models (=different parameter values)
- Use case 1: Obtain a better estimate of the generalization error
- Use case 2: Selection of hyper parameters, split/train pattern of cross validation can be used to find optimal hyper parameters

Problem with (80/20) Data Separation

- Test Error depends on random set
- For different sets, the test error would be different

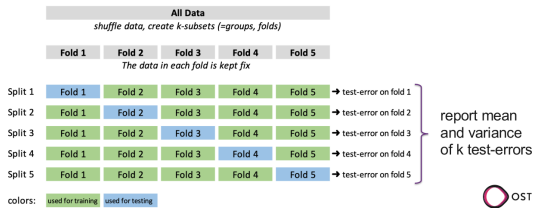
without cross-validation



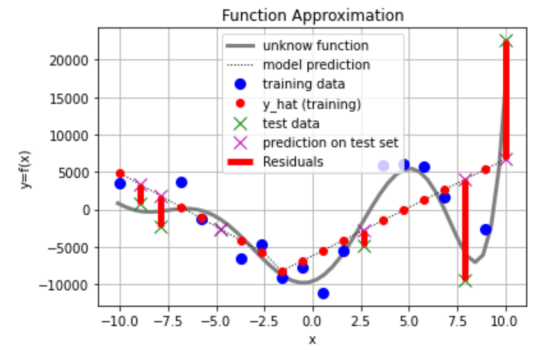
8.1 k-fold Cross-Validation

With k-Fold Cross-Validation

- The data is split once into k folds
- Repeats the split-train-test procedure k times, using a systematic resampling procedure
- Then train/test is repeated k-times.
- Each fold participates in k-1 training phases and is used once for testing



- Typical Values for k are 5,10 or N
- The data of a fold does not change during procedure
- Do not preprocess the whole dataset
- Apply the preprocessing pipeline (standardization) to each split
- Each split generates a different model
- With regularization, each split may yield a different model and a different optimal  $\lambda$

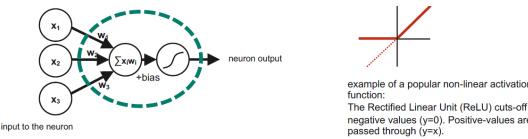


9 Artificial Neural Networks (ANN)

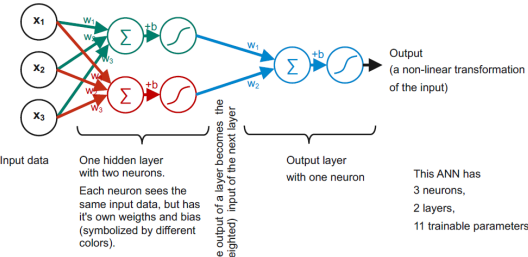
An ANN is a data-structure to define arbitrarily complex mathematical functions

9.1 Artificial Neurons

- Receives an input vector  $[x_1, x_2, \dots]$
- Each neuron has its own input weights  $[w_1, w_2, \dots]$  and bias b (=intercept)
- Calculates the sum of the weighted input (dot product  $\vec{x} * \vec{w}$ ), adds a bias b, and passes it through a nonlinear activation function



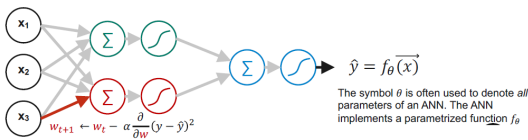
9.2 Simple Artificial Neural Network (ANN)



9.3 Training an ANN

Supervised learning

- Data with label
- For each input  $\vec{x}$  we are given the output  $\vec{y}$
- ANN is initialized with random weights
- An optimizer reduces a cost-function (e.g. MSE)
- At every iteration, and for every single weight  $w$  and bias  $b$ , the partial derivative needs to be calculated. (Backpropagation algorithm)





13 Clustering

13.1 Unsupervised Learning

- We are given Data (features, x) without labels (y)
  - It learns something through the structure of the data
  - The goal of unsupervised learning is to self-discover patterns from the data
- Clusters
- Data points which have shared properties
  - Fall into one cluster or one alike group
  - Similar Data Points are close together
  - Group  $n$  data points into  $k_C$  number of clusters

- Applications
- Social Network Analysis
  - Astronomical Data
  - Market segmentation
  - Recommendation systems

13.2 Naive K-means

1. Let us assume we know the number of clusters  $k_C$
  2. Initialize the value of  $k$  cluster centres (aka, means, centroids) ( $C_1, C_2, \dots, C_{k_C}$ )
  3. Assignment :
    1. Find the squared Euclidean distance between the centres and all the data points.
    2. Assign each data point to the cluster of the nearest centre.
  4. Update: Each cluster now potentially has a new centre (mean). Update the centre for each cluster
    1. New Centres ( $(C'_1, C'_2, \dots, C'_{k_C})$ ) = Average of all the data points in the cluster (1, 2, ...,  $k_C$ )
  5. If some stopping criterion met, Done
  6. Else, go to Assignment step 3
- Initialization: choose  $(C_1, C_2, \dots, C_{k_C})$ . ( $C_{i..}$  are vectors in the same space as the features. They can be randomly initialized)

- Assignment Step:
- $$V_{k+1}^U V_{k+1}^U d_{ik} = (x_i - C_k)^2 \quad (\text{squared euclidean distance})$$
- $$d_{i,k}^{min} = \text{Minimum}(d_{i,1}, d_{i,2}, \dots, d_{i,k_C})$$
- $x_i \in \text{Cluster } k$
- Update Step: (centre, = mean of all data in cluster,)
- $$V_{k+1}^U C_k = \frac{1}{\text{size}(C_k)} \sum_{x_i \in \text{cluster } k} x_i$$

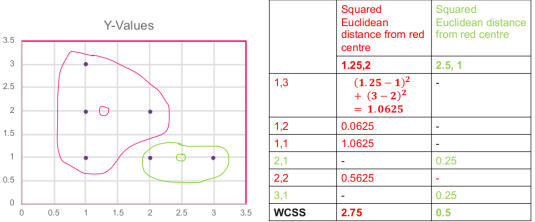
- Stopping Criterion
- When centres don't change (time consuming)
  - The datapoints assigned to specific cluster remains the same (takes too much time)
  - The distance of datapoints from their centres  $\geq$  threshold we have set
  - Fixed number of iterations have reached (choose wisely)
- Initialization
- Performance depends on the random initialization
  - Some seeds can result in a poor convergence rate
  - Some seeds can converge to suboptimal clustering
  - If centres are very close, it takes a lot of iterations to converge
  - Initialize randomly, run multiple times
- Standardization of data
- Features with large values may dominate the distance value
  - Features over small values will have no impact
  - Normalize values!

13.3 Sklean k-means

- Initialization
- Init = K-means++
  - Only initialization of the centroids will change
  - Chosen centroids should be far from each other
- max\_iter
- Number of iterations before stopping
- n\_init
- Number of time the k-means algorithm will be run with different centroid seeds

13.4 Evaluate Cluster Quality

- Make clusters so that for each cluster the distance of each cluster member from its center is minimizes
- Inertia or within-cluster sum-of-squares (WCSS)
- Sum of squared distances of samples (each point) to their closest center
  - As small as possible



- Silhouette Score
- How far the datapoints in one cluster are from the datapoints in another cluster
  - Silhouette Score of a point:  $\frac{b-a}{\max(a,b)}$
  - a: average intra-cluster distance (distance between each point within)

- b: average inter-cluster distance (distance between a cluster and its nearest neighbour)

14 Ensemble Methods

- Suppose we have many different weak models (better than random)
  - Get prediction from all of them and take a vote
  - Class with most votes is the predicted class
  - Commonly used towards the end of a project
  - Requirement enough models / diverse models
- Wisdom of Crowd
- Suppose you have a difficult question
  - Ask many people and aggregate the answer
  - This might work very well instead of finding the best suited person
- Transfer
- Wisdom of Crowd can be applied to ML
  - Instead of finding the best model, aggregate the results of weak models
  - Aggregate predictions of regressors or classifiers
  - Might get better accuracy than the best predictor
  - Ensemble: group of predictors
- Aggregate predictions
- Hard voting Predict class with most votes
  - Soft voting Predict class with the highest class probability

14.1 Bagging and Pasting

- Bagging (Bootstrap Aggregating)
- Sampling with replacement
  - Allows data points to be used several times



- Sampling without replacement

14.2 No free lunch theorem

No single machine learning algorithm is universally the best-performing algorithm for all problems

14.2.1 Out of Bag (oob) Evaluation

- Using Bagging
- Some Data Points may not be used at all
- Use them for evaluation

Silhouette Score

- Silhouette(1,3) =  $\frac{(3.65 - 1.47)}{3.65} = 1 - \frac{1.47}{3.65} = 0.59$
- Silhouette(2,2) =  $\frac{(1.2 - 1.27)}{1.27} = \frac{1.2}{1.27} - 1 = -0.05$
- The range of silhouette score is from -1 to 1.
- Score should be closer to 1 than -1.

	Distance from points in the GREEN cluster		
	2,1	3,1	b
1,3	√3	√8	3.06
1,2	√2	√3	1.8
1,1	1	2	1.5
2,2	1	√2	1.2

	Distance from other points in the cluster				
1,3	1,2	1,1	2,2	a	
1,3	-	1	2	√2 + 1.41	1.47
1,2	1	-	1	1	1
1,1	2	1	-	√2	1.47
2,2	√2	1	√2	-	1.27