KU LEUVEN kulak

Groep Wetenschap & Technologie Kulak

Bachelor fysica, informatica, wiskunde
ingenieurswetenschappen

X0B53A – Probleemoplossen en ontwerpen, deel 1
X0A22D – Informaticawerktuigen

Levens redden met machine learning: Support Vector Machines voor tumorclassificatie

Eindverslag van de teamopdacht

Team ∃uler

Vincent Van Schependom, Daan Vanhaverbeke, Jasper Benoit, Lasha Shergelashvili, Marie Taillieu, Zeineb Kharbach, Florian Degraeve, Younes Mebarki

Academiejaar 2023 – 2024

Inhoudsopgave

Inleiding

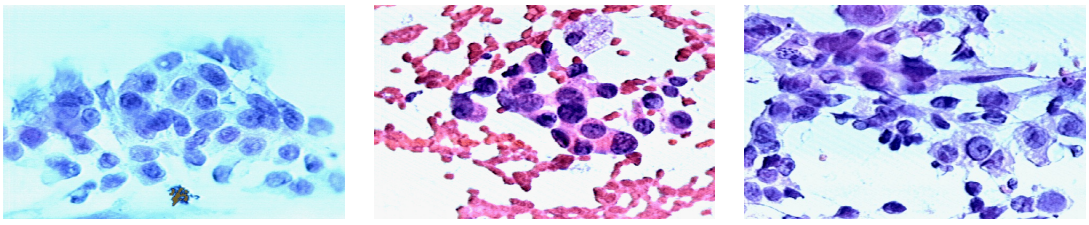
Dit is het eindverslag van de teamopdracht, waar we met alle leden van team ∃uler gedurende 9 weken aan gewerkt hebben. We maakten doorheen de sessies kennis met machine learning en we verwierven week na week meer en meer kennis over dit interessante onderwerp. We oefenden eerst op het onder de knie krijgen van twee basistechnieken, waarna we in week 6 begonnen met het uitwerken van onze eigen machine learning toepassing.

Inhoud van dit eindverslag

We zullen in dit eindverslag de lezer eerst laten ontdekken wat machine learning precies is. Vervolgens bespreken we wat supervised learning inhoudt, in vergelijking met unsupervised learning. We bespreken ook wat classificatie juist is. Hierna zullen we het principe achter Support Vector Machines, de toepassing waarvoor we kozen, uitleggen. Hierbij zullen we verschillende classificatietechnieken vergelijken en beargumenteren waarom voor de tumor-dataset SVM een geschikte keuze blijkt te zijn.

De gegevens

We maken in ons verslag gebruik van de resultaten van een onderzoek naar de verschillende eigenschappen van tumoren bij borstkankerpatiënten. Hierbij onderzochten wetenschappers W. Nick Street, W. H. Wolberg, en O. L. Mangasarian [1] in 1993 welke kenmerken het meest invloed hadden in het al dan niet kwaadaardig zijn van een tumor. Deze kenmerken werden berekend op basis van een gedigitaliseerd beeld van een fijne naaldaspiraat (FNA) van de borstmassa. Dergelijke beelden zijn te zien in figuur 1. De data die in het onderzoek vergaard werd, werd nadien publiek vrijgegeven [2], wat ons in de mogelijkheid stelt om de data te verwerken met hedendaagse Machine Learning technologie.



Figuur 1: Gedigitaliseerde beelden van fijne naaldaspiraten van de borstmassa.

Hoofdstuk 1

Machine Learning

1.1 Wat is machine learning?

Machine learning is een tak binnen het gebied van de artificiële intelligentie, waarbij we een model trainen op basis van een gegeven dataset. Naarmate de training vordert, zal het model bepaalde verbanden beginnen leggen en bepaalde structuren beginnen herkennen in de data waarop het getraind wordt. In principe neemt de kwaliteit van het model toe wanneer we de data, waarop het model zich baseert, toeneemt.

In ons geval zullen we dus een model trainen dat verbanden zal zoeken in de dataset met tumoren van borstkankerpatiënten. Het model zal trachten een link te vinden tussen de verschillende kenmerken van deze tumoren en de klasse waartoe deze tumoren behoren; die van de goedaardige of die van de kwaadaardige.

1.2 Terminologie en notatie

De dataset beschikt n verschillende datapunten. Deze individuele datapunten x_i hebben elk een eindig aantal kenmerken, die we *features*, *inputs* of *onafhankelijke variabelen* zullen noemen. We duiden de hoeveelheid *features* aan met p. Elk individueel datapunt x_i uit onze set van datapunten $x_1, x_2, ..., x_n$ is dus van de vorm $x_i(x_1, x_2, ..., x_p)$ (met $1 \leq i \leq n$). Deze features van de datapunten kunnen we voorstellen aan de hand van een matrix X, waarvoor geldt:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_p \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1p} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2p} \\ x_{31} & x_{32} & x_{33} & \dots & x_{3p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & x_{n3} & \dots & x_{np} \end{bmatrix}$$

Hierbij is X_{ij} de meetwaarde van de j-e feature voor het i-e datapunt met met $i = 1, 2, 3, ..., n$ en $j = 1, 2, 3, ..., p$. De tumordataset bevat gegevens over 569 tumoren, die elk 30 kenmerken hebben. In de dataset is dus $n = 569$ en $p = 30$.

Aangezien we aan *supervised learning* doen (hier gaan we in Hoofdstuk 2 verder op in), hebben we voor elke input x_i in de dataset ook een bijhorende *output* y_i. We hebben dus n y-waarden $y_1, y_2, ..., y_n$, die we - net zoals de features - matricieel kunnen voorstellen als volgt:

$$Y = \begin{bmatrix} y_1 & y_2 & y_3 & \dots & y_n \end{bmatrix}$$

Elke tumor is ofwel kwaadaardig, ofwel goedaardig, dus elke individuele output y_i is ofwel gelijk aan 1 (goedaardig), ofwel gelijk aan -1 (kwaadaardig).

1.3 Predictie

Na het trainen van ons model, hebben we hopelijk een verband gevonden tussen de verschillende *features* van de tumoren en hun aard. Indien we ook het validatieproces (zie 4.3) doorlopen, kunnen we ons model toepassen op een nieuwe *batch* datapunten om zo voorspellingen te kunnen doen over de aard van deze - door het model nog niet eerder geziene - tumoren. We noemen dit *predictie*.

Hoofdstuk 2

Supervised Learning

2.1 Supervised vs unsupervised

Machine learning kan worden opgesplitst in twee grote categorieën; *supervised learning* enerzijds en *unsupervised learning* anderzijds. De naam van beide vormen geeft eigenlijk al weg wat deze vormen precies inhouden.

Bij *unsupervised learning* hebben we voor elk datapunt in onze dataset zowel features als outputs voor handen. Er is sprake van zogenaamde 'input-outputparen'. We kunnen ons model trainen op het vinden van een verband tussen de x-waarden en de y-waarden in onze trainingsdataset. Het uiteindelijke doel is dan dat het model voor een nieuwe dataset, op basis van de features in die dataset, een output zal voorspellen.

Bij *unsupervised learning* zal de data enkel uit inputs of x-waarden bestaan en hebben we dus geen afhankelijke variabelen. Een *unsupervised learning*-model zal zelf patronen of structuren trachten te vinden in de data en vervolgens de data zelf onderverdelen in klassen. De details achter *unsupervised learning* liggen buiten de scope van dit rapport en zullen we dus achterwege laten.

2.2 Vormen van supervised learning

Supervised learning kan nog verder gesplitst worden in twee categorieën; de twee voornaamste vormen van *supervised learning* zijn regressie en classificatie.

Bij regressie zijn de outputs altijd getallen. Deze numerieke outputs kunnen eender welke waarde aannemen. We kunnen hierbij dus spreken van *kwantitatieve data*. Het getrainde model zal trachten een continu numeriek resultaat te voorspellen voor de output van een nieuw datapunt - waarop het model dus niet getraind is.

Dit staat in tegenstelling met classificatie, waar de outputs enkel een telbaar aantal waarden kunnen aannemen. We zullen in hoofdstuk 3 classificatie in meer detail toelichten.

Hoofdstuk 3

Classificatie

3.1 Wat is classificatie?

Classificatie in machine learning is een vorm van supervised learning waarbij het doel is om een input toe te wijzen aan een van de vooraf gedefinieerde klassen. De klassen worden vaak ook aangeduid als *label* of *categorie*. Het ultieme doel van classificatie is om te bepalen in welke categorie nieuwe gegevens zullen vallen.

We passen dit even toe op onze tumordataset. We willen aan de hand van classificatie onderzoeken of wat de aard van een tumor is. Er zijn dus twee mogelijke categorieën waartoe een tumor behoort: die van de goedaardige tumoren of die van de kwaadaardige. Omdat er slechts twee mogelijke klassen zijn, is er sprake van *binaire classificatie*. Er bestaan ook echter ook classificatietechnieken waar meerdere klassen aan bod komen.

3.2 Enkele toepassingen

- **Beeldherkenning:** Classificatie wordt vaak gebruikt in beeldherkenningstoepassingen, zoals het identificeren van objecten in foto's of video's.
- **Tekstclassificatie:** Hier wordt classificatie gebruikt om tekst te categoriseren, bijvoorbeeld het identificeren van spam-e-mails of het toewijzen van artikelen aan specifieke onderwerpen.
- **Medische diagnose:** Naast het eerder genoemde voorbeeld van tumorclassificatie, wordt classificatie ook toegepast op het diagnosticeren van andere medische aandoeningen op basis van verzamelde gegevens.

Hoofdstuk 4

Het model

4.1 Bias

In machine learning verwijst bias naar de systematische fout die door een model wordt geïntroduceerd wanneer het de onderliggende patronen in de trainingsgegevens consequent verkeerd weergeeft. Dit kan leiden tot onnauwkeurige voorspellingen en onbetrouwbare prestaties op nieuwe datasets, zoals de validatie- en testdatasets.

Wanneer we bias bespreken in de context van classificatie, verwijzen we vaak naar de neiging van het model om bepaalde klassen boven andere te bevoordelen. Er zijn dan twee belangrijke soorten bias die we willen toelichten.

4.1.1 *Algorithmic bias*

Algorithmic bias doet zich voor wanneer het algoritme zelf is getraind op een manier die bepaalde uitkomsten of groepen bevoordeelt. Het wordt bepaald door de gegevens die worden gebruikt voor training, en de keuze van het type model. Als een classificatiemodel bijvoorbeeld wordt getraind op vertekende gegevens die overwegend één groep vertegenwoordigen, kan het model leren nauwkeuriger te zijn voor die groep, terwijl het slecht presteert voor andere groepen.

Als we ons SVM model bijvoorbeeld bijna uitsluitend trainen op gevallen waar de tumor kwaadaardig is, zal het hoogstwaarschijnlijk niet goed presteren voor nieuwe datapunten waar de tumor goedaardig is. Het model is namelijk niet getraind op tumoren van die aard en zal dus geen nauwkeurige predicties kunnen doen.

4.1.2 *Sampling bias*

Sampling bias treedt op wanneer de trainingsgegevens niet representatief zijn voor de populatie waarnaar ze willen generaliseren. Dit kan gebeuren als bepaalde groepen in de populatie onder- of oververtegenwoordigd zijn in het trainingspakket. Als we ons SVM model bijvoorbeeld enkel trainen op tumoren van vrouwelijke borstkankerpatiënten, is de kans dat het model slecht presteert voor een nieuwe dataset van mannelijke patiënten vrij groot.

4.2 Variantie

De variantie is de maat voor de gevoeligheid van het model. Het geeft inzicht over de flexibiliteit van een model, met name hoe nauwkeurig de voorspellingen zijn bij verschillende datasets.

Bij een hoge variantie is het model sterk aangepast aan de trainingsdata. Dit betekent dat zelfs kleine veranderingen in de trainingsdata een grote invloed zullen hebben op het model, wat resulteert in een fenomeen genaamd *overfitting*. In het geval van tumorclassificatie zou een hoge variantie betekenen dat het model zeer nauwkeurig kan voorspellen of een tumor goedaardig is bij de trainingsdata. Maar doordat het model te sterk is afgestemd op de trainingsdata, zal het moeite hebben met generaliseren en levert dit geen precieze voorspellingen op voor de andere datasets, met name de validatie- en testdatasets.

Aan de andere kant kan een model ook een te lage variantie hebben. Dit betekent dan dat het model te veel generaliseert en weinig verandert indien de trainingsdata verandert. Een te lage variantie leidt echter tot *underfitting*. Hierdoor levert het ook geen betrouwbare voorspellingen op. Toegepast op tumorclassificatie zal een te lage variantie leiden tot een te sterk gegeneraliseerd model, waarbij het model niet goed presteert op de 3 verschillende datasets.

4.3 Flexibiliteit van het model bepalen

Hoe bepalen we nu de ideale flexibiliteit van ons model? Met andere woorden: hoe voorkomen we zowel *overfitting*, waarbij het model de trainingsdata te nauwgezet probeert te volgen, als *underfitting*, waarbij er sprake is van overgeneralisatie?

4.3.1 Metaparameters

Machine learning modellen hebben over het algemeen een of meerdere *metaparameters* die de flexibiliteit van het model beïnvloeden. Het model zal zich meer of minder aanpassen naar gelang de waarden van deze metaparameter(s). Zoals we later zullen zien in hoofdstuk 6, is een zekere λ de metaparameter die bij Support Vector Machines van belang is. Deze parameter wordt voor SVM ook wel de *regularisatieparameter* genoemd. Deze λ zal uiteindelijk de breedte van de marge - en dus de variabiliteit van het model - beïnvloeden (zie later).

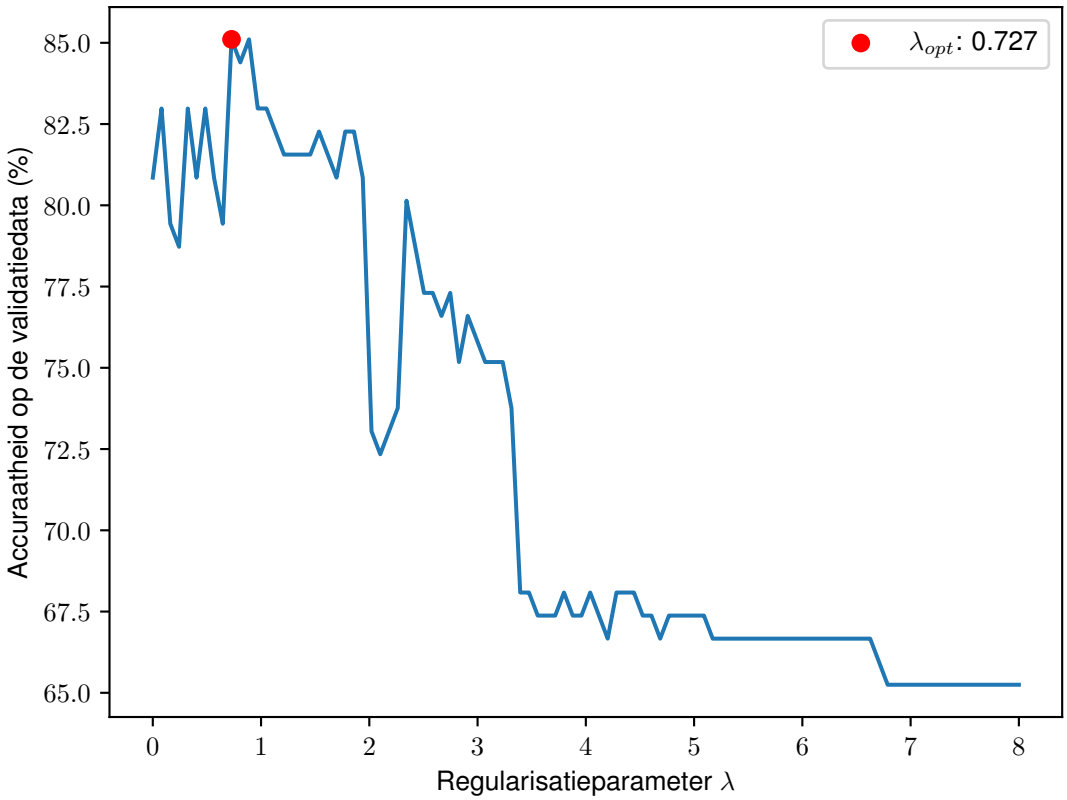
4.3.2 Cross-validation

Om de optimale metaparameters te bepalen, gebruiken we het principe van *cross-validation*. Hiervoor splitsen we eerst de data die we voor handen hebben op in 3 delen: 50% trainingsdata, 25% validatiedata, 25% testdata.

De eerste stap is het trainen van ons model op de trainingsdata met een bepaalde waarde voor de metaparameter(s). Hierna zullen we de accuraatheid van dit model voor die bepaalde metaparameter(s) testen op de validatiedataset. Dit proces herhalen we voor - liefst zo veel mogelijk - verschillende waardes van de metaparameter(s), tot we de grootste accuraatheid van de voorspelling voor de trainingsdata hebben verkregen. De metaparameter(s) die voor de grootste accuraatheid zorgen op de trainingsdata, zullen we dan beschouwen als 'optimaal'.

We kunnen de accuraatheid van het getrainde model t.o.v. de trainingsdata uitzetten in functie van een metaparameter in een grafiek, zoals te zien is in figuur 4.1. De waarde van de metaparameter waarvoor de grafiek die de accuraatheid weergeeft een maximum bereikt, zal de optimale waarde voor deze metaparameter zijn.

Tot slot gebruiken we onze testset om de algemene prestatie van ons model te beoordelen. Het uiteindelijke doel is om een zo goed mogelijke voorspelling te kunnen maken, gegeven een nieuwe set tumoren. We willen dus op basis van de 30 kenmerken de aard van nog niet eerder geziene tumoren voorspellen.



Figuur 4.1: De accuraatheid van het model t.o.v. de trainingsdata, uitgezet in functie van de metaparameter λ voor ons SVM-model.

Hoofdstuk 5

Features

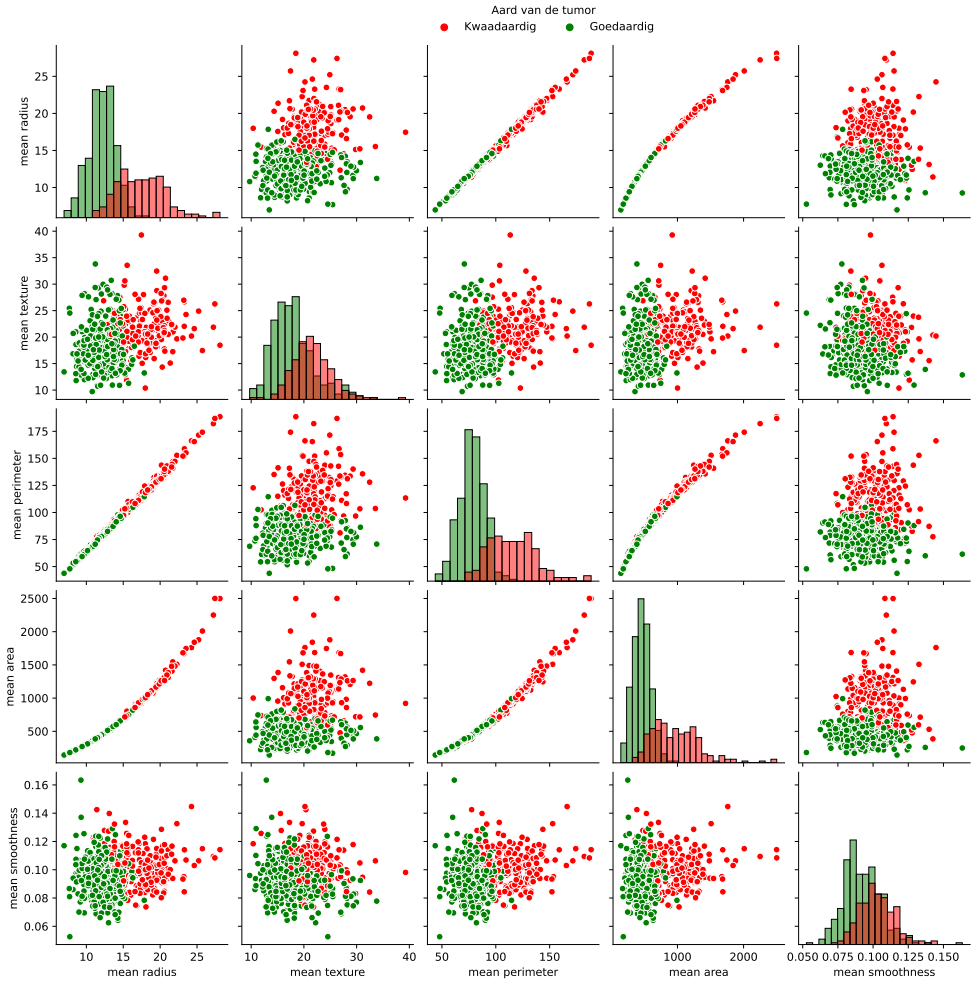
5.1 Overzicht

Zoals in de inleiding reeds vermeld werd, bestaat onze dataset uit een 500-tal datapunten met elk 30 features. Hieronder lijsten we de eerste features op om de lezer een idee te geven van de data die voor handen is:

- Straal (gemiddelde van afstanden van het middelpunt tot punten op de omtrek)
- Textuur (standaardafwijking van grijswaarden)
- Omtrek
- Oppervlakte
- Gladheid (lokale variatie in straallengtes)
- Compactheid ($omtrek^2/oppervlakte - 1.0$)
- Concaafheid (ernst van concave delen van de contour)
- Concave punten (aantal concave delen van de contour)
- Symmetrie
- ...

5.2 Limitaties

Zoals we verder zullen ontdekken in hoofdstuk 6, botsen we tegen een limitatie wat betreft het aantal features die we kunnen gebruiken voor ons model. Indien we het aantal features opdrijven, worden de wiskundige berekeningen namelijk een pak complexer en zouden we dus meer tijd nodig hebben om dit helemaal uit te werken. Daarnaast wordt het visueel voorstellen van de dataset ook des te moeilijker bij een toenemend aantal features. Wegens deze twee limitaties, zullen we ons dus beperken tot slechts twee features per tumor.



Figuur 5.1: Een pairplot van de eerste 5 features in de tumordataset.

5.3 Selectie

Hoe selecteren we nu de twee interessante features? We kunnen daarvoor gebruik maken van een *pairplot*, zoals er een te zien is in figuur 5.1. Een *pairplot* is een visuele weergave van de relatie tussen paren van features in een dataset. In machine learning worden dit soort voorstellingen vaak gebruikt om patronen en verbanden tussen de verschillende features van een dataset te identificeren. Op basis van deze pairplot bepaalden wij dat *mean texture* (textuur) en *mean radius* (straal) twee interessante features waren om te bestuderen.

Hoofdstuk 6

Support Vector Machines

6.1 Inleiding

Support Vector Machines is een classificatietechniek binnen het domein van supervised learning. We zullen, gegeven een dataset met grootte n voorspellen tot welke klasse een bepaald datapunt behoort. Zo'n datapunt heeft p verschillende features $(x_1, x_2, ..., x_p)$ en één y-waarde, zijnde de bijhorende klasse. We noemen SVM ook wel een *binaire classificeerder*, aangezien er maar twee mogelijke y-waardes zijn en elk punt maar tot een van twee mogelijke klassen kan behoren.

Uit wat later zal blijken, is het heel moeilijk om datapunten met meer dan 2 features te scheiden. Vanaf hogere dimensies is het ook heel moeilijk of zelfs onmogelijke om de werking van het model te visualiseren. Daarom beperken we ons in dit eindverslag tot een dataset waarbij elk punt slechts 2 features heeft. We zullen de dataset dan opsplitsen in 2 klassen.

6.2 Het scheidingsprincipe

Het SVM model zal trachten onze datapunten te scheiden in twee klassen. In onze dataset heeft elk datapunt waarde -1 of 1, afhankelijk van de klasse waartoe het punt behoort. Indien elk punt in de dataset p verschillende features heeft, zullen er een $(p-1)$-dimensioneel objecten, genaamd *hypervlakken*, berekend worden om de dataset in twee te verdelen.

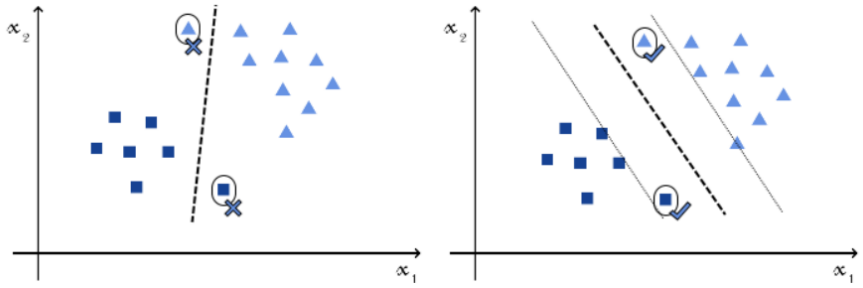
We construeren twee zo'n hypervlakken die de twee gegevensklassen scheiden, zodat de afstand daartussen zo groot mogelijk is. Het gebied dat wordt begrensd door deze twee hypervlakken wordt de *marge* genoemd, en het hypervlak ertussen noemen we de beslissingsgrens. We willen de marge zo groot mogelijk maken.

In het geval dat elk datapunt 3 features heeft, zal zo'n hypervlak een 2-dimensionaal vlak zijn. In ons geval heeft elk datapunt 2 features, dus zal de dataset kunnen gescheiden worden door een rechte, dat strikt genomen een 1-dimensioneel hypervlak is.

6.3 Waarom SVM en geen andere classificatietechniek?

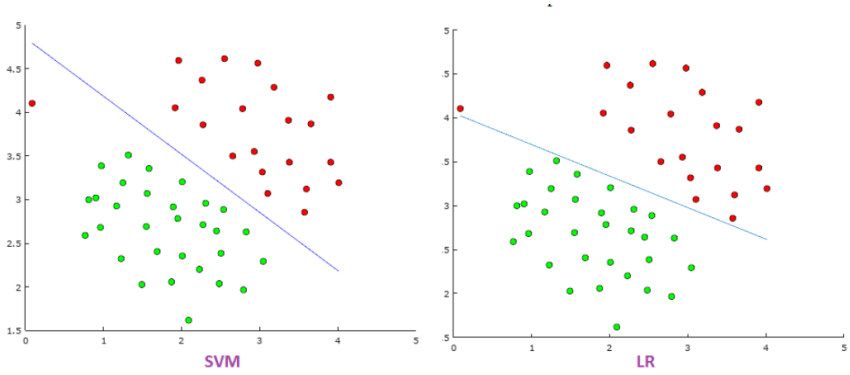
Waar Support Vector Machines in uitblinken, is het verwerken van datasets waarvan de datapunten elk zeer veel features hebben. SVM werkt namelijk uitermate goed in hoger dimensionale ruimtes, die we helaas dus niet visueel kunnen illustreren. Zo zouden we de tumoren met zeer hoge accuraatheid kunnen classificeren indien we de volledige lijst kenmerken in acht nemen.

Daarnaast is SVM minder gevoelig voor *overfitting* in vergelijking met complexere modellen zoals neurale netwerken. Dit is vooral belangrijk wanneer de data die voorhanden is, eerder beperkt is in grootte. SVM zal dus voor kleinere datasets heel goed presteren, i.h.b. als het aantal features groter is dan het aantal datapunten. Ook zal SVM in vergelijking met neurale netwerken bij het toevoegen van extra punten minder snel de mist in gaan zoals op figuur 6.1. Dit is waarom de hyperplane zo belangrijk is bij SVM.



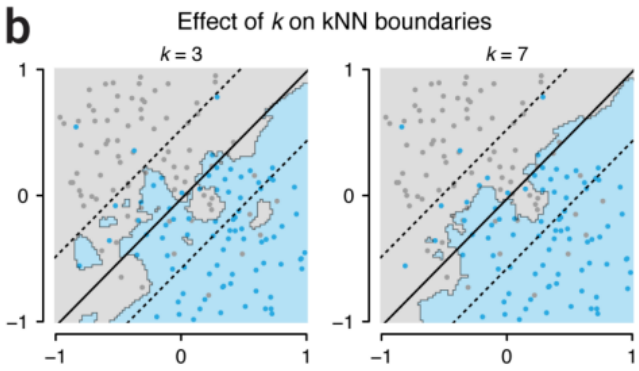
Figuur 6.1: Deze figuur uit bron [4] geeft een belangrijk verschil tussen neurale netwerken en SVM weer. Links wordt de werking van een neuraal netwerk geïllustreerd, rechts die van een SVM-model. Hierbij valt op het dat als je een nieuw punt zou classificeren, er bij neurale netwerken een veel grote kans was dat deze op het vorige model niet incorrect zou zijn, terwijl er bij SVM wel een veel grotere kans is dat het punt correct geclassificeerd zou worden. Deze figuur benadrukt dus ook het belang van de hyperplane bij SVM en zijn rol bij het voorkomen van *overfitting*.

Indien er heel veel variabiliteit is in de dataset, zal SVM ook een *edge* hebben tegenover andere classificatietechnieken: juist door het feit dat de marge tussen de scheidingshyperplanes zo groot mogelijk wordt gehouden, zullen *outliers* of andere vormen van ruis in de dataset niet in acht worden genomen. SVM is dus totaal niet gevoelig voor overfitting. Dit wordt nog eens extra benadrukt aan de hand van figuur 6.2.



Figuur 6.2: Deze figuur uit bron [5] toont de vergelijking tussen SVM en logistische regressie. Hierbij valt het duidelijk op dat wanneer er sprake is van een *outlier*, logistische regressie de mist in gaat. Dit toont dan ook aan dat Support Vector Machines veel beter bestendig zijn tegen overfitting.

Wanneer SVM in twee dimensies kan geplot worden, is deze ook zeer eenvoudig te begrijpen. Het is namelijk zeer duidelijk of een punt nu tot de ene of andere klasse behoort volgens het



Figuur 6.3: Deze figuur uit bron [6] vergelijkt KNN-classificatie met SVM-classificatie. Bij de eerste van de twee technieken, ontstaan er twee 2 gebieden, die worden bepaald door naar de k dichtste buren te kijken van elk datapunt. Op de linkerfiguur is $k = 3$, op de rechterfiguur is $k = 7$. M.b.v. *cross-validation* werd bepaald dat $k = 7$ de optimale waarde is voor de metaparameter k, met een accuraatheid van 87%. De rechte die door de grafiek gaat, is de beslissingsgrens van SVM en de stippellijnen zijn de grenzen van de hyperplanes van SVM. Hierbij valt het duidelijk op dat SVM veel makkelijker te interpreteren is in vergelijking met KNN, terwijl de zekerheid van SVM nog steeds gelijk is aan 85%. We boeten, door gebruik te maken van SVM, dus een heel klein beetje in op accuraatheid, maar dit is serieus in het voordeel van de interpreteerbaarheid van het model.

model door naar de scheidingslijn te kijken. Deze interpretatie is moeilijker bij andere classificatietechnieken zoals KNN-classificatie. Hierbij wordt er gekeken naar de K-nearest-neighbours. Door naar deze dichtste buren te kijken kan deze plot veel moeilijker begrepen worden. Er kunnen namelijk hierbij allemaal aparte wolken ontstaan van punten die tot een bepaalde klasse behoren waardoor de figuur er chaotisch zou kunnen uitzien. Dit kan je ook zien op figuur 6.3.

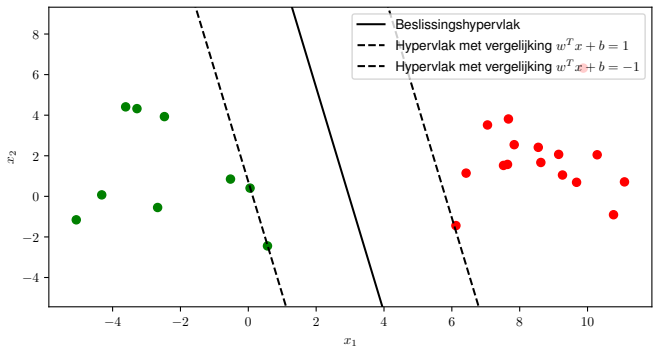
Tot slot is SVM zeer geheugenefficiënt: het model houdt enkel rekening met de dichtste punten tot de hyperplanes en zal dus, in tegenstelling tot bijvoorbeeld logistische regressie, niet met elk punt rekening moeten houden in de berekening van de beste beslissingsgrens.

6.4 Wiskundige berekening van het model

6.4.1 De hypervlakken

We stellen een voorschrift op voor de twee scheidingsrechten, die te zien zijn in figuur 6.4, waartussen zich eem marge bevindt. Als $\vec{w}$ de normaalvector is op de hypervlakken, kunnen we de vergelijkingen van deze rechten schrijven als $w^T x - b = 1$ en $w^T x - b = -1$. Hierbij is b dan de *intercept* van de rechte die de beslissingsgrens beschrijft. Alle punten boven het eerste hypervlak worden geclassificeerd als horend tot de ene klasse. De punten onder het tweede hypervlak worden geclassificeerd als horende tot de andere klasse.

De marge is het gebied tussen deze twee hypervlakken. De breedte van hiervan is gelijk aan $\frac{2}{\|\vec{w}\|}$ en aangezien we die breedte willen maximaliseren, zullen we m.a.w. dus $\|\vec{w}\|$ trachten te minimaliseren.



Figuur 6.4: Twee lineair scheidbare wolken van punten. De kleuren van de punten duiden aan tot welke klasse ze behoren.

6.4.2 Trainen van het model

Als de dataset twee lineair scheidbare 'wolken' vormt en er geen *outliers* - punten die niet tot de juiste wolk behoren - zijn, is het bepalen van de maximale marge vrij makkelijk. Het wordt moeilijker wanneer er wel *outliers* zijn en de twee wolken dus niet meer perfect lineair scheidbaar zijn, zonder dat punten aan de verkeerde kant van de twee hyperplanes belanden.

Daarom voeren we nu de *hinge loss* in. Dit is een soort foutterm die we toevoegen aan punten die niet goed of met geen grote zekerheid worden geclassificeerd. Deze *hinge loss* wordt berekend door de formule

$$\max\left(0, 1 - y_i(\vec{w} \cdot \vec{x}_i - b)\right)$$

. Deze foutterm laat ons toe om de fouten te beperken, terwijl we de marge maximaliseren. Voor elk punt hebben we dan twee mogelijkheden wat de waarde van de *hinge loss* betreft:

1. Het punt met afhankelijke variabele y_i werd correct geclassificeerd.

   Dan ligt het punt dus aan de juiste kant van een van de twee hyperplanes. In dit geval is $y_i(\vec{w} \cdot \vec{x}_i - b) \geq 1$ en zal $1 - y_i(\vec{w} \cdot \vec{x}_i - b) \leq 0$. Omdat de *hinge loss* het maximum neemt van 0 en dit laatste getal - dat ofwel ook 0, ofwel negatief is - zal de *hinge loss* in dit geval 0 zijn.

2. Het punt ligt in de marge of het punt ligt aan de verkeerde kant van de beslissingslijn.

   De *hinge loss* wordt dan $1 - y_i(\vec{w} \cdot \vec{x}_i - b)$. Indien het datapunt binnen de marge ligt, zal de hinge loss tussen 0 en 1 liggen. Het punt wordt dan met lagere zekerheid geclassificeerd, maar we willen het ook niet te hard bestraffen. Indien een datapunt buiten de marge, maar aan de verkeerde kant van de beslissingslijn ligt, zal de *hinge loss* groter dan 1 zijn. We willen het model wel hard bestraffen, want zo'n fouten willen we miniem houden.

Om het SVM-model te trainen, moeten we nu dus met twee zaken rekening houden. Enerzijds willen we $\|\vec{w}\|$ minimaliseren om een zo groot mogelijke marge te bekomen anderzijds willen we de *hinge loss* of strafterm zo klein mogelijk houden. We voeren een kostfunctie J in en zoeken het minimum $\min_{w,b} J$ van deze functie. Het minimalisatieprobleem wordt dan gegeven door de formule

$$\min_{w,b} J = \min_{w,b} \left[\frac{1}{n} \sum_{i=1}^{n} \max\left[0, 1 - y_i(\vec{w} \cdot \vec{x}_i - b)\right] + \lambda \cdot ||\vec{w}||^2 \right]$$

.

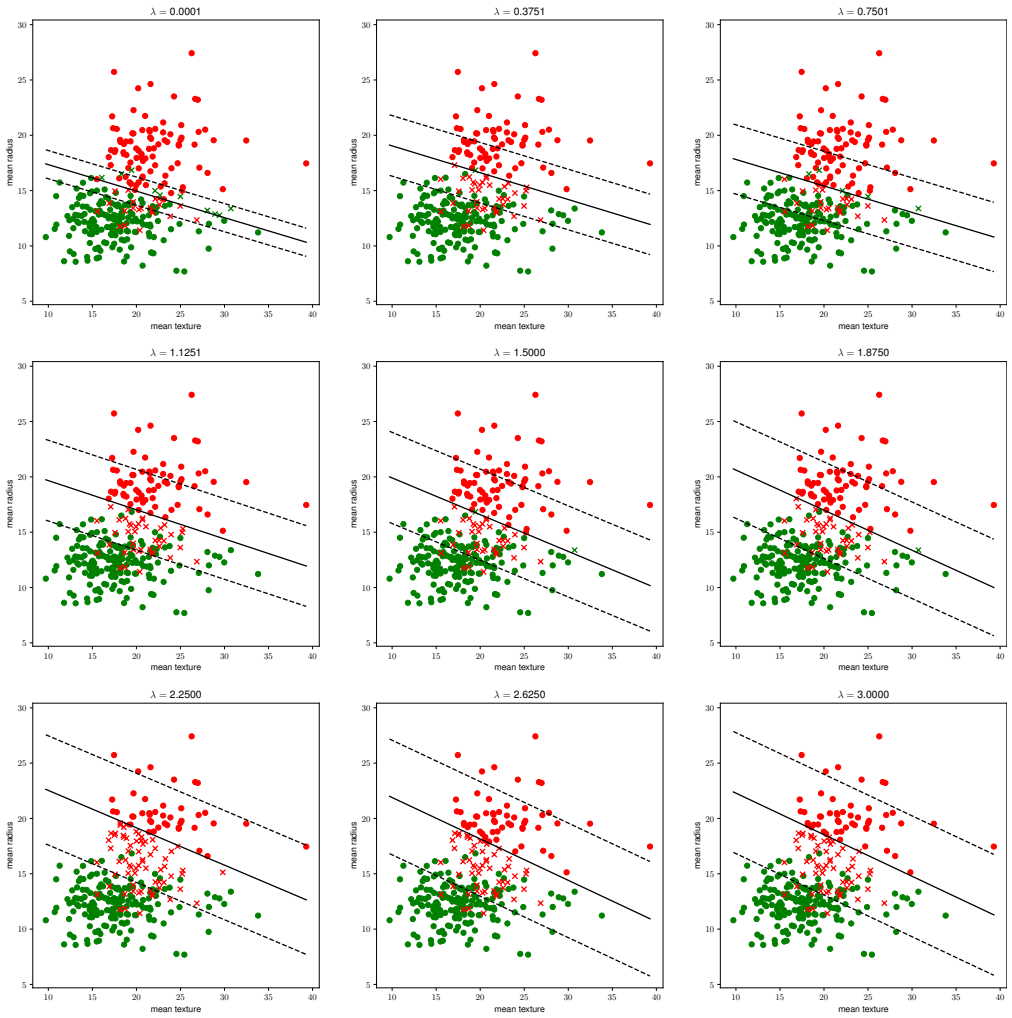
Hierbij is $\frac{1}{n}\sum_{i=1}^{n}\max\left[0, 1-y_i(\vec{w}\cdot\vec{x}_i-b)\right]$ de gemiddelde *hinge loss*, dus de gemiddelde foutterm die werd toegekend over alle n datapunten in de dataset. De tweede term $\lambda\cdot||\vec{w}||^2$ regelt het evenwicht tussen enerzijds het minimaliseren van de fouten op de trainingsdata en anderzijds het maximaliseren van de marge. λ is een metaparameter van het model en deze parameter vertelt aan het model hoeveel belang we hechten aan het minimaliseren van $\|\vec{w}\|$.

6.4.3 Regularisatieparameter

In het geval van SVM noemen we de metaparameter (zie deel 4.3.1) λ ook wel de *regularisatieparameter*. Deze parameter bepaalt de breedte van de marge en dus ook de variabiliteit van het model:

- Als λ klein - en eventueel zelfs 0 - is, komt het berekenen van een minimale kostfunctie J neer op het op nul zetten van zo veel mogelijke *hinge losses* van zo veel mogelijk punten. Dit resulteert in een kleine marge.
- Als λ groot is, ligt er meer nadruk op het minimaliseren van de grootte van $\|\vec{w}\|$ en dan spelen de *hinge losses* een minder grote rol. Hoe kleiner $\|\vec{w}\|$, hoe groter de marge, want de marge is $\frac{2}{\|\vec{w}\|}$ breed. We concluderen dat λ resulteert in een grotere marge, zoals te zien is in figuur 6.5.

Voor het vinden van de optimale waarde λ_{opt} voor de regularisatieparameter, verwijzen we graag terug naar deel 4.3.2, waar via *cross-validation* de beste waarde voor de metaparameter wordt bepaald. Een illustratie van de invloed van λ op de accuraatheid van het model is te zien in figuur 4.1.



Figuur 6.5: De invloed van de metaparameter λ op het SVM-model. Een grote λ resulteert in een grote marge, een kleine λ resulteert in een kleine marge.

Besluit

Bibliografie

[1] William Nick Street, William H. Wolberg, and Olvi L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. In *Electronic imaging*, 1993.

[2] Mangasarian Olvi Street Nick Wolberg, William and W. Street. Breast Cancer Wisconsin (Diagnostic). UCI Machine Learning Repository, 1995. DOI: https://doi.org/10.24432/C5DW2B.

[3] Stijn Rebry. Kennismaking met machinaal leren, academiejaar 2023-2024. KU Leuven Campus Kulak Kortrijk.

[4] Gabriele De Luca. Svm vs neural network, Nov 2022.

[5] Klevis Ramo. Spam emails with support vector machines, Dec 2017.

[6] Danilo Bzdok, Martin Krzywinski, and Naomi Altman. Machine learning: supervised methods. *Nature methods*, 15(1):5, 2018.

[7] Wikipedia contributors. Support vector machine — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Support_vector_machine&oldid=1183475870`, 2023. [Online; accessed 24-November-2023].

[8] Tarlan Ahadli. Linear svm classifier: Step-by-step theoretical explanation with python implementation. https://medium.com/, Gepubliceerd op 24 aug, 2021. Bezocht op 20 november 2023.

KU Leuven Kulak
Groep Wetenschap & Technologie
Etienne Sabbelaan 53
8500 Kortrijk