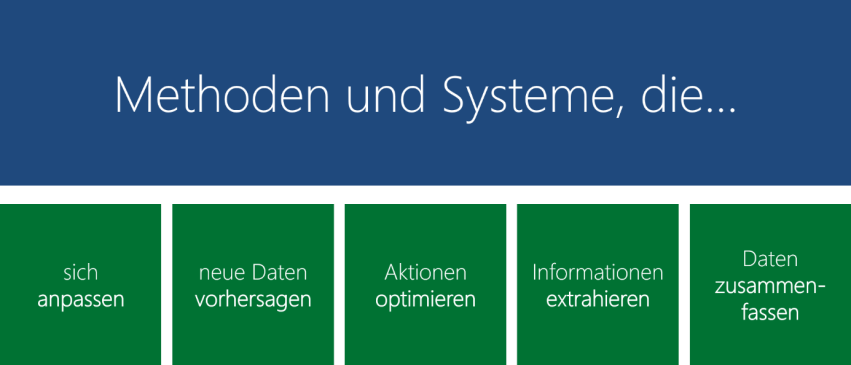
Fragenkatalog ASS

**1. Was ist Machine learning (ML) im Allgemeinen? Wovon hängt ein ML Algorithmus am stärksten ab und wodurch kann dieser daher verbessert werden? Nennen Sie Problemstellungen, die durch ML gelöst werden können.**

Ist ein Konstrukt aus Algorithmen die aus einem Datenbestand lernen und daraus intelligente Aktionen generieren können.



Am stärksten hängt Machine Learning von den Daten ab, je mehr aussagekräftigere Daten desto besser das Modell.

***Problemstellungen:*** Erkennung von Spam Mails (Spamfilter), Spracherkennungssysteme, Erkennung von Kreditkartenbetrug.

**2. Erkläre die MAPE loop. Was kann dadurch vermieden werden? Was passiert in den einzelnen Schritten?**

Der MAPE Loop in Bezug auf Adaption ist ein Verfahren, um ein System zu schaffen, welches sich selbst organisiert und dynamisch laufend an Änderungen anpasst.

**M**onitor – Sammeln von Daten der zu organisierenden Ressource (Topologie Infos, Metriken,…), sucht nach Symptomen welche genauer analysiert werden müssen

**A**nalyze – Führt Datenanalysen aufgrund der durch die Monitoringphase entdeckten Symptome durch. Falls Änderungen nötig sind, wird eine Anfrage an die Planungsphase weitergeleitet

**P**lan – erstellt einen Plan mit korrigierenden Maßnahmen

**E**xecute – Führt die Änderungen des Plans durch

Durch den MAPE loop können adaptive Systeme angepasst/geändert werden, ohne dass diese neu gestartet werden müssen.

**3. Was ist ein Peer-To-Peer Netzwerk? Welcher Peer hat besonderen Stellenwert in einen P2P Netzwerk und wer übernimmt die Organisation? Welche Vorteile hat P2P gegenüber Client-Server Systeme?**

Ist eine spezielle Art eines Rechennetzwerkes bei welchen beliebig viele Rechner ihre *Ressourcen* **bündeln** (Bandbreite, Rechenleistung, Speicher, …) und alle Rechner gleich berechtigt sind. Alle Peers sind in diesem System *gleichberechtigt*. Das zentrale Interesse eines P2P Netzwerkes liegt im Teilen von Ressourcen. Einfache P2P Netzwerke *organisieren sich selbst*, das heißt es gibt keinen zentralen Server. Von P2P Netzwerken gibt es 2 Typen, *Unstructured P2P* und *Structured P2P*. Vorteile von P2P Systemen gegenüber Client-Server Systemen sind: es gibt keinen single Point of failure, P2P Netzwerke sind in der Regel relativ Robust.

**4. Nennen Sie je einen Vorteil, Nachteil von P2P System. Erklären Sie diese konkret an einem Anwendungsfall und stellen Sie einen Vergleich zu Client-Server Architektur.**

**Vorteil:** geringfügige Kosten 🡪 Rechner kann sofort ins Netzwerk integriert werden. Es fallen höchstens Kosten für Kabeln und Instandhaltung an. Im Gegensatz zu Client-Server Systemen, die bei entsprechender Größe einen sehr teuren Server benötigen, sind die Kosten für ein P2P Netzwerk relativ gering. Je nach Größe des Netzwerkes können die Kosten variieren.

**Nachteil:** kaum Sicherheit und schwer administrierbar 🡪 Da jeder Rechner im System gleich berechtigt ist und es keine zentrale Anlaufstelle gibt können keine komplexen Berechtigungs-Hierarchien aufgebaut werden. Bei Client-Server Systemen werden alle Zugriffe vom Server verwaltet. Diese Zugriffe können von ihm geregelt werden.

**5. Was ist Ziel einer Distributed Hash Table? Vergleichen Sie die Anzahl der Zugriffe und 1 Vor-+ 1 Nachteil mit P2P Systemen ohne DHT.**

Ein Schlüssel wird auf einen Wert gemappt. Ziel ist es Struktur zur Kommunikation einzelner Nodes in einem P2P Netzwerk einzuführen. Ein System ohne Distributed Hash Table ist bei der Suche nach einer Datei oder einem Node sehr ineffizient. Das System wird hier gefloodet, d.h. jeder Rechner schickt seinen unmittelbaren Nachbarn eine Anfrage ob er der gesuchte Rechner ist bzw. die gesuchte Datei besitzt.

**Vorteil P2P ohne DHT:** Kein Aufwand für Verwaltung der Informationen (🡪 Keine Routing- bzw. Fingertable vorhanden)

**Nachteil P2P ohne DHT:** Bei Suchen nach Rechnern oder Dateien in großen Netzen sehr inneffizient.

**6. Wie funktioniert DHT (Chord) und welche Funktionen gibt es?**

Peers sind in Chord Ringförmig angeordnet. Jeder Peer kennt seinen direkten Nachfolger als auch seinen direkten Vorgänger

**Ein Peer speichert folgende Informationen:** Vorgänger, Nachfolger, Fingertable.

Die Navigation erfolgt über einen Index in einem virtuellen Netzwerk 🡪 Nicht über die IP Addresse  
Damit die Navigation schneller vonstattengeht als ohne (Laufzeitverhalten *O(n)*), benötigt man die Fingertabelle, dazu speichert jeder Knoten weitere Nachfolger in dieser Tabelle um Abkürzungen nehmen zu können.



***Successor:*** Funktion lautet *find\_successor(k)* und gibt den nächsten direkten Nachfolger vom

Knoten *k* an.

***Hash Funktion:*** Um aus einen Key einen identifier zu erzeugen.

***Insert:*** Einfügen eines Key-Value-pair auf einem Knoten.

***Lookup:*** Anfragender Knoten sendet seine IP und seine Key an seinen Nachfolger. Diese Nachricht wird solange weitergereicht bis die Ressource gefunden ist. Der Knoten welche die Ressource hat schickt diese dann direkt zum Anfragenden Knoten.

***Join the swarm:*** Ein neuer Knoten wird ins Netzwerk eingefügt.

**7. Wie funktioniert der Insert in DHT? Skizzieren Sie die Schritte mit einem m = 2 Ring.**

Es wird ein Key-Value Pair *(torrent, myIP)* an einem Index im Ring eingefügt.

1. Key = *hash(torrent)* 🡪 Erzeugt den Index.
2. Durch die Methode *successor(key)* wird der für das Speichern zuständige Knoten gefunden.
3. Diesem wird anschließend das einzufügende Key-Value Pair zugewiesen.

So wird der Index zufällig im Netzwerk verteilt. Der Key wird immer auf dem Knoten gespeichert der dem Key am nächsten ist.



**8. Wie funktioniert Join in DHT und welche Möglichkeiten gibt es als Node eine DHT wieder zu verlassen?**

**Join:**

1. Der Knoten der den Ring joinen will sendet einen Broadcast in den Ring den er joinen will
2. Irgendein **Knoten s** meldet sich.
3. Der einzufügenden Knoten hasht seine IP-Addresse und bekommt als Ergebnis einen Key

🡪 *Key = hash(IP)*

1. Der neue eingefügte Knoten fragt den **Knoten s** wer ist mein **successor x/predecessor y** Knoten

🡪 **s.find*successor(key)***.

1. Der eingefügte Knoten speichert sich danach diese Information und fügt sich zwischen Knoten x und Knoten y ein.
2. Danach werden die Fingertables der Knoten upgedatet.

In Chord können Knoten eine DHT entweder *Graceful* (gibt seine Keys an seinen *successor* weiter) oder via *Crashing* (Knoten ist einfach weg) verlassen.

**9. Wie funktioniert Destination Sequenced Distance Vector? Welche Daten müssen pro Node gespeichert werden?**

Ist ein Routing Algorithmus basierend auf den Bellman-Ford algorithmus. Es wird versucht den kürzesten Weg in einem Graphen (Netzwerk) zu finden. Jeder Knoten hat eine Tabelle mit dem besten und günstigsten Weg zu seinen Nachbar Knoten. Diese Tabellen werden upgedatet durch Informationsaustausch und Abgleich mit Nachbarknoten. Jeder Knoten ist so in der Lage die beste Verbindung zu jedem Ziel zu berechnen.

Pro Node müssen folgende Infos gespeichert werden: *next hop, number of hops* and *sequence number*

**10. Wie funktioniert Link State Routing? Warum sind berechnete Routen in LSR immer optimal?**

Durch Link-State Routing wird der kürzeste Weg in einem Netzwerk von A nach B errechnet. In Link-State Routing werden 5 folgende Schritten pro Knoten angewandt:

1. Erforsche Nachbarn und lerne Ihre Adressen
2. Erstelle Distanz oder Kosten-Metrik zu jedem Nachbar Knoten
3. Erstelle ein Paket mit allen gerade gelernten Informationen.
4. Sende das Packet an alle Nachbarn des Knotens und empfange auch deren Informationen.
5. Berechne den kürzesten Pfad zu jedem anderen Knoten.

Berechnete Routen sind immer optimal weil der Dijkstra Algorithmus bei der Berechnung des kürzesten Pfades eingesetzt wird. Durch diesen wird immer der kürzeste Pfad errechnet.

**11. Wie funktioniert das optimierte Link-State-Routing? Was wird in OLSR optimiert und wodurch? Skizzieren Sie ein kleines Beispiel indem ein OLSR sinnvoll verbessert werden kann (im Gegensatz zu LSR).**

Alle Knoten / Router kennen die vollständige Netztopologie, somit können sie jederzeit den kürzesten Weg zum Ziel festlegen.

Proaktives Routing Protokoll -> Daten immer gespeichert

Topologie bei OLSR: via HELLO und TC (Topology Control) Nachrichten.

1 und 2 Hop Nachbarn via periodische HELLO Nachrichten (enthalten IP aller bekannten 1 Hop Nachbarn, sowie Status der Verbindung)

Aus allen 1 Hop Nachbarn selektiert jeder Knoten MPRs (Multipoint Relay) womit er alle 2 Hop Knoten erreichen kann MPRs leiten Broadcast Nachrichten weiter.

MPRs erstellen TC Nachrichten mit allen Knoten von welchen sie als MPR ausgewählt wurden.

**Optimierung:** OLSR optimiert das Routing indem instabile Verbindungen ausgespart werden und die Netzbelastung durch die Umgehung von MPRs reduziert wird. Jeder Knoten ist durch seine IP gekennzeichnet.

Im OLSR-Protokoll wird der Dijkstra-Algorithmus eingesetzt.



**12. Wie funktioniert ad-hoc on-demand distance vector? Was unterscheiden AoDDV grundsätzlich zu DSDV?**

AoDDV = Adhoc on Demand Distance Vector DSDV = Destination Sequenced Distance Vector Topologie basiertes, reaktives Routingprotokoll für mobile ad-hoc Netze ->

Routen werden bei Bedarf ermittel (hop zu hop), solange gültige Routen vorhanden sind ist AODV inaktiv.

Schleifenfreiheit durch Sequenznummer

Routingtabelle besteht aus: IP Adresse des next hop, Sequenznummer des Zielknotens, Distanz Zielknoten, Status Flags RREQ = Route Requests -> Per Broadcast, endet bei Zielknoten oder bei Zwischenknoten der eine aktuelle Route besitzt, alle bisherigen + aktueller Knoten wird immer weitergeleitet RREP = Route Replies -> Unicast, gesamte Route zu Ziel RERR = Route Errors -> wenn Unterbrechung in Verbindung erkannt wird gesendet

Der Ad-hoc On-demand Distance Vector-Routingalgorithmus (AODV) ist ein Algorithmus zum Weiterleiten von Daten durch ein mobiles Ad-hoc-Netz. Das Protokoll gehört zu den Topologie basierten, *reaktiven* *Routingverfahren*.

Das heißt Routen zu bestimmten Zielen werden erst bei Bedarf ermittelt (hop bei hop).

**13. Was ist Dynamic Source Routing? Wodurch unterscheidet es sich von anderen besprochenen Mechanismen? Erklären Sie das Prinzip von RREQ und RREP.**

DSR = Dynamic Source Routing für wireless mesh netzwerke, reaktives Routing -> es wird in jedem Paket eine Liste aller Zieladressen verpackt Teilnehmer bekommen Routen von weitergeleiteten Nachrichten. Protokoll ist ähnlich zu AODV, DSR -> weniger Datenaufkommen route discovery -> suche nach route zu gewünschtem Ziel route maintenance -> überprüfe ob Verbindung noch aktuell -> Fehler aus Zwischenspeicher bereinigen route caching -> optimierte Routenfindung, neue Routen von RREQ und RREP, auch Zwischenrouten

**Prinzip von RREQ und RREP:**

1. Ursprungskonten sendet (RREQ = Anfragenachricht)
2. Route ist bestimmt wenn RREQ Ziel erreicht oder Knoten findet der eine aktuelle Route hat
3. Die Route wird in der Antwortnachricht (RREP=Unicast) zurück zum Ursprung geschickt
4. Jeder Knoten der RREQ empfangen hat speichert sich Ursprungsknoten
5. Bei Unterbrechungen der Verbindung wird Fehlernachricht (RERR) gesendet.

**Dynamic Source Routing vs. AODV:**

* **Route discovery** 
  + Suche nach einer Route zu einem gewünschten Ziel;
  + Speichere festgestellte Routen in einen eigenen Zwischenspeicher)
* **Route maintenance**
  + Überprüfe ob die Verbindungen noch immer aufrecht sind und aktiv
  + Besondere Fehlermeldungen um Einträge im Zwischenspeicher zu bereinigen
* **Route caching**
  + Optimiert Routenfindung in Zwischenknoten
  + Knoten erlernen neue Routen von Route Request (RREQ) und Route Replies (RREP)
  + Knoten können auch von Zwischenrouten einer Route lernen

**14. Nennen Sie 2 Strategien für Kommunikation und vergleichen Sie diese. Nennen Sie je ein Beispiel für eine Anwendung und nennen Sie ein Beispiel, wo Sie die Kommunikation nicht einsetzen würden. Begründen Sie ihre Antwort.**

**Antwort von Rosi (weis nicht ob das reicht?):**

Message-Oriented Transient Communication (synchronous = TCP/IP, UDP/IP, sockets, MPI) Remote Procedure Calls (synchronous and asynchronous = RMI, XML/RPC) Data-Space Communication (asynchronous = blackboard, tuple space, JavaSpaces) Code Migration (passing programs instead of data -> applets / flash / silverlight)

**Bestehende Antwort:**

* **Message-Oriented Transient Communication** (Beispiel: **FTP (TCP), Video streaming (UDP)**)
* **Remote Procedure Call** (Beispiel **Network File System NFS**)

Message-Oriented Transient Communication:

Synchron, TCP/IP, UDP/IP, sockets, MPI (Message-Passing Interface), … . Ist flexible und kümmert sich selbst um Fehler, sessions, marshalling etc.

**Transient**: Eine Nachricht wird nur so lange gespeichert bis der Absender „sender“ und Empfänger „receiver“ aufhören zu arbeiten. (TCP, UPD, IP routing). Das heißt die Übertragung erfolgt synchron. Client und Server müssen beide aktiv sein während der Kommunikation. Der Client gibt Anfragen aus und blockiert solange bis er eine Antwort empfängt. Der Server wartet eigentlich nur auf eintreffende Anfragen und verarbeitet diese anschließend.

Remote Procedure Call:

Synchron und Asynchron, RMI (Remote Method Invocation), XML-RPC,.. ; Abstraktion, aber Seiteneffekte können verschleiert werden

Ist ein Protokoll für die Ebenen 5 und 6 und gewährleistet einen entfernten Prozeduraufruf. Jeder Server im Netz stellt im Rahmen dieses Konzeptes eine Anzahl von Diensten zur Verfügung, die mit RPC angefordert werden können. Diese Funktionen sind als Prozeduren eines Programms realisiert und können unter Angabe von Serveradresse, Programmnummer und Prozedurnummer angesprochen werden. Der Server, der RPC-Dienste unterstützt, bekommt Anfragen von Clients, auf denen die Anwendung lokal verbreitet wird. Erhält der Server eine RPC-Anfrage vom Client, führt er eine lokale Prozedur aus. Nach Beendigung der Prozedur überträgt er die Ergebnisse zum Client und wartet auf die nächsten Anfragen. Um Sicherheitsaspekte zu gewährleisten, muss sich der Client beim Server ausweisen. Diese Berechtigung erstreckt sich auch auf die Nutzung anderer Dateien.   
Remote Procedure Call (RPC) ist anwendungsorientiert und setzt auf dem UDP- oder dem TCP-Protokoll auf.

**15. Welche Klassifizierung gibt es bei Data-Space Kommunikation? Geben Sie 2 Beispiele (gesamt) mit Begründung warum diese Kategorie an Kommunikation hier gut passt.**

In Data-Space Kommunikation gibt es 4 Arten Klassifizierungen:

* Zeitlich gekoppelt, Referenziell gekoppelt: direkte Kommunikation
* Zeitlich entkoppelt, Referenziell gekoppelt: Mailbox
* Zeitlich gekoppelt, Referenziell entkoppelt: Meeting-orientiert
* Zeitlich entkoppelt, Referenziell entkoppelt: Generative Kommunikation

**Bsp.:**

Referentiell gekoppelt, Zeitlich gekoppelt:

- Direkte Kommunikation

- Die Beteiligten kennen sich und es muss zeitgleich stattfinden (Prozesse müssen zeitgleich aktiv sein)

**Referentiell gekoppelt, Zeitlich entkoppelt:**

- Mailbox

- Die Beteiligten kenne sich, die Nachricht muss allerdings nicht zeitgleich zur Verfügung stehen (Prozesse müssen nicht zeitgleich aktiv sein)

**16. Welche JavaSpaces Operationen werden unterschieden? Erklären Sie wie die CRUD (Create Retrieve Update Delete) Actions auf einen Space durchgeführt werden können.**

In JavaSpace Wird zwischen folgende Operationen unterschieden:

* Write 🡪 Schreibt Elemente in den TupleSpace
* Read 🡪 Liest Element aus dem TupleSpace (Macht eine Kopie davon)
* Take 🡪 Liest Element aus dem TupleSpace und entfernt es gleichzeitig

CRUD (Create Retrieve Update Delete) Actions werden auf dem Space wie folgt durchgeführt:

* **C**reate: mit der JavaSpace Methode „write“ wird ein Objekt in den JavaSpace geschrieben
* **R**etrieve: ein Objekt kann mit der JavaSpace Methode „read“ aus dem JavaSpace abgerufen werden
* **U**pdate: ein Update eines im JavaSpace befindlichen Objektes kann durch folgende Operationen realisiert werden.

1. „take“ (Liest Element und entfernt es aus dem Space)
2. Ändere Objekt nach Wunsch
3. „write“ Schreibe modifiziertes Objekt zurück in den Space

* **D**elete: ein Objekt wird aus dem JavaSpace entfernt durch die Methode „take“

**17. Was sind die Vor- und Nachteile von Code-Migration? Welchen SPOF (Single Point of Failure) kann man hier identifizieren?**

**Vorteile:** Load Balancing, Performance, keine vor-Installation

**Nachteile:** oft proprietär, Sicherheit

Als single Point of Failure kann hier das Repository (Stelle von dem der Code bezogen wird) identifiziert werden. Fällt dieses aus ist die Funktionalität von Code Migration nicht mehr gegeben.

**18. Wie ist der klassische Ablauf einer Service Discovery? Welche Integrationsstrategien für Ad-Hoc Kommunikation von Geräten gibt es?**

Service Discovery ist für die automatische Erkennung von Geräten und Services, welche von diesen angeboten werden, in einem Computernetzwerk. Dafür muss der Service Provider eine Möglichkeit haben seine Dienste nach außen hin bekannt zu machen.



Der User Agent sendet eine Anfrage nach einem Service über Multicast oder über Broadcast request aus. *(****Broadcast****: zu allen Rechnern im Netzwerk welche auf Broadcast Adresse hören.* ***Multicast****: an alle Geräte die sich dafür interessieren, d.h. sie müssen sich vorher als Interessent angemeldet haben).* Wenn der Service Agent eine Anfrage über ein Service erhält welches er anbietet sendet dieser dem User Agent über Unicast *(direkt Übertragung)* die Adresse *(in einem Service Advertisement verpackt, binhaltet URL und Service Beschreibung)* unter welcher dieses Service zu erreichen ist. Es gibt folgende Integrationsstrategien bei Ad-Hoc Kommunikation von Geräten:

**DHCP** 🡪 Um Service Informationen zu verteilen

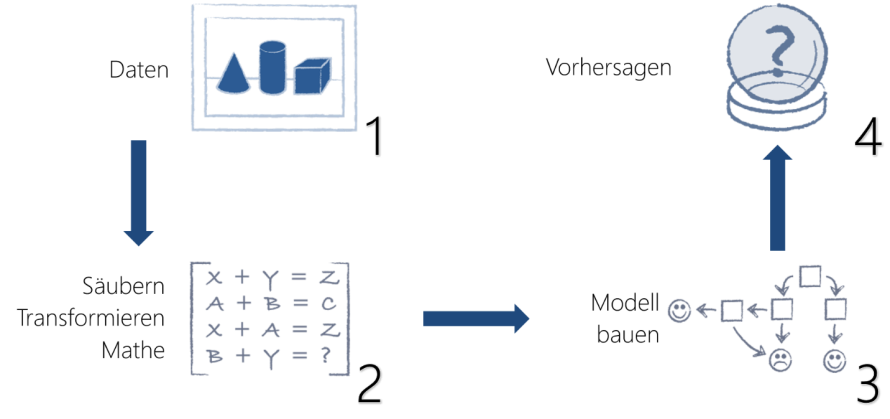
**Automatisch** 🡪 basiert auf der Service Beschreibung (UDDI / WSDL) **Manuell** 🡪 nicht skalierbar, nicht zu empfehlen bei sich stark ändernden Netzen

**19. Was ist Data Mining? Was ist der Unterschied/Zusammenhang zu Machine Learning?**

Unter Data Mining versteht man die systematische Anwendung von Methoden auf einen Datenbestand. Das Ziel ist die Erkennung von neuen Mustern. Beim Data Mining geht es primär um Datenbestände die so groß sind, dass sie nicht mehr von Menschenhand verarbeitet werden können.

Der Unterschied zu Machine Learning ist, dass beim Data Mining der Fokus auf dem Finden von neuen Mustern liegt. Im Gegenteil zu Machine Learning bei dem bekannte Muster in neuen Daten wiedererkannt werden sollen. Verfahren aus dem Machine Learning finden oft beim Data Mining Anwendung und umgekehrt.

**20. Woraus besteht der Lernprozess in Machine Learning? Wie kann das Model verbessert werden?**



Vor dem Modellieren sollte der Datenmüll gesäubert werden. Bei der Datenmodellierung sollten die einzelnen Spalten/Attribute/Features genauer untersucht werden, da nicht jedes davon interessant ist.

**21. Was bedeutet Bias und was bedeutet Over-Fitting? Wie stehen die beiden Werte zueinander? Wie kann man Overfitting verhindern?**

Beim Overfitting wird die Datenrepresentation zu sehr an die Trainingsdaten angepasst. Das kann entweder aufgrund von fehlerhafte Daten (Rauschen) oder zu wenig Datensätzen auftreten.

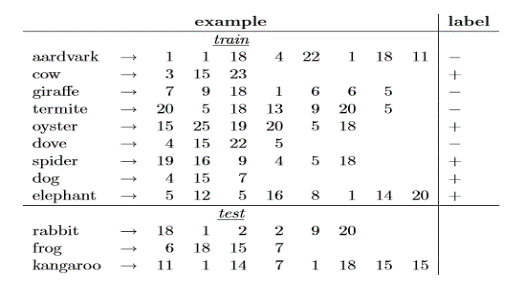
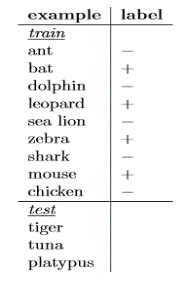
Bei parametrischen Modellen sollte versucht werden, eine geringe Anzahl an Parametern anzustreben.  
Bei nichtparametrischen Modellen sollte versucht werden die Anzahl der Freiheitsgrade im Vorhinein zu beschränken. Weiters können Methoden wie die Kreuzvalidierung oder v-fache Kreuzvalidierung zur Vermeidung von Overfitting angewandt werden. Modell so klein wie nur möglich halten.

**22. Welche Schritte im Prozess sind anzuwenden um mit machine learning Algorithmen zu arbeiten?**

**…**

**23. Was sind Features und Examples in machine learning? Beschreiben Sie diese und erläutern Sie im Anschluss anhand eines konkreten Beispiels.**

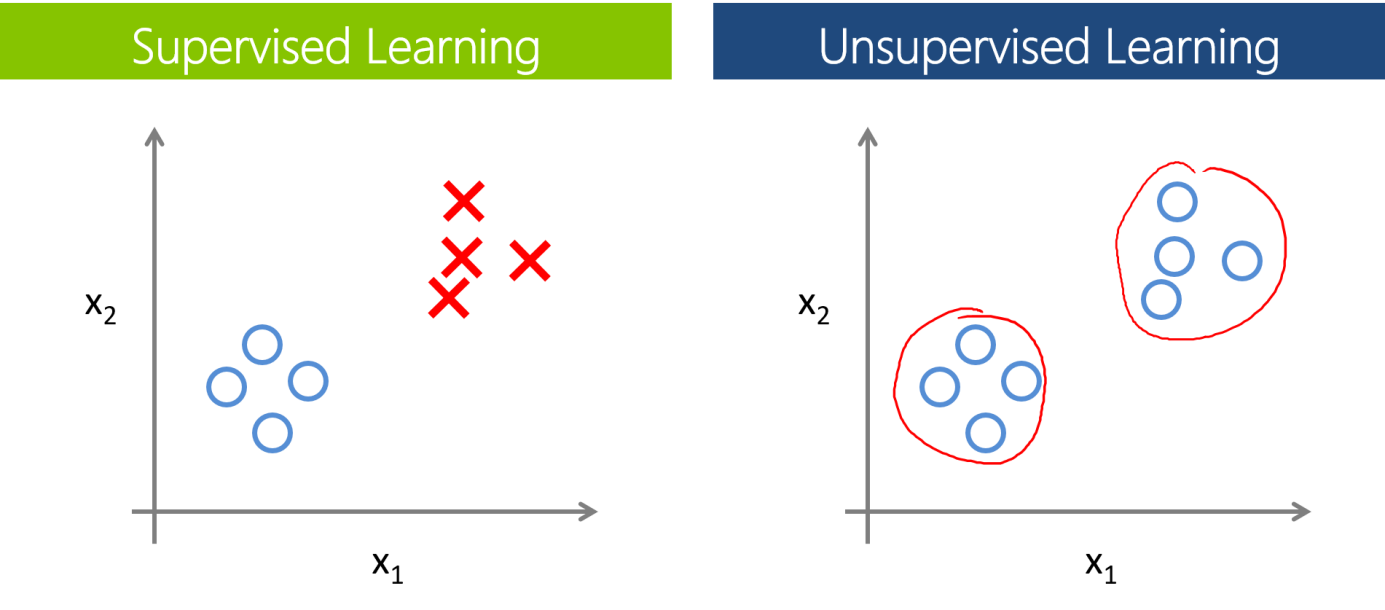
Ein Example ist ein Trainingseintrag.



Ein Feature ist eine Charakterisierung durch Attribut-Wert-Paare der Eigenschaften der Daten. Diese Features werden als Datenbasis für Algorithmen verwendet. Features werden verwendet um Daten besser zu verarbeiten.  
Beispiel 🡪 Ein Feature beim Filtern vom Spam Mails könnte die E-Mail Struktur, grammatische Korrektheit oder die Sprache sein.

**24. In welche 2 Kategorien können machine learning Algorithmen eingeteilt werden? Nennen Sie je ein Beispiel für eine konkrete Anwendung.**

**Überwachtes Lernen:** Beim überwachten Lernen wird versucht Gesetzmäßigkeiten nachzubilden. Die Ergebnisse sind bekannt und werden verwendet um das System anzulernen. Die Ergebnisse des Lernprozesses werden mit den bekannten, richtigen Ergebnissen verglichen.  
Ziel: „Zukunft voraussagen“  
Beispiel >> Handschrifterkennung  
**Unüberwachtes Lernen:** Beim unüberwachten Lernen sind die Ergebnisse im Voraus nicht bekannt. Es wird versucht anhand der Eingaben Muster zu erkennen. Das System lernt durch die Eingabemuster und versucht diese in Kategorien zu klassifizieren (Clustern).   
Ziel: Vorhandene Daten zu vestehen  
Beispiel >> Klassifizierung von Münzen, Genomforschung



**Während wir bei Supervised Learning schon im Voraus wissen, dass es genau zwei Kategorien gibt (blauer Kreis oder rotes Kreuz), ergeben sich bei Unsupervised Learning zwei Gruppen.**

**25. Welches Ziel hat die Regressionsanalyse? Gehen Sie sowohl auf die Regression im Allgemeinen als auch auf die lineare Regression ein. Was passiert bei der Berechnung linearer Regression wenn bei den Messdaten exakt 1 Messfehler auftritt und statt einem durchschnittlichen Wert ein extrem hoher Wert gemessen wird (Beispielsweise statt 15 kommt 15.000.000)?**

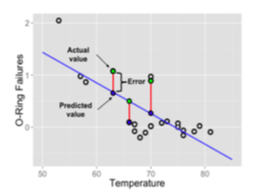
Regressionsanalysen sind statische Analysen. Sie stellen die Beziehung zwischen einer abhängigen und einer oder mehrerer unabhängigen Variablen her.

Eine lineare Regression definiert die Abhängigkeit einer Variable Y zu einer unabhängigen Variable X. Wären es mehrere unabhängige Variablen X, dann sprechen wir von einer multiplen linearen Regression.

Hoher Ausreißer. Verzerrt die Linie stark. Zum Beispiel bei der Methode der kleinsten Quadrate.

**26. Was ist Ziel des OLS (Ordinary least Square; de: Methode der kleinsten Quadrate) – Vorgehens? Erklären Sie dabei auch den Begriff Residuum.**

Es wird versucht die Abweichung der Gerade zwischen den Messpunkten zu minimieren. Die Gerade soll so nah wie möglich an die Messpunkte angeglichen werden. Residuum ist dabei der Abstand/Abweichung von der geschätzten Gerade zum tatsächlichen Messpunkt.



**27. Was besagt der „correlation coefficient“? Erklären Sie dabei   
a) welchen Sinn der Korrelationskoeffizient hat,   
b) welche Einheit,   
c) welchen Wertebereich,   
d) was die Extremwerte und 0 bedeuten,   
e) unter welchen Randbedingungen er Anwendung findet und   
f) ob der Korrelationskoeffizient zwischen X und Y   
 gleich dem Korrelationskoeffizienten zwischen Y und X ( rxy ?= ryx).**

Der „correlation coefficient“ erklärt den Zusammenhang zwischen zwei intervallskalierten Merkmalen.

1. Er bestimmt in wie weit ein linearer Zusammenhang zu einem Messwert besteht. Im Fall der linearen Regression kann damit bestimmt werden, ob der Messwert wichtig für die Voraussage ist oder nicht.
2. Es gibt keine Einheit
3. Von -1 bis +1
4. Extremwerte: vollständiger linearer Zusammenhang (bei +1 steigt die Gerade an, bei -1 sinkt die Gerade). 0: beide Merkmale hängen gar nicht linear voneinander ab.
5. Es muss eine lineare Abhängigkeit gegeben sein. Gibt den Grad des linearen Zusammenhangs zu einem Messwert an.
6. Ja es ist gleich

**28. Was ist „Entropy“? Erklären Sie in Folge die Entropie anhand des konkreten Beispiels „Würfeln mit Spezialwürfel“. Der Spezialwürfel hat dabei 2 blaue Seiten und 4 rote. Erklären Sie „Information gain“.**

Die Entropie beschreibt die Menge an Zufall, die in einem oder mehreren zufälligen Ergebnissen steckt. Die Entropie wäre 0 wenn ein sicheres Ergebnis oder eine Folge von sicheren Ergebnissen vorliegt.

Würfel = 6 Seiten

Blau = 2/6

Rot = 4/6

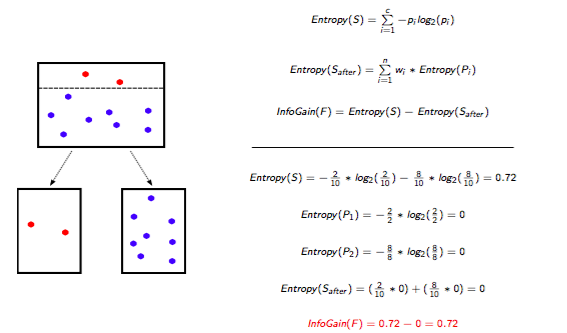
Entropie = - (p \* log2 (p) + q \* log2 (q)) =

- (4/6 \* log2 (4/6) + 2/6 \* log2 (2/6)) =

- (4/6 \*( -0.584963) + 2/6 \*( -1.584963)) =

-(0.0817036666666 - 1,2516296666666) = 1.169926

Information Gain: Ist die Änderung der Homogenität der Daten nach dem Split. Es geht also darum, ob nach dem Split die Daten besser aufgeteilt sind. Wenn in einem Segment nur rote sind und im anderen nur blaue, dann ergibt es den maximalen info gain bei einem Split.



**29. Warum sollte man einen Entscheidungsbaum kürzen (=„pruning“)? Gehen Sie auf Pre- und Post-Pruning Algorithmen ein.**

Falsche Attributwerte oder Klassenzugehörigkeiten, die die Daten verfälschen, vergrößern den Entscheidungsbaum. Durch Pruning werden die unnötigen Sub-Bäume gekürzt und dadurch die Größe entscheidend verringert und die Klassifizierungsgenauigkeit ungesehener Objekte verbessert.

Beim Post Pruning werden Knoten und Teilbäume durch Blätter ersetzt.

**30. Was ist die grundlegende Idee von Regression Trees? Was wird wie partitioniert? Warum heißt es "Recursive Partitioning"? Was haben Elemente innerhalb einer Partition gemeinsam? Was ist der Unterschied (im Prinzip, in der Vorgehensweise) zwischen Regression und Decision Tree?**

**31. Wie bildet man Klassen für k-NN Algorithmus und wie funktioniert er? Wie werden unklassifizierte Examples klassifiziert? Worauf ist zu achten wenn man k klein oder k groß ansetzt? Welche Ansätze gibt es, um den k-NN Algorithmus zu verbessern?**

**32. Was versteht man unter k-fold cross validation? Was versteht man unter data splitting?**

**33. Wozu gibt es AOP im Allgemeinen? Was wird dadurch gelöst, was in "herkömmlichem" OOP nur mit sehr viel Aufwand verbunden wäre.**

Aspekt orientierte Programmierung (AOP) ist ein Programmierparadigma für die objektorientierte Programmierung, um generische Funktionalitäten über mehrere Klassen hinweg zu verwenden (**Cross-Cutting Concern**). Beispiele für solche Cross-Cutting Concerns sind: Security, logging, caching, etc.

Mit AOP wird die Trennung von **Core-Level-Concerns** (Businesslogic 🡪 Anforderungen an ein Programm 🡪 funktionalen Anforderungen) und **System-Level-Concerns** (betreffen das gesamte System/ technische Randbedingungen 🡪 Logging, Security, etc) ermöglicht, welches durch die herkömmliche OOP nur mit viel Aufwand gelöst werden kann.

Die beiden Teile Core-Level-Concerns und System-Level-Concerns sind miteinander „verwoben“. Die Core-Level-Concerns kann man als Komponenten bezeichnen und die System-Level-Concerns sind die Aspekte. Core-Level-Concerns werden überlicherweise als Module oder Objekte implementiert. Für die Aspekte gab es vor der aspektorientierten Programmierung keine elegante Lösung.

Das Problem der miteinander verwobenen Anforderungen wird auch als Cross-Cutting Concerns bezeichnet, denn sie „schneiden“ quer durch alle logischen Schichten des Systems. AOP ist das Werkzeug, um die logisch unabhängigen Belange auch physisch voneinander zu trennen. Dabei wird angestrebt, Code zu erzeugen, der besser wartbar und wiederverwendbar ist.

**34. Wozu dient ein Aspekt und aus welchen Bestandteilen setzt er sich zusammen?**

Die Idee eines Aspekts ist die Implementierung eines Crosscutting Concerns, wie bspw. Logging, Security, Caching. Durch einen Aspekt muss ein Anliegen wie Security nicht im Code verstreut implementiert werden, sondern die Implementierung kann an einer zentralen Stelle (im Aspekt) erfolgen.

Ein Aspekt beinhaltet:

* Weaving Rules: „definiert“ WAS (Advice) WOHIN (Pointcut) gewebt wird
* Pointcuts
* Advices
* Statische Crosscutting Concerns (Struktur)

**35. Was ist ein Aspekt Weaver? Welche Strategien gibt es und wodurch unterscheiden sich diese?**

Ein Aspekt Weaver verwebt den Programmcode/Bytecode an den definierten Verwebungspunkten (Joinpoints) mit dem Aspektcode.

Es gibt drei Strategien:

* Source Weaving: Der Programmcode wird mit dem Aspektcode vor der Übersetzung verwebt.
* Binary Weaving: Der Aspektcode wird mit dem bereits übersetzten Bytecode verwebt.
* Load-time Weaving: Der Aspektcode wird mit dem übersetzten Bytecode erst dann verwoben, wenn dieser geladen wird.

**36. Welche technischen Komponenten hat AspectJ? Nennen Sie ein weiteres AOP-Produkt.**

* Eigener Aspekt Compiler (ajc – AspectJ Compiler)
* Tool Unterstützung für Eclipse (AspectJ Development Tools)
* Aspekt-Definition: durch eigene Keywords (früher) oder durch Annotationen

Weitere Implementierungen: SpringAOP (Java, .NET), PostSharp (.NET)

**37. Nennen Sie einen Vor- und einen Nachteil von AOP und dessen wichtigste Begriffe/Konzepte mit deren Zusammenhang zueinander.**

Nachteile: Lernaufwand für AOP, wenige Tools

Vorteile: Sauberere Implementierung (Crosscutting Concerns sind nicht verteilt sondern zentral an einer Stelle definiert), Code wiederverwendbarer.

Wichtigste Begriffe:

**Join Point**: Möglicher Ausführungspunkt für einen Advice

**Pointcut**: Eine Auswahl von **Join** **Points**

**Statisches** **Crosscutting**: Inter-type declarations (betrifft statische Struktur), weave-time declaration

**Dynamisches** **Crosscutting**: before/after/around **advices** (betrifft Methoden)

**Advice**: Hinzuzufügendes/alternatives Verhalten (vor/nach/um JoinPoint)

**Aspect**: zentrale Einheit, welche die Weaving Regeln beinhaltet (**Pointcuts**, **Adivces** (dynamisch), **statisches** **Crosscutting**)

**38. Welche Arten von Advices gibt es? Welche Bestandteile hat ein Advice?**

* Before: Wird vor JoinPoint ausgeführt
* After: Wird nach JoinPoint ausgeführt
* Around: Umgibt JoinPoint oder ersetzt ihn durch alternatives Verhalten

**Bestandteile:**

* Advice Deklaration: Before, After, Around
* Pointcut Spezifikation: Welche Join Points vom Advice adressiert werden
* Advice Body: Welcher Code beim Join Point ausgeführt wird

**39. Was ist der Unterschied zwischen einem Call und einem Execution Pointcut? Kann überall wo ein Call Pointcut angewendet wird, auch ein Execution Pointcut angewendet werden? Begründen Sie Ihre Antwort.**

Call Pointcut: Der Advice wird beim Aufruf der Methode verwoben (beim Aufrufenden). Dadurch steht auch Information über den Aufrufenden zur Verfügung.

Execution Pointcut: Der Advice wird in der Methode selbst verwoben.

Nein, beispielsweise können System-Klassen oder Third Party Klassen nicht durch execution Pointcuts verändert werden. Es kann aber sehr wohl auf ihre Aufrufe durch call Pointcuts reagiert werden, da das Verweben im Aufrufer stattfindet.

**40. Was unterscheidet einen Advice von einer Methode? Welche Unterschiede haben before/after Advices zu around Advices?**

* Advices haben keine Namen
* Advices können nicht direkt aufgerufen werden
* Advices haben keine Access Specifier
* Advices haben zusätzliche Variablen (thisJoinPoint, thisJoinPointStaticPart, thisEnclosingJoinPointStaticPart)
* Before/After Advices haben keinen Rückgabewert
* Around Advices können die Methode proceed() aufrufen, um die Zielmethode aufzurufen.
* Around Advices können Rückgabewert haben.

**41. Welche Methoden gibt es für Strukturänderungen?**

* Inter-type declaration (Deklaration für andere Typen, Typhierarchie Änderungen)
* Weave-time error and warning declarations (Findet join-points, zeigt Fehler und Warnungen auf)
* Exception softening (Exceptions werden abgefangen und auf eine Crosscutting Art und Weise verarbeitet)