michiel vd broek

Onderzoeks-document  
machine Learning

Inhoud

[Inleiding 2](#_Toc510860201)

[Welk machine learning algoritme kan ik het best gebruiken binnen mijn project? 3](#_Toc510860202)

[Wat maakt een algoritme “machine learning”? 4](#_Toc510860203)

[Wat is het verschil tussen machine learning en artificial inteligence? 5](#_Toc510860204)

[Wat voor machine learning methodes zijn er? 6](#_Toc510860205)

[Wat is het verschil tussen Supervised learning en Reinforcement learning? 7](#_Toc510860206)

[Welke machine learning algoritmes sluiten goed aan op een zelf lerende speler in een computer spel? 8](#_Toc510860207)

[Voor welke doeleinden wordt machine learning zowel gebruikt? 13](#_Toc510860208)

[Conclusie 14](#_Toc510860209)

## Inleiding

In dit onderzoek probeer ik één hoofdvraag te beantwoorden. Dit is “Welk machine learning algoritme kan ik het best gebruiken binnen mijn project”. Wat mijn project inhoud staat gedefinieerd in het concept document. [[1]](#footnote-1) Bronnen zullen vermeld worden in voetnoten. Daarnaast zal deze hoofdvraag beantwoord worden in de conclusie aan het eind van dit document.

## Welk machine learning algoritme kan ik het best gebruiken binnen mijn project?

Om deze vraag te beantwoorden wordt deze opgedeeld in deel vragen. Onder al deze documenten zal een conclusie komen te staan waarin de informatie van de deelvragen wordt samen gepakt. In deze conclusie zal een duidelijk antwoord te vinden zijn op deze vraag.

### Wat maakt een algoritme “machine learning”?

Machine learning is zoals de naam al doet vermoeden het principe dat een programma of machine zelf kan leren hoe hij bij een oplossing komt. De focus van machine learning ligt vooral in het zo efficiënt mogelijk verwerken van zijn input data tot antwoord. Voor machine learning is het proces om tot het antwoord te komen echter niet gedefinieerd. De machine moet zichzelf dit dus aanleren. Een goed voorbeeld wat laat zien wat machine learning precies doet, en wat het kan bereiken is een evolution simulator. [[2]](#footnote-2) [[3]](#footnote-3)

Er zijn twee belangrijke principe’s als het om machine learning gaat. [[4]](#footnote-4)

1. De ontwikkelaar kan tijdens het maken van het programma niet negeren dat informatie die geleverd wordt aan het pogramma foutief kan zijn. Daarnaast moet de ontwikkelaar letten op geheugen management en performance.
2. Machine learning heeft velen velden waarin het toegepast kan worden. Dit kunnen dingen zijn zoals: diagnoses samen stellen, plannen, handelen op een beurs, het spelen van een game, het leren van een taal en nog veel meer. Daarom is Machine learning niet alleen de basis voor Artificial Inteligence (AI), maar ook een belangrijk deel voor research and development en data wetenschappen.

#### Conclusie

Het eerste punt is belangrijk om in het zicht te houden, maar niet extreem schokkend. Hier moet je tijdens het programmeren van andere programma’s immers ook rekening mee houden. Zelfs in het dagelijks leven zal iedereen er tegen aan lopen dat informatie die ze ontvangen niet altijd klopt, en dit moet goed afgehandeld worden. Wat het geheugen management en de performance betreft, moet een ontwikkelaar hier ook altijd op letten. Anders zal je programma’s maken die niet werken naar behoren, of zo traag werken dat ze niet te gebruiken zijn. Voor machine learning zal dit vooral impact hebben op de performance van het programma. Als het programma de berekening namelijk niet op tijd kan doen, zal zijn data nooit betrouwbaar zijn en zal hij hier zelf nooit van kunnen leren.

In mijn project ga ik machine learning erg doelgericht inzetten. Ik ga machine learning gebruiken om een NPC (non playable character) zichzelf aan te leren om een spel te spelen. Ik ga machine learning dus niet toepassen in andere velden zoals R&D of datawetenschappen zoals genoemd als mogelijkheid in het tweede punt.

### Wat is het verschil tussen machine learning en artificial inteligence?

Zoals eerder vermeld is machine learning dus een onderdeel van artificial Inteligence. De twee onderwerpen lijken echter relatief veel op elkaar, en worden te pas en te on pas met elkaar verwisseld. Machine learning en artificial inteligence hebben allebei een focus op het maken van een machine die data kan analyseren, en hier zijn eigen conclusies van kan trekken. Met andere woorden wordt er in beide gebieden een apparaat gemaakt dat (de illusie wekt dat het) zelf kan leren.

Sinds ongeveer 1950 zijn de velden echter uit elkaar gegroeid. Machine learning heeft zich namelijk meer gericht op het analyseren en toepassen van data en wiskundige functies, terwijl AI in een bredere zin verder gegaan is. AI focust namelijk niet alleen op het analyseren van data, maar ook op het menselijk maken, en gebruik van de uitkomsten van deze analyse.

AI heeft dus minder diepgang dan machine learning op het gebied van data-analyse, maar gaat steeds verder de breedte in op andere gebieden zoals spraak, en creativiteit. AI maakt wel vaak gebruik van machine learning. Een machine learning programma kan namelijk leren van zijn eigen resultaten, en conclusies trekken uit groten hoeveelheden data. Deze analyse heeft AI ook nodig in al zijn oplossingen. Als dit niet voorkomt is het programma niet langer AI, maar gewoon een sequentieel ontworpen stuk code.

Een aantal doelen waar AI zich op kan focussen zijn:

1. Logica
2. Kennis en intelligentie
3. Plannen
4. Taal
5. Creativiteit
6. Perseptie
7. Machine learning (geheugen)

Hierin is dus te zien dat Machine learning niet alleen een essentieel deel is van AI, maar ook een van de doelen is die AI probeert te bereiken. Met dit doel wordt bedoeld dat AI van zichzelf probeert te leren. Zijn resultaten en mogelijke andere data worden gebruikt om te kijken of hij zijn aanpak in bijvoorbeeld het leren van een taal moet aanpassen. Het leren van zijn eigen fouten wat een AI hier doet is niet alleen deel van de AI, maar is dus eigenlijk alles waar machine learning voor staat. [[5]](#footnote-5)

#### Conclusie

Machine learning en AI groeien als maar verder uit elkaar. Machine learning focust veel op het verbeteren van zijn conclusies, die hij met behulp van wiskundige functies uit zijn data haalt, terwijl AI veel verder gaat op het menselijk maken van zijn resultaten. AI kan echter niet bestaan zonder machine learning.

### Wat voor machine learning methodes zijn er?

Machine learning algoritmes zijn op te delen in 3 categorieën. Deze categorieën zijn “Supervised Learning”, “Unsupervised learning” en “Reinforcement learning”.

#### Supervised learning

In een supervised learning algoritme is er een duidelijke uitkomst. Deze uitkomst wordt berekend met een aantal voorspellende variabelen, die vervolgends weer input pakken van input variabelen. Tijdens het trainen van het programma worden de functies die achter de voorspellende variabelen zitten constant geweizicht. Dit gaat door tot het programma consequent genoeg het goede antwoord berekend. Wanneer de training fase van het algoritme voorbij is hoeft het niet meer te leren. Het doel is bereikt. Het algoritme krijgt input waardes, waar hij de goede conclusie uit trekt en deze heeft hij als output waarde terug. Een goed voorbeeld van zo’n algoritme is een Neural network. Hieronder vallen ook een Dicision Tree, Random Forest, KNN, Regression, etc.

In supervised learning veranderen de voorspellende waarden niet op basis van de output die het programma geeft. Dit is bij Reinforcement Learning wel het geval.

#### Unsupervised learning

Unsupervised Learning is bedoeld om een cluster aan data in verschillende groepen te verdelen. In dit soort machine learning werk je dus niet naar één specifiek doel. Dit wil zeggen dat het algoritme geen output variabele heeft. Het algoritme analyseert zijn data, en verdeeld deze op hoe verschillend het is ten opzichte van elkaar. Voorbeelden van unsupervised learning zijn het Apriori algoritme en K-means.

#### Reinforcement Learning

In Reinforcement learning wordt het programma getraind om specifieke antwoorden te geven. Deze antwoorden worden gebaseerd op zo veel mogelijk relevante kennis die het programma heeft. Deze kennis komt niet alleen uit databases, maar in de meeste gevallen voornamelijk uit eerdere ervaringen met een gelijke soort input. Deze manier van machine learning verschilt dus van Reinforcement learning en Supervised learning lijken op elkaar in het opzicht dat ze allebei naar een gedefinieerd doel werken, maar ze verschillen van elkaar omdat Reinforcement learning leert van zijn vorige ervaringen en hiermee consequent tot het goede antwoord probeert te komen, terwijl Supervised learning vooral leert van patronen die voorkomen in de input. Een voorbeeld van Reinforcement learning is Markov Decision Process. [[6]](#footnote-6)

### Wat is het verschil tussen Supervised learning en Reinforcement learning?

Supervised learning leert van de zelfde input waardes. Als hij dus drie keer een input waarde krijgt waar hij op moet reageren met een specifiek antwoord zal hij leren dat hij de volgende keer dit zelfde specifieke antwoord moet geven. De input data verandert **niet** ten gevolg van het gegeven antwoord.

In Reinforced learning leert het programma meer van zijn vorige ervaringen met de input. Bij Reinforced learning heeft de output effect op nieuwe input waardes. Hierdoor leert hij van zichzelf, maar is Reinforced learning ook een stuk trager dan Supervised Learning. Bij Supervised learning weet het programma gelijk of hij de goede keuze gemaakt heeft of niet. Bij reinforced learning is dit echter niet het geval. Omdat Reinforced learning leert van zijn eerdere ervaringen kunnen er verschillende stappen leiden tot een ultiem antwoord. Alle stappen in de goede volgorde worden gekoppeld aan dit ultieme antwoord. Dit antwoord kun je ook wel zien als beloning of straf voor het opstellen van de goede sequentie van antwoorden. Reinforced learning is daarom dus ook ook trager dan Supervised learning.

Het grootste verschil is dus dat Supervised learning leert van zijn inputs en voor elke keer dat hij draait een antwoord krijgt of hij het goed gedaan heeft of niet. Reinforced learning werkt met de zelfde feedback, maar leert vooral van vorige ervaringen, waardoor hij pas een antwoord krijgt of hij het goed gedaan heeft na een sequentie van goede antwoorden te geven. [[7]](#footnote-7) [[8]](#footnote-8)

### Welke machine learning algoritmes sluiten goed aan op een zelf lerende speler in een computer spel?

Er zijn verschillende soorten machine learning algoritmes met verschillende doel einden. Het is belangrijk om algoritme te kiezen dat bij je probleemstelling past. Een aantal voorbeelden waar we op in zullen gaan zijn:

* Random Forest (een verzameling van meerdere Decision trees)
* Linear of Logistic Regression
* Markov Decision Process (neural network)

Er zijn natuurlijk nog veel meer algoritmes te vinden op het internet die voor verschillende doeleinden vaak gebruikt worden. [[9]](#footnote-9)

#### Random Forest - Decision trees

Een random Forest is een verzameling van verschillende Decision Trees. Het voordeel hiervan is dat je zo veel mogelijk fouten afhandeld. Hierin kun je eerst af filteren welke decision tree of decision trees je wil gebruiken. Het voordeel van meerder decision trees in plaats van een grote decision tree is dat je ze op verschillende volgordes achter elkaar kunt gebruiken, of kunt kiezen om sommige trees helemaal niet te gebruiken.

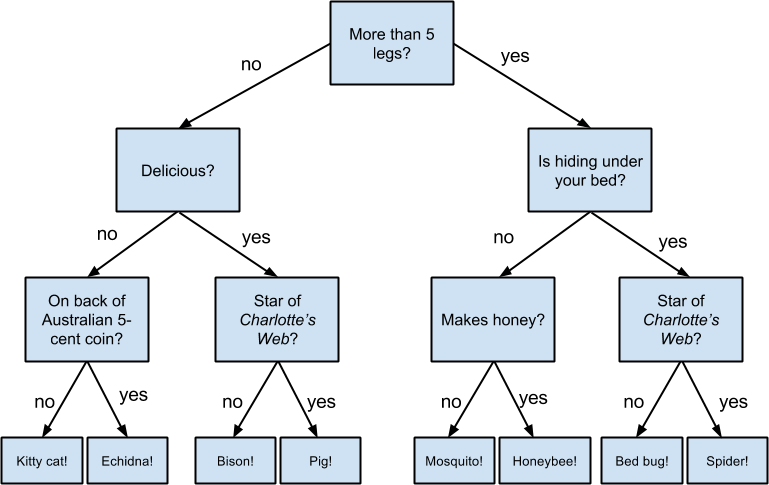
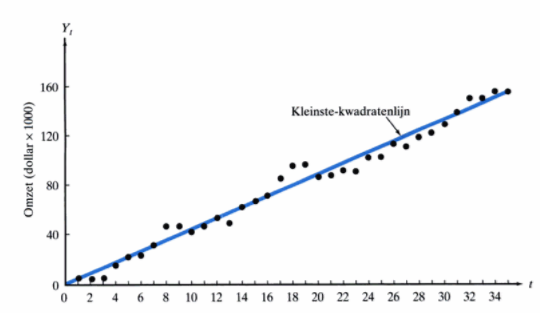
Een decision tree valt onder de methode Supervised machine learning. Een decision tree bouwt in zijn leaning phase zijn boom op. Dit doet hij door zo veel mogelijk samples te vergelijken. Deze samples hebben de juiste ouput-waarde bij hun input-waardes. Met deze waardes bouwt hij een decision tree op. Dit is niets meer dan een stappenplan wat je kan volgen om op een bepaald antwoord uit te komen. Dit algoritme kan nadat deze fase compleet is natuurlijk ook data vergelijken die niet in zijn eerste data cluster heeft gezeten. Als hij de output voor deze data nog nooit gezien heeft zal hij echter op een verkeerd antwoord uitkomen met data die voor de decision tree wel bekend is.

Figure 1: Decision tree om te bepalen wat voor dier iets is.

Als je in het voorbeeld hier rechts een struisvogel in zal vullen, en je volgt de decision tree zul je uit komen op het antwoord: “Kitty cat!”. [[10]](#footnote-10) [[11]](#footnote-11)

#### Linear Regression

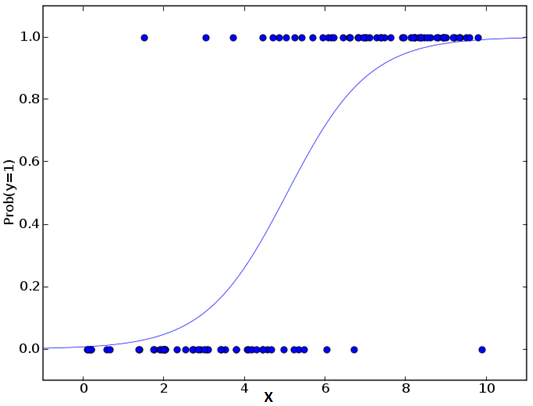
Met linear regression wordt data vergeleken op twee verschillende variabelen die afhankelijk zijn van elkaar. Door deze data te vergelijken op het assen stelsel wordt er een verband gelegd tussen deze twee variabelen. [[12]](#footnote-12) [[13]](#footnote-13)

Wat opvalt in de grafiek die je hier rechts ziet is dat dat parameters in verband staan met elkaar. Dit is nodig voor linear regression. Hieruit wordt namelijk een lijn gehaald voor de gemiddelde waardes. Een programma dat hier gebruik van maakt kan leren welke variabelen in relatie staan tot elkaar, en vervolgens een inschatting maken wat de onbekende waarde is wanneer de andere gegeven wordt. Wanneer het programma voor deze grafiek bijvoorbeeld t = 72 krijgt, zal het de conclusie trekken dat Y = 320.

Dit wordt dus vaak gebruikt als er op basis van veel data ingeschat moet worden wat andere data inhoud. Het voordeel in verhouden met een decision tree is dat je hiervoor data kunt gebruiken waar nooit op getest is. Het nadeel is wel dat het resultaat niet accuraat hoeft te zijn, als de relatie tussen de variabelen (t en Y) niet overeen komt met de relatie die de variabelen hadden binnen de testdata.

In het kort maakt Linear Regression gebruik van relaties tussen twee variabelen. Deze relatie wordt gebruikt voor interpolatie en extrapolatie. Om een inschatting te maken voor nieuwe of incomplete data.

#### Logistic Regression

Logistic Regression is fundamenteel anders dan linear Regression. In Logistic Regression wordt data net als mij Linear regression op een formule gemapt. Echter heb je hier geen output variabele van een integere waarde, maar een uitkomst die alleen Waar of Niet waar kan zijn. Uit Logistic Regression komt een 1 of een 0. Met Logistic Regression wordt namelijk bepaald hoe groot de kans is dat de inputwaardes Waar, of Niet waar aan moeten geven. [[14]](#footnote-14)

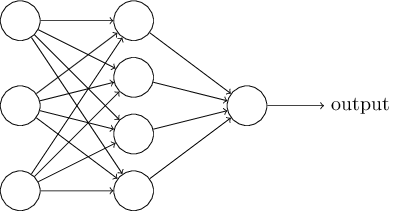
Voor machine learning werkt dit dus ook met Supervised learning. In de leer fase van het programma leert het hoe groot de kans is dat bepaalde input variabelen gecombineerd uitkomen op Waar en hoe groot de kans is dat deze variabelen uitkomen op niet waar. Hiermee wordt een logaritmische functie opgesteld. Deze wordt hierna gebruikt om input waardes te mappen, en een uitkomst te geven. Dit kan gebruikt worden voor een kans berekening. Dit is handig voor bijvoorbeeld een verzekering. Deze kan zijn premie verhogen als hij ziet dat de klant door zijn omstandigheden zoals “roker” en “in een mijn gewerkt” meer kans heeft op een ziekte.[[15]](#footnote-15)

#### Markov Decision Process – Neural Network

In het kort is dit beslissings process een reinforced machine learning protocol waarin we de meest efficiente oplossing zoeken door middel van trial and error. Ook is het een soort van devide and qonquer algoritme. Dit is duidelijk uit te leggen aan de hand van een voorbeeld.[[16]](#footnote-16)

Laten we zeggen dat je een programma wilt schrijven dat Flappy Bird kan spelen. Om dit te doen hebben we kennis nodig van wat zich afspeelt op het scherm. We moeten weten waar de onderste balk eindigd, en de bovenste begint zodat het programma kan zien waar flappy bird door heen moet. Daarnaast moeten we weten hoe hoog flappy bird vliegt. De hoek waarin hij vliegt ten opzichte van een horizontale lijn is niet van belang, omdat het programma zelf gaat bedenken wat de handigste oplossing is. Hier hoeven we zelf dus niet meet te rekenen. Daarnaast willen we de score weten, zodat het programma kan zien hoe goed hij het gedaan heeft. Als laatste willen we weten wanneer flappy bird dood is, zodat het programma niet verder hoeft te kijken. Dit is te doen door te kijken naar de score. Als deze te lang niet veranderd is flappy bird dood.

De output variabele voor het programma is vrij simpel in dit geval. Dit zijn de acties die flappy bird kan doen. In dit geval is dat “Springen”.

De input en output variabelen kunnen we zien als neuronen. Hier zijn de inputs de linker rij bolletjes. In deze neuronen staan functies om de input waardes te transformeren in waardes waar het programma mee kan rekenen. Deze waardes worden door de tweede laag gebruikt om een uitkomst te berekenen. De tweede laag is ook wel een “hidden layer”. In een complexer netwerk kunnen dit meerdere lagen zijn. Deze laag berekent met verschillende formules de waarde voor de output variabelen. In ons geval is dit er één maar dit kunnen ook meerdere variabelen zijn. Deze wordt in zijn eindwaarde neuroon omgezet in een variabelen waar de game, flappy bird, dan weer iets mee kan.

De “hidden layer” is een wat lastigere laag. Dit komt omdat je hier in principe alles mee kunt doen wat je wil, als je maar door middel van de input bij de output uit kan komen. Hier zul je zien dat dit algoritme een devide en qonquer algoritme is. Elke node in deze laag of lagen heeft namelijk een functie. Meestal zijn dit vrij eenvoudige formules, zoals een lineaire functie. Al deze functies zijn verschillend, en zijn gevoeliger of minder gevoelig voor de input neuronen in verhouding tot elkaar. Dit wil dus zeggen dat Hidden neuroon 1 gevoeliger kan zijn voor input waarde 1, en minder gevoelig voor input waarde 2 dan neuroon 2. In deze hidden neurons staan vervolgens formules om tot output waardes te komen. Het output neuroon vertaalt deze naar echte output. Deze kan ook gevoeliger zijn voor sommige neuronen dan voor anderen. [[17]](#footnote-17)

Dit algoritme wordt gebruikt voor path-finding in games of het analyseren en conclusies trekken uit data zoals beeld of geluidsfragmenten. Hierdoor zul je als je neural netwerk zoekt veel vinden over object of gezichtherkenning, maar ook over evolution simulators en games.

### Voor welke doeleinden wordt machine learning zowel gebruikt?

In het onderzoek tot nu toe is al opgevallen dat dit een erg brede vraag is. Machine learning wordt voor veel dingen gebruikt. Dit gaat van deep learning algoritmes om video beelden te analyseren tot een AI maken in een game tot weer het leggen van linken in data in een database.

Machine learning wordt voor van alles gebruikt. Hierdoor zijn er zo veel verschillende methodes. Het is belangrijk om een methode te kiezen die past bij het probleem dat je op wil lossen omdat sommige methodes zo fundamenteel verschillend zijn. Hierin is de vraag “hoe wil ik dat mijn programma leert?” het belangrijkst met daarop de antwoorden “reinforced machine learning”, “supervised machine learning” en “unsupervised machine learning”.

Machine learning wordt tegenwoordig overal toegepast. Als je er op zoekt zul je al snel uitkomen op bedrijven zoals google, wat overal projecten in heeft. Machine learning wordt echter ook toegepast bij overheid instanties, banken, Medische diensten zoals ziekenhuizen, marketing en transport. Hier worden natuurlijk erg verschillende algoritmes gebruikt. [[18]](#footnote-18)

Google gebruikt machine learning bijvoorbeeld om je zoekresultaten te analyseren, en adds te laten zien die hiervoor relevant zijn. Ook is dit essentieel voor het youtube algoritme. Door machine learning wordt bepaald wat gebruikers waarschijnlijk willen zien. Hiernaast past google machine learning toe op een erg interessante manier. Google heeft namelijk zijn eigen servers geanalyseerd, waardoor deze nu efficiënter kan draaien. [[19]](#footnote-19)

### Conclusie

Onder de vraag “wat voor machine learning methodes zijn er?” worden Supervised learning, unsupervised learning en Reinforced learning toegelicht. Omdat Supervised Learning en Reinforced Learning in sommige gevallen op elkaar kunnen lijken is het verschil nog een keer benadrukt in “Wat is het verschil tussen Supervised Learning en Reinforced Learning”.

Voor mijn programma is de beste mogelijkheid Reinforced learning. Deze keuze maak ik omdat ik wil kunnen zien hoe de tegenstander in het spel steeds beter wordt. Hier mag hij meerdere potjes over doen, en ik wil zien dat hij zelf uitzoekt wat de beste manier is om tot zijn oplossing te komen.

Daarnaast kies ik voor het Neural network. Dit algoritme is misschien minder handig voor mijn probleem dan een decision tree. Echter neem ik in deze overweging mee dat het Neural network mij interessanter lijkt en ik hier mogelijk meer van kan leren. Daarnaast is er over een neural network erg veel informatie te vinden. Ook zijn er veel voorbeelden van mensen die soortgelijke problemen hebben opgelost met een neural network. Hiermee erken ik dat het probleem dus makkelijker en mogelijk beter of overzichtelijker op te lossen zou moeten zijn met een decision tree. Ik kies echter voor de optie waar ik denk het meest plezier in te zullen hebben.

Voordelen van een decision tree zijn dat er geen formules toegepast hoeven te worden. Hier hoeven alleen maar vergelijkingen gemaakt te worden. Voor elke nieuwe situatie kan een nieuwe vergelijking opgesteld worden, en zo leert het programma zichzelf foutloos te spelen. In het neural network zitten niet alleen vergelijkingen, maar ook afhankelijkheden en formules. Dit geeft meer vrijheid, maar maakt het algoritme complexer en minder overzichtelijk terwijl het hetzelfde doel zal behalen.

1. <https://github.com/MichielvdBroek/machineLearing/> “Documentatie/Concept.docx” [↑](#footnote-ref-1)
2. <https://www.openprocessing.org/sketch/377698> evolution simulator. [↑](#footnote-ref-2)
3. <https://www.youtube.com> “evolution simulator (part 1/4)”. [↑](#footnote-ref-3)
4. <https://books.google.nl/books> “Elements of Machine learning” hoofdstuk 1 pagina 1 [↑](#footnote-ref-4)
5. <https://www.digitaldoughnut.com/> “The difference between AI and machine learning” [↑](#footnote-ref-5)
6. <https://www.analyticsvidhya.com/> “Essentials of Machine learning Algorithms (with Python and R Codes)” [↑](#footnote-ref-6)
7. <https://stackexchange.com> “Supervised learning, unsupervised learning and reinforcement learning: Workflow basics” [↑](#footnote-ref-7)
8. <https://www.quora.com> “What is the difference between supervised learning and reinforcement learning?” [↑](#footnote-ref-8)
9. <https://www.analyticsvidhya.com/> “Essentials of Machine learning Algorithms (with Python and R Codes)” [↑](#footnote-ref-9)
10. <https://towardsdatascience.com> “what is a decision tree?” [↑](#footnote-ref-10)
11. <https://www.youtube.com/> “Visualizing a Decision Tree - Machine Learning Recipes #2” [↑](#footnote-ref-11)
12. <https://www.analyticsvidhya.com/> “Essentials of Machine learning Algorithms (with Python and R Codes)” [↑](#footnote-ref-12)
13. <https://books.google.nl/books> “statistiek. een inleiding voor het hoger onderwijs” – James T. McClave, Terry Sincich Hoofdstuk 15.7 [↑](#footnote-ref-13)
14. <https://www.analyticsvidhya.com/> “Essentials of Machine learning Algorithms (with Python and R Codes)” [↑](#footnote-ref-14)
15. <https://www.let.rug.nl/nerbonne/teach/stats/Moore-McCabe-H15.pdf> [↑](#footnote-ref-15)
16. <https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/markov_decision_process.html> [↑](#footnote-ref-16)
17. <https://stats.stackexchange.com/questions/63152/what-does-the-hidden-layer-in-a-neural-network-compute> [↑](#footnote-ref-17)
18. <https://www.sas.com/en_id/insights/analytics/machine-learning.html#machine-learning-today-world> [↑](#footnote-ref-18)
19. <https://www.quora.com/How-does-Google-use-machine-learning> [↑](#footnote-ref-19)