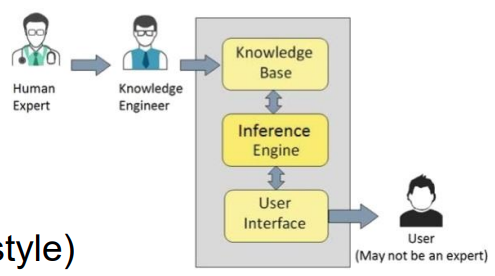
**Machine Learning Summary WS 2021/2022**

The resulting document is not a complete summary of the lecture. Information may be incorrect. Especially later sections like explainable AI and SVMs are not complete. It may also be more advisable to learn the exam questions as they do not seem to change much from year to year.

**1st Lecture**

**What is the purpose of a rule-based system?**

* Making deductions or choices based on rules



**What are components of a rule based system? What do these components do?**

* Knowledge base: facts & rules
  + IF → THEN style (ex. IF hot and smoke THEN fire)
* Inference engine
  + applies rules to make new deductions
  + Forward chaining: assert new facts
  + Backward chaining: start with goal → Determine which facts have to be asserted

**What is a central problem of rule-based systems? What is the solution?**

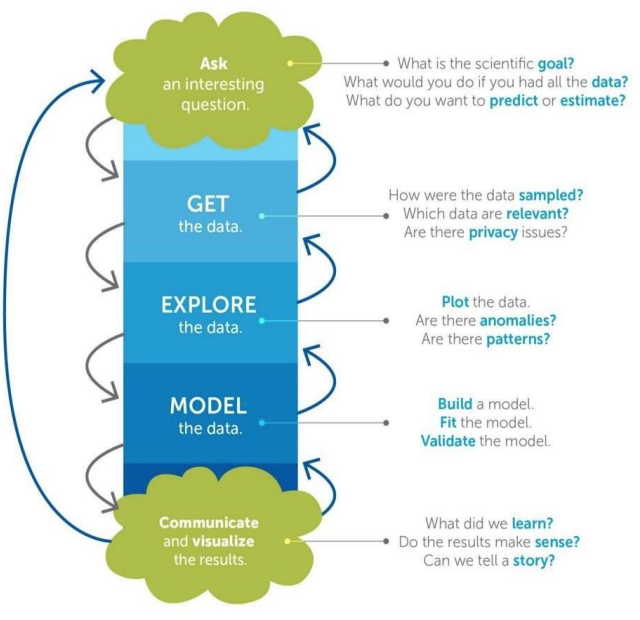
* Rules often created manually by expert
* Scales poorly

→ Machine Learning automatically creates rules

**Name 3 common sub-disciplines of machine learning and explain them.**

* Unsupervised Learning
  + attempt to discover structures in the data
  + associated with data mining
  + no predefined classes or label structure
* Supervised Learning
  + data divided into input and output
  + examples are regression and classification
* Reinforcement Learning
  + agent (i.e. the learning solution) takes actions in a (simulated) environment and is rewarded based on a performance metric
  + example: machine learning algorithm playing mario or snake

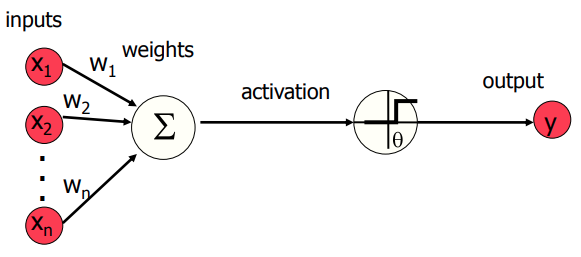
**Outline and explain the data science process?**



**2nd Lecture**

**Describe Perceptrons? 3 Facts at least!**

* Multiple inputs X = x1, x2, …. ,xn
* Linearly combines inputs using weights (includes intercept)
* Activation function at the end results in boolean output

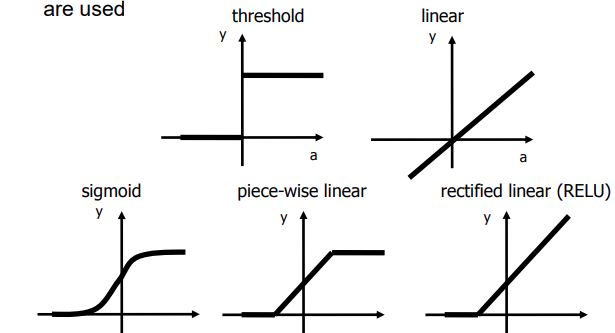


**Why would you care about Perceptrons today?**

* Building block for Neural Networks

I decided to skip the “algorithm” underlying the fitting of the perceptron as it is outlined really roughly in the slides.

**Name different activations Perceptrons (and for that matter Neural Networks) can use.**



**Describe KNN**

* is named after its most important parameter “k”
* predicts by evaluating the distance between a given data point and its “k” nearest neighbors → prediction is formed based on these neighbors (f.e. majority voting)
* lazy learner (computation is done during prediction)
* can use different distance measures

| Euclidean | Manhattan | Linfinity | Cosine |
| --- | --- | --- | --- |
|  |  |  |  |

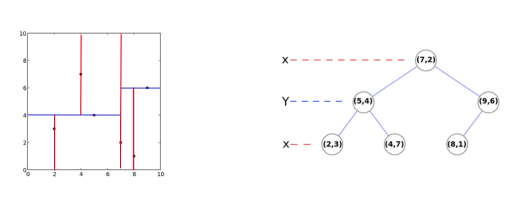
**What are alternatives to majority voting (kNN)?**

* Rank weighting
  + Every neighbor has a vote with the strength ⇒ 1/rank
  + the slides don’t really specify how this rank is being determined tho :/
* Distance weighting
  + Every neighbor has a vote with the strength ⇒ 1/distance to point

**What problem occurs with KNNs when there are many items to classify and what is a strategy to solve that?**

* Distance calculations to other points in the training data become very computationally intensive
* Solution: Partition the search space with K-d-tree

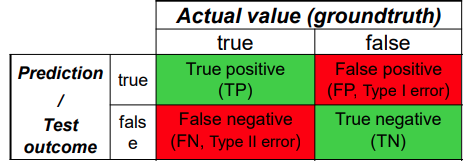
**How does K-d-tree search space partitioning work?**

* Split the search space along points alternating the dimensions

⇒ results in a tree structure

* For predictions: Traverse the tree based on the new point, always taking the path of the nearest point
* Terminate if:
  + parent node is closer to new point than child nodes
  + current node is a leaf node

**For binary classification what are possible results regarding the correctness of the predictions? (Tipp TP, think of it like a matrix)**



**Name and describe 4 performance measures for classification.**

|  | Accuracy  (TP + TN) / Number of samples  ⇒ Correctly predicted samples |
| --- | --- |
|  | Precision  (TP) / (TP + FP)  ⇒ Correctness of the samples that were predicted positive |
|  | Recall  (TP) / (TP + FN)  ⇒ Proportion of identified correct samples |
|  | F1-Score  2 \* (Precision \* Recall) / (Precision + Recall)  ⇒ Harmonic mean of Precision and Recall |

**What is a problem that can occur when you only have a single train and test set? What is a solution for it?**

* Train and test split is randomly determined

⇒ No guarantee it is a good test split

⇒ Might yield wildly different performance than other splits

* Solution: K-Fold cross validation

**Describe K-Fold Cross Validation**

* Split the data in K different equal sized partitions
* K-times use one of the partitions as a test set, while the other partitions are training sets
* train the model on the different splits and then predict their performance

⇒ Resulting performance is average of all resulting performances

⇒ Can compute variance/standard deviation and therefore confidence intervals

**3rd Lecture**

**Name the three different classes/types all data fall into**

* Nominal data
  + unordered
  + example: eye color
* Ordinal data
  + ordered, but no quantifiable distance between different values
  + example: military ranks
* Interval data
  + clearly quantifiable distance between values, but ratios i.e. “one divided by the other” doesn’t make any sense
  + example: temperature (Celsius), pH value
* Ratio data
  + continuous values that also include 0
  + mathematical operations allowed
  + example: temperature (Kelvin), distances

**What is the purpose of Scaling/ Normalization / Standardization?**

* some ML algorithms rely on measuring the distance between the data points (KNN)

⇒ variables with broader value ranges are favored

**Name two different ways for scaling**

* Min-Max-Scaling
* Z-Score standardization

**How does Min-Max-Scaling work**

Each value xi element of variable X gets the following treatment:

xnew = ( xi - min(X) ) / ( max(X) - min(X) )

**How does Z-Score normalization work?**

Each value xi element of variable X gets the following treatment:

xnew = (xi - μ) / σ

μ := average of X

σ := standard deviation of X

**What is label encoding?**

* used on categorical variables
* each unique category is assigned a numeric value
* variable is encoded using the numeric value

**What are the two high-level strategies to deal with missing values?**

* Deletion
  + Row
  + Column
  + Pairwise (Not required for exam)
* Imputation
  + Substitute missing values

**Name at least three different procedures for data imputation?**

* Mean value of the variable
* Random selection of value from other sample
* Use linear regression to predict the missing value
* Cluster values and use the value you get from the center of the cluster
* Nearest neighbor

**Roughly outline the algorithm that can be used to create a decision tree.**

While stopping criterion not reached

For each attribute

determine all possible splits for this attribute

Out of all determined splits over all attributes pick the best split

⇒ Based on some criterion

Stopping criterion might f.e. be all nodes in leave must be the same class

**What are the 4 different performance criterions used for determining splits?**

* Error rate
* Gini index (Gini impurity)
* Information gain
* Variance reduction

**What is the formula for the absolute and relative error rate?**

Number of wrong classifications.

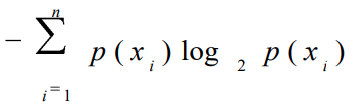
Absolute:

* False Positives + False Negatives

Relative:

* (False Positives + False Negatives) / Number of samples
* 1 - Accuracy

**What is the formula for information entropy?**



H(X) =

X := set of values

p := proportion/probability of class x in X

n := number of classes in X

⇒ Measure of uncertainty in the set of values

**What is the formula for information gain?**

IG(X, S) = H(X) - p(S) \* H(S) - p(SC) \* H(SC)

X := the original set of values before any splits

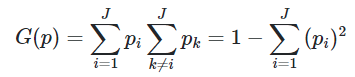
S := split of the original data set

SC := counterpart of the split

p := proportion/probability of the split regarding the size of X

Information gain is a measure for the reduction in entropy achieved by a split.

**What is the formula for the gini index?**



J := number of classes

p := proportion/probability for a given class

Describes how often an instance from the set would be mislabeled, were the labeling do be done randomly according to the distribution in the set

**Is there a meaningful difference between using gini index and information entropy?**

Slides say no

**4th Lecture**

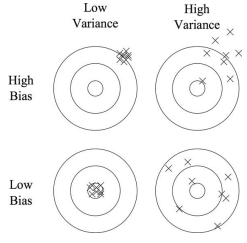
**What problem can occur when the training data of a decision tree gets shifted around slightly?**

Potentially very different tree ⇒ Decision trees are very susceptible to small changes in data

**What does macro-averaging performance measures refer to?**

Computing the mean for a performance measure across all classes

**Explain the terms bias and variance**



Bias:

→ Errors which stem from wrong/erroneous assumptions

→ Model’s predictions are consistently off

Variance:

→ Model is unstable or includes a lot of error/noise

→ Results vary wildly between data sets

→ Often stems from Noise in the training data

**What are methods used for the pre-pruning of decision trees?**

* limit maximum tree depth
* number of observation is lower than threshold
* entropy in leaf node is lower than threshold
* information gain through possible splits is lower than threshold

**Describe the process of simple bottom-up pruning**

Do this for all parents of leaf nodes

1. Make this node a leaf node for its majority class
2. Compare performance
3. If performance is not decreased or lower than threshold ⇒ Keep changes

**Pro/Cons of decision trees?**

* choice of many different split criteria
* can deal with any kind of data
* easily visualized and human friendly
* Stability problems
* Easily over fits

**What is leave-p-out cross validation? Problems, special case?**

* cross validation with a test split size of p
* becomes computationally expensive really quickly
* special case: leave-1-out cross validation

**What is bootstrapping?**

* creating multiple sets of the same source data by sampling with replacement

**Lecture 5**

**What is one of the base assumptions of Naive Bayes?**

* predictive variables are independent from each other

**What is the formula for Bayes’ rule?**

P(H | E) = P(E |H) \* P(H) / P(E)

P(H) → Prior probability of H

P(H | E) → Posterior probability of H: “Probability of H, knowing E occured”

**How can Bayes' rule be used for classification?**

* H represents a possible class of an observation
* E the values of its independent variables

⇒ We can model P(H | E) with Bayes rule (n = number of variables):

P(H | E) =

= P(E | H) \* P(H) / P(E) =

= P(E1 | H) \* P(E3 | H) \* … \* P(En | H) \* P(H) / P(E)

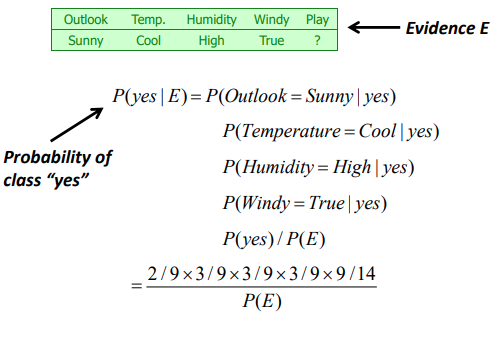
For a variable k (k in [1, n]), we know the probability P(Ek | H) from the data set.

The probability can be computed by:

P(Ek | H) = (α + “number of training observations that have Ek and H”) / (α + “total number of training observations with H”)

→ α is 1 most of the time (explanation later)

⇒ The class with the highest probability is the prediction



**What is the zero frequency problem?**

* occurs when a target class has zero occurrences for a given variable k
* P(Ek | H) would technically be 0 → (H | E) = 0
* μ is introduced into P(Ek | H) to avoid this collapse in probability
* μ is mostly 1, but can also take other forms if appropriate

**Explain Laplace smoothing**

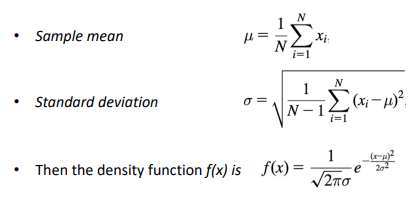
* Another solution to zero-frequency problem
* P(Ek | H) = (α + ”number of true Ek for H”) / (α \* “Number of Features” + “Number of H”)
* alpha is the smoothing parameter and is larger than 0

**How are missing values dealt with?**

* Training: excluded from the frequency calculation P(Ek | H)
* Prediction: excluded from probability calculation

**How would P(Ek | H) calculated for a numeric variable?**

* all variables assumed to be normally distributed
* using probability density function to determine probability for given value



**Are correlated predictive variables a problem?**

* Despite independence being a base assumption - No
* Philipp: Yes that's a problem, because it violates the base assumption, that the variables are independent. In Theory Yes, but in Practice mostly no.
* As long as the correct class gets a higher probability the prediction is correct
* problems start to only occur when many variables are redundant

**How does the algorithm for PRISM work? Pseudo-code**

For each class C

Initialize E to the instance set1

While E contains instances in class C

Create a rule R with an empty left-hand side that predicts class C

Until R is perfect (or there are no more attributes to use) do

For each attribute A not mentioned in R, and each value v,

Consider adding the condition A = v to the left-hand side of R

Select A and v to maximize the accuracy p/t

(break ties by choosing the condition with the largest p)

Add A = v to R

Remove the instances covered by R from E

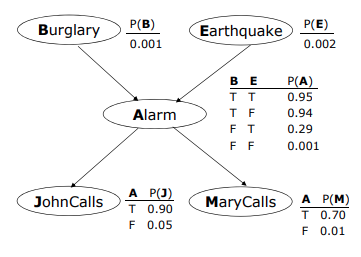
(1) E = all instances

*These slides are really messy and confusing so I left some gaps as this is really hard to gain any useful knowledge from it!*

**Lecture 6**

**Describe the basic components and structure of a Bayesian network**

* made up of a set of variables → nodes
* nodes are linked with a set of directed connections
* resulting graph is acyclic
* probabilities for every node are given → nodes with parents have conditional probabilities for parents
* can be queried for the probability of a certain event

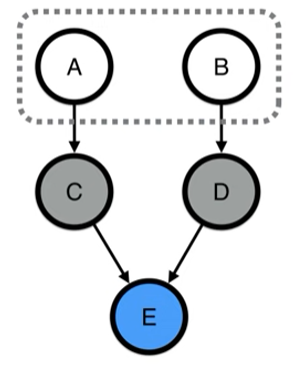


**Explain how the conditional probability of a child node is given**

* always given in relation to its parents
* P(A | B ∩ E)

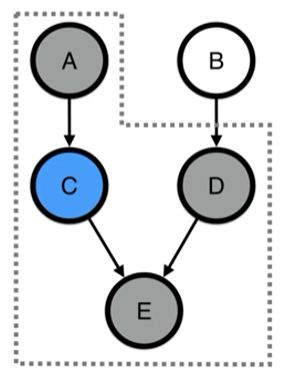
**Explain D-Separation**

* D-Separation is a criterion to be able to tell variable independence easily from a graph
* D-Separation always requires one or more variables to be already observed (i.e. “given”)
* Example:



Given C and D, E is independent of A and B!

* Generally speaking: Each variable is conditionally independent from EVERY other variable given its Markov blanket



* Markov blanket: parents, children and children's-parents are given

very very poorly explained in slides: <http://bayes.cs.ucla.edu/BOOK-2K/d-sep.html>

TODO: I doubt this will be part of the exam. Might expand on it later.

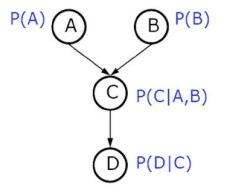
TODO: Skipped over lots of Bayesian Networks Theory here because the slides are the worst and give me a headache. This explanation is better: <https://www.youtube.com/watch?v=TuGDMj43ehw>

**Explain the chain rule**

Given a graph and a set of events you can re-express the event using Bayesian logic.

Example:

Graph:



Event: P(A ∩ B ∩ C ∩ D)

*Yes, this notation is weird! P(¬A) would be the negation of event P(A) i guess.*

P(A ∩ B ∩ C ∩ D) =

= P(D| A ∩ B ∩ C) \* P(A ∩ B ∩ C) =

= P(D| C) \* P(C| A ∩ B) \* P (A ∩ B) =

= P(D|C) \* P(C|A ∩ B) \* P (A) \* P(B)

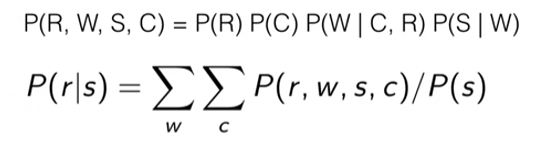
**What are possible strategies for learning Bayesian Networks?**

* Human experts
* Learning from data
* Combination of both approaches
  + Experts create structure
  + Probabilities are learned

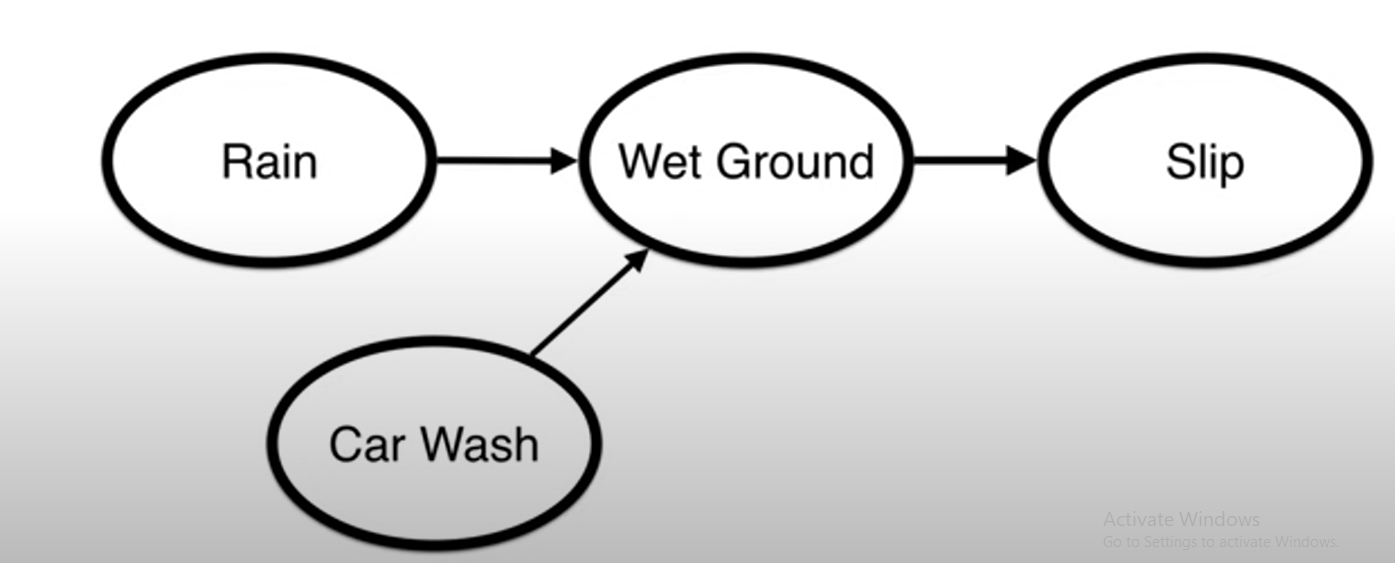
**What is Inference in Bayesian Networks?**

* Brute force way of estimating a probability of a variable in a bayesian network
* Writing out all possibilities for occurrences of event

⇒ “unwrap” the probabilities via chain rule



For network:

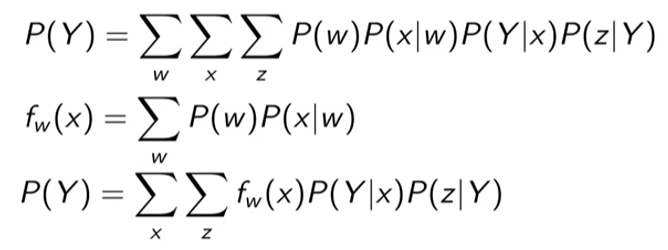


⇒ Iterate over all possibilities of Car wash and Rain in order to get total probability of P(r|s)

**Variable elimination**

* way to simplify computationally intensive statements that result from extrapolating a query by the chain rule
* Simple rule: Every variable that is not an ancestor of a variable in the query or is not contained in the query itself, is irrelevant to it
* Chain rule can be applied by extracting out the effect of a certain variable into a function

⇒ Less computationally expensive



**Lecture 7**

**Formula for RSS (Residual Sum of Squares)?**

RSS := ΣNi=1(y\_predi - y\_truei)2

**What are two ways for the beta-coefficients to be obtained?**

* gradient descent
* hat-matrix

**Pro/Contra of gradient descent**

* Might be less performance intensive when number of features is high
* Possibly many iterations needed
* Might get stuck in local optima

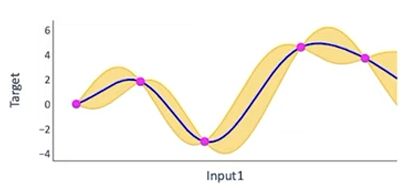
**Explain the basics of Bayesian Optimization**

* process for optimizing parameters
* treat function to optimize as a “black box”

→ no derivatives

* Process

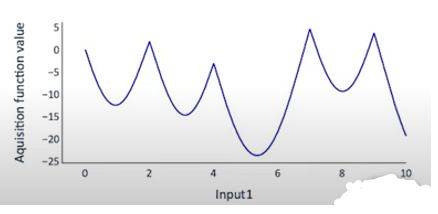
1. Sample loss function at different points (parameters)
2. Train a gaussian process regressor on it



pink → sampled points

yellow → standard deviation of model

1. compute the acquisition function with hyperparameter kappa



lower bound of gaussian process regressor - kappa \* standard deviation

1. find lowest point
2. repeat from step 2 until some stopping criterion

Source: <https://www.youtube.com/watch?v=M-NTkxfd7-8>

**What differentiates Ridge Regression from regular Regression?**

* includes extra penalty term based on tuning parameter λ
* cost function: RSS + λ \* ΣNi=1 βi2
* ΣNi=1 βi2 → Squared sum of coefficients also called L1-penalty

**What is the effect of λ on model bias and variance?**

* Large λ

⇒ low variance ⇒ less overfitting, better generalization

⇒ high bias ⇒ worse model fit to training data

* Low λ

⇒ high variance ⇒ more overfitting

⇒ low bias ⇒ Higher fit to training data

**What differentiates Lasso from Ridge Regression?**

* The penalty differs
* Called L2-penalty
* ΣNi=1 |βi|
* generally results in more coefficients being shrunken to 0

→ feature selection

**Name 3 performance measures for regression models**

* Mean squared error (MSE): (1/n) \* ΣNi=1(y\_predi - y\_truei)2
* Root mean squared error (RMSE): square root of MSE
* Correlation of prediction and true values

**What are regression trees? 4 facts**

Similar to normal decision trees

* Splitting criterion: minimize intra-subset variation
* Termination criterion: std. dev. becomes small
* Pruning criterion: based on numeric error measure
* Prediction: Leaf predicts average class value of instances

**What are model trees?**

* Extension of regression trees
* Each leaf has its own linear regression function
* Results are handed up to the root of the tree and are being interpolated with the results of the traversed nodes
* Formula:

p’ → the prediction passed up to the next higher node

p → prediction passed to this node from below

q → the value predicted by the model at this node

n → number of training instances that reach the node below

k → smoothing constant

**Lecture 8**

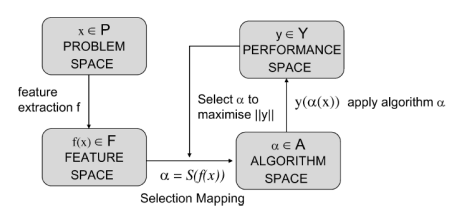
**Explain the “No Free Lunch” (NFL) theorem**

* for any algorithm, any elevated performance over one class of problems is offset by performance over another class
* any two algorithms are equivalent when their performance is averaged across all possible problems

**Describe the Term “Model Combination”**

* Also known as “ensemble methods”
* Combination of different algorithms to create a single system with high stability
* Also considered Meta-Learning like Auto-ML

**Describe Rice’s Framework for Model Selection**



* Problem Space: Set of potential problems
* Feature space: Meta-Information about the features the problems come with
* Algorithm Space: The set of all algorithms that can be selected from
* Performance space: Each algorithm has associated performance metrics

⇒ Determine α to minimize performance metrics

**How should algorithms be chosen for the algorithm space of a model selection system?**

* representatives from different model classes

⇒ f.e.: Bayesian, Decision tree, MLP, etc

* Ideally smallest subset to cover all types of learning algorithms

⇒ very hard to do/impossible

**What are examples of entries in the feature space?**

* Number of attributes
* Number of classes
* Ratio of examples to attributes
* Average class entropy
* Degree of correlation between features and target concept

**Explain the concept of Landmarking**

* “Map” of expertise for the algorithms in the algorithm space
* Map coordinates are made up of the features in the feature space
* each algorithm occupies a region in the feature space
* using obtained features of a problem enables you to “locate” a fitting algorithm

**What is a constraint that landmarking faces?**

Computational cost of landmarking needs to be lower than simply running all algorithms in A on a given problem

**Explain AutoML and what it entails**

* Process of automating of machine learning when applied to a data set
* Automated feature selection, preprocessing & construction
* Automated algorithm selection
* Automated optimization of hyperparameters
* Automated model validation

**Lecture 9**

**Explain the basic concept of random forests**

* make n different bootstrap samples from the training data
* Create n decision trees based on these samples
* At each individual tree node only make a random subset of features available for splits

⇒ Accentuate difference between trees

* For Predictions: Let trees “vote” on the class of a given sample (majority wins)

**Explain the out-of-bag performance measure**

* General approach for evaluating algorithms using bootstrap
* For each bootstrap sample take instance not included in sample
* Compute a fitting performance measure with these samples
* Average the performance results over all the samples used for training

**How do Multilayer Perceptrons need to be modified to allow for Regression?**

* Output can be taken directly from the last node in the network
* no activation function needed

**Give a high level overview of how to train Multi-Layer Perceptrons**

* Weights for each node are randomly initialized
* Carried out through backpropagation

⇒ Starting at the output layer and working towards the input layer

* Uses gradient descent

⇒ Reduce each weights contribution to the error

⇒ Compute partial derivative of the loss function for the current weight

⇒ Step towards the direction that minimizes loss based on learning rate

Here I skipped over the exact algorithm for training MLPs. This might be something to look into before the exam, though it is most likely too complex to be featured in it.

**What are design choices and tuning parameters for Multi-Layer Perceptrons?**

* How many neurons on each of the hidden layers?
  + higher than 2/3 of input layer neurons
  + less than double the number of input layer neurons
* How many hidden layers to choose?
  + Generally 1 to 2
* What kind of activation function to use?
  + These days typically RELU

**Explain the problem of the vanishing gradient**

* Problem that occurs when training a MLP through backpropagation
* Only the weights in the neurons near the output layer get changed in a meaningful manner
* Due to applying the chain rule to determine the partial derivative of the loss function it often occurs that the derivative becomes increasingly small for later neurons
* Gradient decreases exponentially with each layer further from the output layer
* Can create situations where the network effectively ceases to learn any new weights for any layers after the output layer

[Source](https://en.wikipedia.org/wiki/Vanishing_gradient_problem#:~:text=In%20machine%20learning%2C%20the%20vanishing,based%20learning%20methods%20and%20backpropagation.&text=The%20problem%20is%20that%20in,weight%20from%20changing%20its%20value)

**Why split the data into Training-Validation-Test Split?**

* The test set is often used for model selection and parameter tuning

⇒ Model implicitly optimized for test set

⇒ Test set does not yield an exact estimate for the generalization performance of the model

* Remedy: Create a third split ⇒ Validation split
* Use this split for model tuning and only ever use test set to estimate the “real world” performance of the final model

**What are appropriate ratios for Training-Validation-Test Split?**

* 60% training set
* 20% validation set
* 20% test set

**Name 4 steps used for text feature extraction**

1. Tokenization

⇒ Splitting up the sequence of words into tokens

1. Normalization

⇒ map tokens that have the exact same meaning to the exact same token

⇒ U.S.A. and USA should be the same token

1. Stemming

⇒ Reduce the words to their word stem

⇒ authorized and authorization have very similar meaning and will be resolved to the same token

1. Stop word removal

⇒ Omitting very common words that have little meaning

⇒ This step is optional

**Why does scaling/standardization play an important role for text features?**

* Text documents have different sizes

⇒ Absolute word count will favor longer documents

**Explain the term “Deep Learning”**

* Term associated with Neural Networks
* Learning is considered to be deep when there are many (hidden) layers involved

⇒ Especially applicable to image classification

Edge detection layer

Image Feature Layer

Actual classification layer

**Contrast Deep Learning with Traditional ML**

Traditional ML:

* express problems through mathematical function
* Learn parameters from prepared data

⇒ Feature extraction hard coded

Deep Learning:

* Use an ensemble of simple functions to create solutions
* Let algorithm operate on raw data (e.g. images) and learn its own feature extraction
* Learn parameters based on extracted features

**Why do these networks need to be deep?**

* technically a network with only one hidden layer can approximate any desired function
* However: This exact network might be hard to find

⇒ In practice easier to train a deep neural network

* Also self learned feature extraction needs additional layers anyway

**Explain Convolutional Neural Networks**

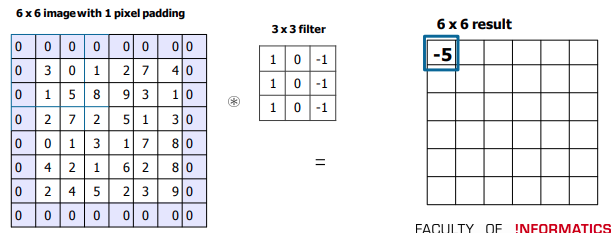
* used mainly for image classification
* chains convolutional and pooling layers before passing the results to a fully connected layer (i.e. the actual classifier)
* Convolutional layers:
  + Slide a convolution-tensor over the input tensor (f.e. a raw image) → Results in output tensor
  + convolution-tensor is initialized randomly and then learned → more useful tensors are developed while training
  + convolution-tensor has the function of an image filter that can f.e. detect edge
* Pooling layers:
  + Reduce the dimensionality of the output tensors
  + done to reduce the amount of features
  + can be done by taking the maximum or the average in a subset of a tensors values

Slides were bad. Source: <https://www.youtube.com/watch?v=tH0A1nrIrzc>

**What is the stride size regarding convolutions?**

* It is by how many units the matrix/tensor gets shifted when sliding

**Why is a padding needed for convolutions?**



→ Otherwise the results for the outermost values in the output tensor would be NaN

**Explain the kernel size and how it might be optimized**

* kernel is how big the tensor used for convolution is (f.e. 3x3, 7,7)
* bigger → less detail but more context
* smaller → more detail but less context

⇒ solution: instead of increasing the size of kernels you can stack multiple small ones

⇒ 3 stacked 3x3 kernels “see” as much as a single 7x7 kernel

**Lecture 10**

**Explain data augmentation**

* permute training data in across properties that are supposed to be irrelevant to model
  + f.e.: picture rotation, facing or scale, color, contrast
* creates more training data
* makes the trained model more robust

**Ways to do data augmentation for audio data?**

* pitching
* shifting
* noise

**Why is ReLu preferred over sigmoid as an activation function?**

* training/learning is faster
* vanishing gradient problem is less likely

**How can you avoid overfitting in a Neural Network?**

* Make the model simpler ⇒ Less layers, reduce input size
* Regularization ⇒ Penalize the magnitude of weights
* Data augmentation ⇒ For example for image classification turn and shift the image
* Dropout ⇒ Drop neurons from the network in each iteration
* Early stopping ⇒ Avoid overfitting by limiting the number of epochs

Source: <https://www.kdnuggets.com/2019/12/5-techniques-prevent-overfitting-neural-networks.html>

**What is a Batch as it relates to Gradient descent?**

* A batch consists of a set of samples/observations

**What is Batching as it relates to gradient descent for Neural Networks?**

* during the gradient descent process multiple batches are created
* the error/loss is computed for said batch and the weights are adjusted accordingly
* then another batch is taken and the process starts over again ⇒ iterative

**Explain the difference between Batching, Mini-Batching and stochastic gradient descent?**

* Batching ⇒ Batch size is entire training set
* Mini-Batching ⇒ 1 < Batch size < training set size
* stochastic gradient descent ⇒ batch size is 1

**What is an Epoch regarding Neural Networks?**

* One run through the entire training set to estimate the weights
* training a Neural Network is often done in multiple Epochs

**What is Dropout**

* dropout is deleting a random subset of neurons in the network

**What is the purpose of Dropout**

* prevents co-adaptation (layers overfitting to compensate for the mistakes other layers made)
* creates more sparsity → regularizing effect on network

⇒ Overall decreases overfitting

**What does sparsity refer to regarding neural networks?**

* many weights being 0

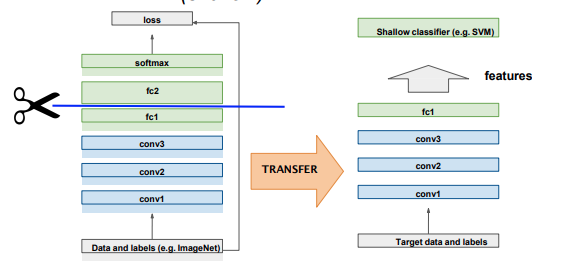
I left some stuff out here like batch normalization, mini-batch gradient descent, Adam (Adaptive Moment Estimation). These topics are really interesting, but they are very specific to Neural Networks and will probably not be covered in the exam.

**Pro/Contra for Deep Learning**

* Benefitted a lot from recent progress in hardware
* Are applicable to a wide range of tasks
* Require less domain knowledge (for feature design)
* Often outperform “hand-crafting” or feature design (images!)
* Open up new design spaces (network architecture, training data, training method) which may be less well understood
* Large networks usually require large data sets (and lots of computing power / electricity)
* A lot of experimenting & parameter tweaking needed
* Trained networks are often hard to analyze (“black box”)

**Explain Transfer Learning**

* Take a pre-trained neural network and use it for another problem
* Possibly make adaptations to make it more compatible with that problem
* also possible to train a network on an adjacent domain to apply it to the original problem
* Variations
  + Same domain → different task
  + Different domain → same task



take feature extraction layers and pass them to another classifier to obtain the desired results

⇒ Continue fine tuning the network afterwards if there is enough data to do so (avoid overfitting)

**Purpose of Transfer Learning**

* having insufficient data to train a network on the problem in question directly

**Which parts of a neural network are best used for transfer learning**

* The feature extraction layer
* Higher layers are often too task specific to be useful for transfer learning

**Explain adversarial ML**

* neural networks are a black box

⇒ lack of understanding of the model makes it exploitable

* Can be fed data, which cause wrong predictions



**Name four types of adversarial attacks**

* Poisoning/Backdoor attacks
* Evasion attacks/Adversarial Inputs
* Membership Inference Attack
* Model Extraction/Stealing

**Explain Poisoning/Backdoor attack**

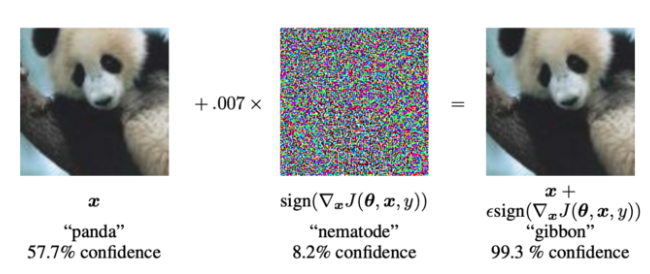
* Attacker needs access to the training data (possible if data is pulled from the web)
* embedding of patterns in the data to create backdoors

⇒ attackers train the model for their desired behavior

* Possibly manipulation of labels

⇒ especially dangerous for reinforcement learners since new data always comes in

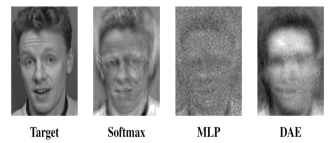
**Explain Evasion attacks/Adversarial Inputs**

* Data is manipulated in a way that leads to misclassification despite obvious indicators that should enable correct classification
* Problem: Many of these “perturbations” (injection of noise) are transferable among DNN models

**Membership Inference Attack**

* possible if an attacker can make any number of desired predictions with the model (f.e. model accessible via open API)
* Make many predictions to make inferences about the training data

⇒ Possibly reconstruct training set



**Model Extraction/Stealing**

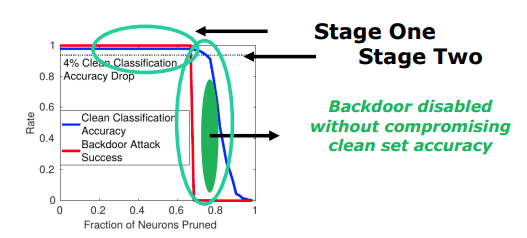
* Attacker reconstructs model from predictions
* Especially harmful if a company invested much into a model just to have it reconstructed for cheap

⇒ Undermines pay for prediction model

⇒ Enables evasion attacks

**How can Poisoning/Backdoor attacks be mitigated?**

* pruning/dropout



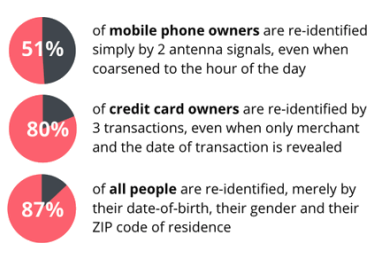
**Name different strategies to retain data privacy even when working with sensitive data**

* Data Sanitisation
* Differential Privacy
* Synthetic Data
* Federated Machine Learning

⇒ all strategies oftentimes comes with a drop in Accuracy

**What is Data Sanitisation**

* Tries to solve privacy issues that arise when people’s information is recovered from models or data sets
* Pseudonymisation → Remove directly identifying information (e.g. name)
* Anonymization → Remove indirectly identifying information (e.g. birthdate and zip-code)



Even a small amount of info can make somebody's identity identifiable

* Achieved by
  + dropping
  + encoding
  + reduce precision

…of sensitive information

**Explain Differential Privacy**

* principle to retain privacy of subjects in a statistical database
* queries can be made to the database, but inferences are prevented through randomness in the output data



Data base “Magic”/Randomness Output of query

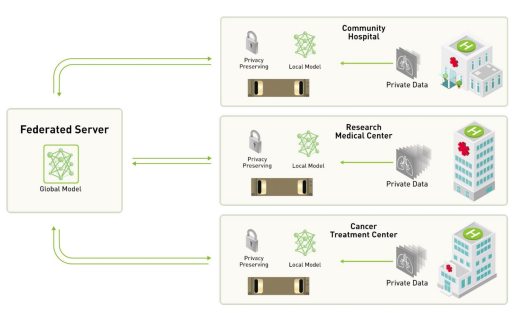
**Explain Synthetic Data as it relates to Privacy**

* alternative to differential privacy or data sanitisation
* create a completely synthetic data set, which preserves characteristics of real world data
* Use other Statistics & ML models to generate data

⇒ nearly equivalent

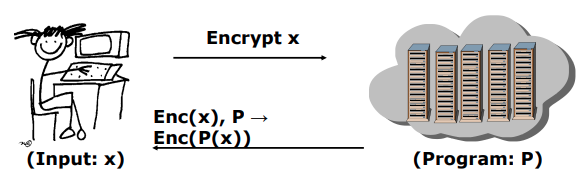
**Federated Machine Learning**

* one model is trained on a central node and sent to all participants
* Participants train model on private and save data set
* send trained model back to central node
* integrate the trained model into central model
* iterative process



**Explain Homomorphic Encryption**

* is an encryption scheme that allows for computation on the data without revealing sensitive contents
* enables users to delegate processing needed for ML to an untrusted computing provider



I skipped over the topic of Secure Computation. The slides are kinda bad.

**Explain k-Anonymity**

* enables privacy of individuals through “hiding in the crowd”
* for each identifying variable the values are being generalized (like binning)

⇒ ZIP code gets shortened to first few numbers, age gets reduced to age range

* results in a group of individuals that share an identifiable value

⇒ k denotes minimum group size

⇒ larger k more anonymity

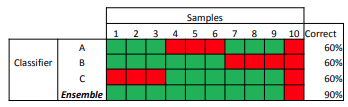
* Can have a large impact on model performance

**Lecture 11**

**Explain Ensemble learning**

* Combining several classifiers and their predictions

⇒ Goal: ensemble performance better than individual performance



**Name three types of classifier outputs/predictions as they relate to ensemble learning**

* abstract level
  + only a single prediction
* rank level
  + ranked list of predictions
* measurement level
  + probability estimate for every single class

**Why is the output type so important for ensemble learning?**

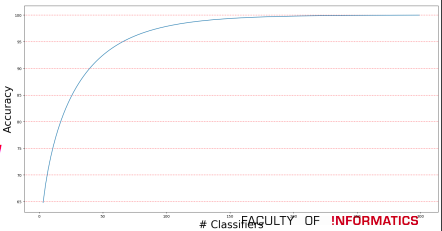
the more detailed the output the more information can be gained by their combination

**Explain why ensemble learning has the potential for very high performance, but why this can be easily realized**

* assuming the predictions of the classifiers is independent

⇒ no (strong) correlation between predictions

* the ensemble classifier has extremely great performance as the number of contained classifiers increase



* Caveat: Performance of individual classifiers seldomly independent

**Explain weighted majority voting**

* prioritize classifiers with better performance
* weights need to be trained on split of the training data

⇒ otherwise data leakage

**What is the difference between homogeneous and heterogeneous ensemble learners?**

* homogeneous

⇒ contains the same type of classifiers

⇒ e.g. random forests → only contain decision trees

* heterogeneous

⇒ contain different types of classifiers

⇒ e.g. mix of KNN, SVM, Decision Tree

**Explain Bagging as it relates to ensemble learners**

* individual classifiers are trained on bootstrap subsamples of the training data

⇒ each classifier has slightly different data set

* increase independence of predictions
* Can make unstable classifiers more stable

⇒ e.g. random forests

**Explain the concept of Boosting**

* classifier depends upon predecessor

⇒ Analyze errors from predecessor, decide on which “part

of the data” to focus on

⇒ Set weights to focus on “hard” training samples

* Iterative weighting and training
* Classifiers are often weak learners like 1R/Decision Stump

**Explain AdaBoost**

* uses boosting to train multiple weak learners in sequence
* each created model has an associated weight to it

⇒ Determined by how much it reduced the previous models error

⇒ Weight is used for contribution to the final prediction

**Explain Gradient Boosting**

* very similar to AdaBoost
* differences:
  + Uses deeper trees (8 to 32 leaves) not just 1R
  + Fits subsequent trees on the (pseudo) residuals of the previous trees

⇒ Offsets the errors made by the previous model by adding the estimated residuals

* + has constant learning rate (e.g, 0.1) instead of weights to regulate contribution of adjustments to the model

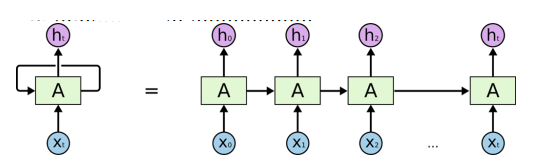
**Why is gradient boosting related to gradient descent?**

* Gradient Descent: minimize a function by moving in the opposite direction of the gradient
* Gradient Boosting; Move towards the opposite direction of the residuals

Better explanation: <https://www.youtube.com/watch?v=3CC4N4z3GJc>

**Explain recurrent neural networks (RNNs)**

* Specialized neural network for sequential data
  + Machine translation → sequence of words
  + Video tagging → sequence of images
* Neurons process inputs from their ancestor as well as their own output from the last piece in the sequence



⇒ Can be seen as unwrapping the network across time

⇒ Previous versions of the network pass their results to subsequent versions

**What is a problem with RNNs and what solution addresses it**

* information from previous steps is not very persistent and quickly vanishes

⇒ Bad long term memory

* Only most recent outputs are being considered
* Solved by LSTM (Long Short Term Memory) networks!

**Explain LSTMs**

* Longer retention of information across sequences than normal RNNs
* Flow of information from previous networks to later networks regulated by gates

⇒ Certain information is retained longer while other info is discarded immediately

**Why is explainable AI necessary?**

* stakeholders need to be able to trust outcomes

⇒ know the reasoning behind them

**What are types of explainability?**

* intrinsic ⇒ explainability features built into the model (e.g. linear models)
* post-hoc ⇒ extracting information from model/results
* ex-ante ⇒ explain by analyzing the training data

**What are the types of explanations/What forms do they take?**

* statistics
* visualizations
* weights
* examples
* proxy-models ⇒ e.g. a decision tree representing the decision process of a DNN

**What are the general types of approaches for explainability?**

* Model specific/model agnostic
* global/instance based

**Explain an explainable AI strategy for images**

* show why the image was selected by:
  + showing similar images
  + showing maximally different images

**Explain counterfactual explanations**

* “It was not classified as Y since it does not observe X”
* Done by minimizing loss function:



x := features

x’ := features that lead to another outcome

y’ := desired other outcome λ

d := distance measure

⇒ Minimize distance between features, while also minimizing distance between results

* Example:



→ This bird is a Crested Auklet, because this is a black bird with a small orange beak and it is not a Red Faced Cormorant because it does not have a long flat bill.

**Explain McNemar’s Test**

* non-parametric test ⇒ no need for a distribution like t-test/z-test
* dichotomous dependent variable

⇒ f.e. binary classification

⇒ values are either 0 or 1

* observations across samples are connected ⇒ “couples”

⇒ n-th observation in the first sample is connected to n-th observation in second sample

⇒ for example for a before and after treatment for fixed cohort of patients

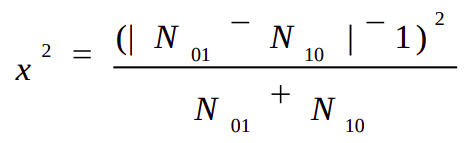
* compute “confusion matrix”

| # samples correct in A & B (N11) | # samples correct A, wrong B (N10) |
| --- | --- |
| # samples wrong A, correct B (N01) | # samples wrong in A & B (N00) |

* H0: discordant cells are equal

⇒ Cells that “did not stay the same”

* Test statistic chi-squared:



⇒ Compare results to results on chi-squared table

**Motivation for variable selection**

* less computational effort
* some ml algorithms have a problem with correlated independent variables

⇒ multicollinearity

* simpler model

⇒ better interpretability

**Explain the bag-of-words approach for text indexing**

* every word has its own feature

⇒ value is the number of occurrences of that word

* the number of features is often too high!

⇒ similar problems in images and music

**Explain filter methods**

* feature selection method
* find the best subset before training ⇒ basically preprocessing step



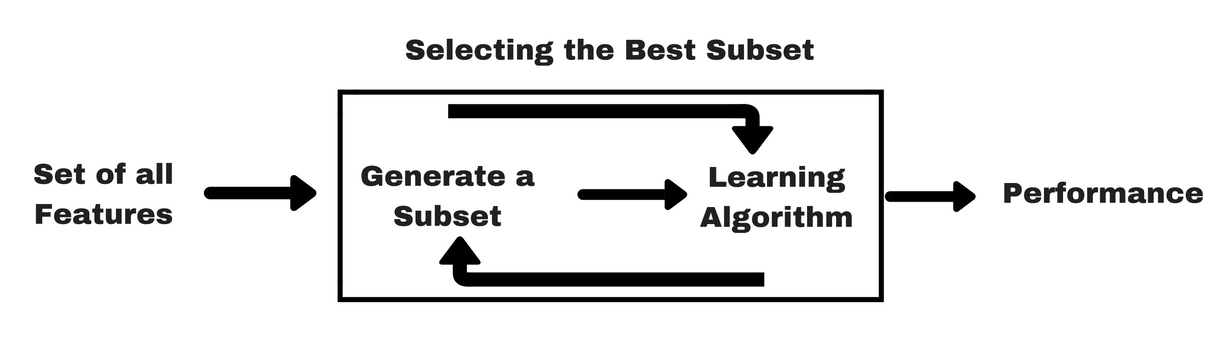
⇒ compute the best subset via different statistical measures

⇒ lower computational cost

⇒ unconnected to algorithm that comes afterwards

**Explain wrapper methods**

* try to find the best subset by successively training the model and evaluating its performance



⇒ basically search space problem

⇒ higher computational cost

⇒ best feature set for this algorithm

Source: <https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/>

**Name methods for unsupervised feature selection**

* PCA (principal component analysis)
* Frequencies

⇒ based on number of occurrences

⇒ f.e. very applicable for bag of words

⇒ can be focusing on the relative “commonness” or “rareness” of a value

* measure correlation between variables to avoid multicollinearity

⇒ Anything over a certain threshold gets dropped

⇒ Probably good to combine with supervised feature selection to rank the features to be dropped by weakest correlation with response

⇒ Also works for Chi-Square

* ranking by variance
* ranking by mean absolute difference

⇒ very nice because it’s proportional to actual values and their range

**Name supervised feature selection methods**

* ranking by correlation to response

⇒ Also works for Chi-Squared

* ranking by fisher’s score
* ranking by information gain
* ranking by mutual information

⇒ The ranking methods require you to pick the n-highest ranked variables to include

**Explain the basic ideas and concepts of SVMs**

* classification algorithm
* used to linearly separate a negative and a positive class
* the decision boundary is based on the so called support vectors

⇒ instances that are problematic, since they are the ones most likely to be misclassified

* the decision boundary is constructed in a way that maximizes the distance of the support vectors to the decision boundary

⇒ results in “best” possible split

⇒ the distance between the decision boundary and the support vectors is called the margin (i.e. the margin is maximized)

* SVMs can employ so called “Kernels” to increase the dimensionality of the data and find better decision boundaries

**Explain Kernels in detail**

* Kernels are functions that create higher dimensional data from the existing data
* A simple of this would be the multiplication of features
* These extra dimensions make more decision boundaries possible

**Explain the difference between the separable and the non-separable case and why it is important**

* The methods to create the decision boundary differs slightly between data sets which are linearly separable and others that are not
* Data sets that are non-separable include so called “slack-variables” in their objective functions

**Questions from last exam**

**Entropy maximal when all elements are from the same class?**

⇒ No, entropy is minimal then: 1 \* log21 = 0

**Random Forests heterogeneous ensemble learner?**

⇒ No is homogeneous, all contained models are of the same type

**PCA supervised feature selection method?**

⇒ No it is unsupervised since it does not relate back to the response variable

**Is McNemar's Test used for significance testing?**

⇒ It is a type of significance test so yes

**Z-scaling vs. min/max scaling**

Z-scaling: (x - μ) / σ

Min/Max scaling: (x - min(X)) / (max(X) - min(X))

**Gradient Descent pseudocode**

1. Initialize the weights W randomly.
2. Calculate the gradient G of the cost function.
   1. This is done using partial differentiation: G = ∂J(W)/∂W.
3. Update the weights by an amount proportional to G, i.e. W = W - ηG
4. Repeat until the cost J(w) stops reducing, or some other predefined termination criteria is met.

η = learning rate

J(W) = loss function for weights W

**Random Forests. What makes a RM random?**

Bootstrap samples are randomly sampled with replacement

At each tree node a random subset of the variables is available

**Something about Feature Selection. Name methods of Feature Selection**

* stepwise feature selection
  + forward
  + backward
  + both
  + based on BIC or AIC
* PCA or PCR
* Eliminate highly correlated variables if correlation is a problem
* Shrinkage methods/Regularization like lasso
* Variable importance in random forests

**What is Deep Learning? How does it compare to "traditional" Machine Learning?**

Deep Learning is an umbrella term for machine learning models that are based on neural networks and have many (most importantly more than two) layers.

Deep Learning is especially useful for image classification due to Convolutional Neural Networks.

LSTMs are very useful for data sets of sequential nature.

Traditional machine learning models are often (but not always) simpler and easier to understand. While DNNs have many learned parameters, traditional models have much fewer, since they do not have many stacked layers that all contain their own weights. Linear regression for example only has one coefficient for each variable + an Intercept, while a Neural Network for a similar task would already have many more weights.

Another big difference is that Deep Learning can include its own “self-learned” feature extraction. This is for example done by the convolutional layers of a CNN.

**Bayesian Networks: Explain…**

* **Inference**
* **Variable Elimination**
* **D-Separation**

**Inference:** Make inferences about probabilities of certain events that can occur in the network. F.e. describe an event by specifying the value of certain variables. Through extrapolating the specified event via the chain rule we get a concrete probability for said event.

**Variable Elimination:** Was not covered in the slides

**Questions to the lecturers**

What are methods for the automatic construction of Bayesian Networks?

⇒ Basically randomly choosing new ones based on

Could you please explain the following topics relating Bayesian Networks:

* Inference
* D-Separation
* Variable Elimination

Is Bayesian optimization used for the construction of Bayesian networks?

Is kNN recommended for large data sets?

Are categorical features supposed to be normalized when using them for kNN?

When should you employ min-max scaling and when z-scaling?

What exactly are model based attributes in meta learning?

Can you use Naive Bayes for continuous data or regression tasks?

Are freezing layers modified during the fine-tuning phase?

No. They don’t get unfrozen.

Is overfit more likely on a smaller test set?

Why can a general Bayesian Network give better results than Naive Bayes?

Which are the important issues to consider when you apply Rice's framework for automated selection of machine learning algorithms?

Is D-Separation used in neighborhood searches regarding Bayesian networks?

off the shelf: no fine tuning and combination with traditional ml ⇒ simply no fine tuning possible

TODO: Review Dropout