UHASSELT

KNOWLEDGE IN ACTION

Faculteit Bedrijfseconomische Wetenschappen

master handelsingenieur in de beleidsinformatica

Masterthesis

Een methode om Support Vector Machines te vertalen naar DMN Decision Tables

Michiel Bongaerts

Scriptie ingediend tot het behalen van de graad van master handelsingenieur in de beleidsinformatica

PROMOTOR :

dr. Frank VANHOENSHOVEN

UHASSELT

KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

2020
2021

Faculteit Bedrijfseconomische Wetenschappen

master handelsingenieur in de beleidsinformatica

Masterthesis

Een methode om Support Vector Machines te vertalen naar DMN Decision Tables

Michiel Bongaerts

Scriptie ingediend tot het behalen van de graad van master handelsingenieur in de beleidsinformatica

PROMOTOR :
dr. Frank VANHOENSHOVEN

Deze masterproef werd geschreven tijdens de COVID-19 crisis in 2020-2021. Deze wereldwijde gezondheidscrisis heeft mogelijk een impact gehad op het schrijf- en verwerkingsproces, de onderzoekshandelingen en de onderzoeksresultaten die aan de basis liggen van dit werkstuk

Universiteit Hasselt

Masterproef Handelsingenieur in de Beleidsinformatica

Een methode om Support Vector Machines te vertalen naar DMN Decision Tables

Auteur:
Michiel Bongaerts

Promotor:
dr. Frank Vanhoenshoven

4 Juni 2021

Samenvatting

Machine learning is een krachtige tool maar de modellen die gecreëerd worden, zijn vaak black boxes en missen transparantie. Deze paper stelt een methode voor die Binaire Lineaire Support Vector Machines omvormt naar DMN decision tables. Deze decision tables bieden een hoge graad van transparantie aan en kunnen gemakkelijk toegepast worden binnen gestandaardiseerde bedrijfsprocessen.

De methode voorgesteld in deze paper tracht hypercubes rond het Support Vector Machine hyperplane op te stellen. Deze hypercubes kunnen dan vertaald worden naar decision rules die verzameld worden in een DMN decision table. De methode wordt gevalideerd aan de hand van 4 datasets. De accuracy van de bevonden decision table wordt vergeleken met de accuracy van het onderliggende Support Vector Machine model en een Decision Tree. De analyse toont aan dat de accuracy van de Support Vector Machines en de Decision Trees geëvenaard en zelfs overtroffen kan worden door de bevonden decision tables. Verder wordt de methode ook vergeleken met een methode uit eerder onderzoek die Decision Trees omvormt naar DMN decision tables. Waar de methode uit vorig onderzoek geen vooruitgang boekt in transparantie, kan de methode uit deze paper een niet-transparante Support Vector Machine omvormen naar een transparante decision table. De decision tables bevonden door de methode in deze paper slagen er in de accuracy en het aantal decision rules van de andere methode te evenaren.

Deze paper stelt dus een methode voor om de beslissingslogica van een Support Vector Machine om te vormen naar meer transparante beslissingslogica zonder in te boeten op accuracy. Een tekortkoming van deze methode is echter dat het aantal decision rules exponentiëel stijgt naarmate de data hogere dimensies aannemen.

Inhoud

1 Introductie

In de bedrijfswereld is er vaak nood aan standaardisatie en automatisering van processen en beslissingen. Verder dienen deze beslissingen ook begrijpbaar en transparant te zijn. Hiervoor is de Business Process Model and Notation en Decision Model and Notation ontwikkeld door de Object Management Group (Biard, Le Mauff, Bigand, & Bourey, 2015). DMN helpt bedrijven hun processen en beslissingen efficiënter te controleren aan de hand van goed ontworpen beslissing- en informatiestructuren (Figl, Mendling, Tokdemir, & Vanthienen, 2018). DMN en de bijhorende decision tables, die transparante beslissingen beschrijven, zijn dus een logische keuze om zowel standaardisatie als transparantie aan te bieden. Daarnaast heeft de opkomst van machine learning gezorgd voor algoritmen die zeer performant zijn met betrekking tot beslissingen nemen of ondersteunen. De beslissingslogica van deze machine learning modellen past echter niet binnen DMN decision tables en is vaak niet transparant.

Dit leidt tot een kloof tussen de wereld van Machine learning en gestandaardiseerde transparante beslissingslogica. Eerder werk van Etinger heeft reeds een methode voorgesteld die het mogelijk maakt Decision Tree modellen om te vormen naar DMN decision tables (Etinger, Simić, & Buljubašić, 2019). Maar Decision Trees zijn slechts één van de vele machine learning algoritmen, voor andere algoritmen zijn nog geen methoden voorgesteld. Bepaalde machine learning algoritmen doen het beter op specifieke problemen dan andere (Caruana & Niculescu-Mizil, 2006). Dit zorgt ervoor dat enkel Decision Trees naar DMN omvormen niet voldoende is. Om de kracht van Machine Learning in DMN decision tables te introduceren, is er nood aan methodes die andere Machine Learning modellen doen passen binnen de DMN standaard.

Deze paper tracht de kloof tussen Machine learning en gestandaardiseerde beslissingslogica te verkleinen. Er wordt een methode voorgesteld die het mogelijk maakt transparante decision rules en DMN decision tables af te leiden uit lineaire binaire Support Vector Machines. Dit betekent dat de methode enkel werkt voor binaire classificatieproblemen waarvan de data lineair onderscheidbaar zijn. Waar de methode van Etinger eerder een vertaling van de beslissingslogica is, gaat deze paper nieuwe beslissingslogica en decision rules afleiden uit het hyperplane van een Support Vector Machine. Deze nieuwe beslissingslogica zal binnen een DMN decision table passen en dus een hogere graad van transparantie aanbieden dan de onderliggende Support Vector Machine. Verder zal deze decision table compleet zijn en een Unique Hit Policy toepassen om de complexiteit zo laag mogelijk te houden. De methode in deze paper tracht met de bevonden decision tables de accuracy van de Support Vector Machines te benaderen en tegelijkertijd een hogere graad van transparantie en interpreteerbaarheid aan te bieden.

De structuur van de paper is als volgt, in sectie 2 wordt de methodologie beschreven. Sectie 3 beschrijft de methode die naar voren wordt gebracht in deze paper. Sectie 4 en 5 voeren een analyse uit op de methode in respectievelijk twee en drie dimensies. En tenslotte bevat sectie 6 de conclusie.

2 Methodologie

In deze paper wordt een methode voorgesteld om een Support Vector Machine om te vormen naar een DMN decision table. Ten eerste werd de methode ontwikkeld voor tweedimensionele data en vervolgens werd deze uitgewerkt om ook toepasbaar te zijn op n-dimensionele data. De validatie van het algoritme bestaat uit twee delen, ten eerste wordt de accuracy van de bekomen decision table vergeleken met de onderliggende Support Vector Machine en een Decision Tree getraind op dezelfde data. De motivatie hierachter is te weten te komen of de decision table accuracy verliest ten opzichte van het Support Vector Machine model. En of het competitief blijft met een Decision Tree die ook bestaat uit decision rules. Het tweede deel van de validatie betreft een vergelijking van het aantal decision rules in de bevonden decision table met het aantal decision rules die afgeleid kunnen worden uit een Decision Tree. Het doel van deze analyse is te weten te komen of het aantal decision rules nodig om een bepaalde accuracy te behalen, vergelijkbaar is bij beide methoden. In de rest van deze sectie wordt verder ingegaan op de gebruikte tools en datasets om de methode op te stellen en de validatie uit te voeren.

Zowel het opstellen als het valideren van de methode gebeurde in Python. De methode is gebaseerd op Support Vector Machine modellen zoals geïmplementeerd in het package Scikit-learn (Pedregosa et al., 2011). De Support Vector Machine modellen worden omgevormd naar een set van decision rules. Deze decision rules worden via het python package xml.etree.ElementTree (*The ElementTree XML API documentation*, n.d.) omgevormd tot een DMN decision table. De bekomen decision tables werden gevisualiseerd via de Camunda Modeler.

De validatie van de methode gebeurde aan de hand van 4 datasets verkregen via de UCI machine learning repository (Dua & Graff, 2017). De datasets zijn Iris, Breast Cancer Wisconsin, Wine data set en Bank Note Authentication dataset. De eerste drie datasets zijn beschikbaar in python via scikit-learn, enkel de Banknote Authentication dataset is manueel ingeladen. Al de datasets werden via min-max normalization genormaliseerd voor de training van de Support Vector Machine en Decision Tree modellen. Aangezien de validatie van de methode vooral focust op 2 dimensionale inputdata werden uit de eerste drie datasets slechts twee parameters gekozen. Indien een dataset meerdere decision classes had, werden er twee uitgekozen aangezien de methode enkel werkt voor binaire classificatie. De keuze van de parameters is gebaseerd op de interpreteerbaarheid van de parameter. Aangezien de parameters van de Iris dataset alle even interpreteerbaar zijn, werden hier de eerste twee parameters gekozen. De keuze van de decision classes werd gemaakt op basis van positie, de eerste twee classes die voorkwamen in de dataset werden behouden, de andere werden verwijderd indien nodig.

Uit de Iris dataset werden de sepal width en length als input parameters gebruikt en werden de data gefilterd zodat enkel de decision class setosa en versicolor overbleef. Voor Breast Cancer Wisconsin werden enkel mean texture en mean area behouden als input parameters. Voor de Wine dataset werden enkel color intensity en alcohol als input parameters en decision class 0 en 1 behouden. De vierde dataset namelijk de Bank Note Authentication dataset werd gebruikt om de methode in drie dimensies te testen. De motivatie hierachter is om te valideren of de

methode ook werkt in hogere dimensies. Hier werden de variance, skewness en curtosis of Wavelet Transformed image gebruikt als drie input parameters. Deze parameters waren opnieuw de eerste drie parameters in de dataset.

Vervolgens werden de vier datasets gebruikt om een Support Vector Machine en een decision tree, via sklearn.tree, te trainen. Dit gebeurde via een 75/25 train/test split. Voor de Support Vector Machines werden de hyperparameters op de standaardwaardes behouden en de Decision Trees werden ingesteld met een maxDepth van 3. Vervolgens werden de accuracies van deze modellen vergeleken met de accuracy van de decision tables bekomen via de methode voorgesteld in deze paper. Deze vergelijking toont aan of de bevonden decision tables de accuracies van de machine learning modellen kunnen benaderen. Tenslotte werd er ook nog een analyse uitgevoerd die het aantal decision rules in de decision table bekijkt. Dit aantal wordt vergeleken met het aantal decision rules dat de methode van Etinger uit Decision Tree modellen verkrijgt (Etinger et al., 2019). De Decision Tree modellen zijn net zoals de Support Vector Machine modellen getraind op de vier eerder beschreven gefilterde datasets. Het aantal decision rules dat de methode van Etinger bekomt, is zeer afhankelijk van de depth van de decision tree. Al de decision trees worden met een maxdepth van 3 getraind. Maar aangezien deze analyse vooral focust op de relatie tussen het aantal decision rules en de accuracy is dit geen groot probleem.

3 De Methode

3.1 Introductie

De visuele representatie van decision rules beschreven door Calvanese, ligt aan de basis van de methode voorgesteld in deze paper (Calvanese et al., 2018). Een decision rule beschrijft voor elke parameter een interval waarbinnen deze decision rule getriggerd wordt. Als deze intervallen geplot worden op een grafiek beschrijft dit in 2 dimensies een rechthoek en in hogere dimensies een hypercube. Op deze manier kunnen decision rules gevisualiseerd worden.

Een binaire Support Vector Machine zal een hyperplane opstellen dat de feature space opdeelt in twee zones die elk een klasse voorstellen. De methode beschreven in deze paper tracht hypercubes rond het hyperplane op te stellen die uiteindelijk decision rules in een decision table zullen voorstellen. De vertaling van de hypercubes naar decision rules is vrij triviaal, de grenzen van de hypercubes beschrijven de grenzen van de intervallen van de decision rules.

De bevonden hypercubes dienen de hele feature space te beschrijven zodat de uiteindelijke decision table compleet is. Dit betekent dat elke mogelijke inputconfiguratie een decision rule zal triggeren (Calvanese et al., 2018). Verder is het ook wenselijk dat de hypercubes of decision rules niet overlappen zodat we in de decision table een Unique Hit Policy kunnen toepassen. Onderzoek geeft aan dat deze policy de laagste decision table complexiteit teweegbrengt (Hasić & Vanthienen, 2019). Het uiteindelijke doel is dus vanuit een Support Vector Machine model een complete decision table met een Unique Hit policy te bekomen, dit betekent dat elke input één en slechts één decision rule uit de decision table zal triggeren.

De methode beschreven in deze paper is enkel gepast voor binaire lineaire Support Vector

Machines. Dit wilt zeggen dat deze methode enkel werkt voor binaire classificatieproblemen die lineair onderscheidbaar zijn.

3.2 Visuele voorstelling

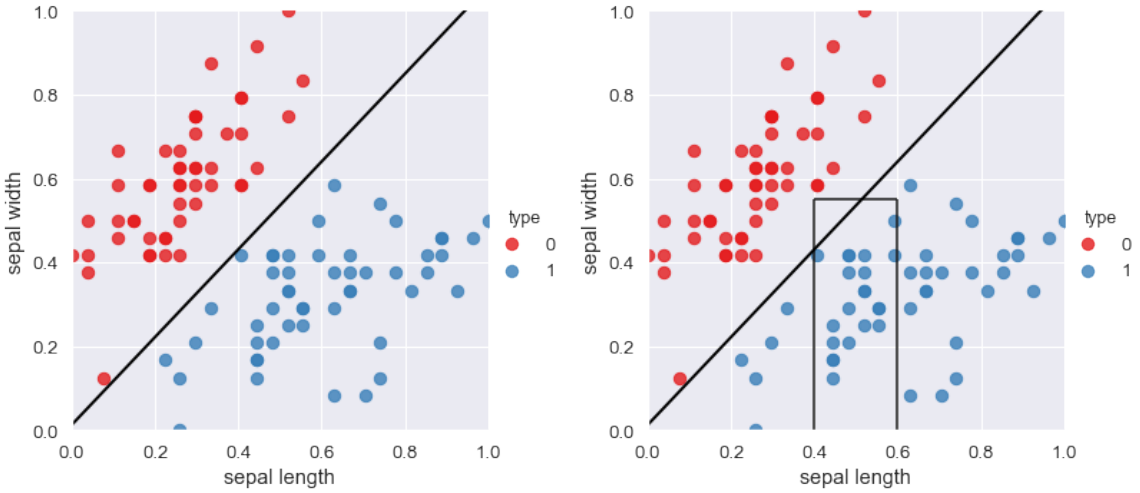
Om het idee achter de methode beter te schetsen wordt in figure 1a tot 1d een visuele voorstelling gepresenteerd van het concept[1]. Op figure 1a wordt de genormaliseerde Iris data en het separating hyperplane dat een Support Vector Machine heeft getraind, weergegeven. Verder zijn de punten ook door middel van kleur opgedeeld om aan te duiden tot welke klasse ze behoren. Volgens het Support Vector Machine model zal elk punt dat onder het hyperplane valt tot klasse 1 behoren en elk erboven tot klasse 0.

Nu dienen de vlakken onder en boven het hyperplane beschreven te worden aan de hand van de input variabelen. Voor deze beschrijving worden zoals eerder aangehaald hypercubes gebruikt. Aangezien dit voorbeeld 2-dimensionaal is, zijn de hypercubes rechthoeken. Stel dat we het vlak onder het hyperplane willen beschrijven met rechthoeken. Een eerste rechthoek wordt getekend op figure 1b. Deze rechthoek zegt dat indien de sepal length tussen 0.4 en 0.6 ligt en de sepal width kleiner is dan 0.55 dit punt dan behoort tot klasse 1. De bovengrens van de decision rule is gecentreerd op het separating hyperplane.

Deze rechthoek doet grotendeels dezelfde voorspellling als het hyperplane, enkel de driehoek die boven het separating hyperplane valt, wordt anders voorspeld. Een punt in de driehoek zou door het SVM model als behorende tot klasse 0 geclassificeerd worden terwijl de rechthoek dit punt tot klasse 1 classificeert. Bij deze methode is het onvermijdelijk dat bepaalde gebieden anders voorspeld worden. De breedte van de decision rule heeft een grote impact op de oppervlakte van het gebied dat anders wordt voorspeld, dit wordt in de volgende sectie besproken.

Indien we dit patroon verder zetten bekomen we op figure 1c een set van decision rules die de voorspelling van klasse 1 beschrijft. Hier zal de meest linkse decision rule geen linkergrens hebben, en de meest rechtse decision rule geen rechtergrens. Op die manier wordt zeker de hele feature space beschreven. De rechthoeken die klasse 0 beschrijven worden toegevoegd op figure 1d. Zo bekomen we een set van 10 rechthoeken die de hele feature space beschrijven en niet overlappen. Deze rechthoeken kunnen nu getransformeerd worden naar decision rules. Dit kan door de grenzen van de rechthoeken te gebruiken als grenzen van de intervallen van de decision rules.

[1]Voor deze voorstelling is Scikit-learn, Matplotlib, Seaborn en data uit de Iris dataset gebruikt.(Pedregosa et al., 2011)(Hunter, 2007)(Waskom, 2021)(Dua & Graff, 2017)



(a) Separating hyperplane

(b) Eerste decision rule

(c) Decision rules klasse 1

(d) Alle decision rules

Figure 1: Visuele voorstelling methode

3.3 Keuzes die de decision rules beïnvloeden

Voor het plaatsen van de rechthoeken of decision rules zijn er drie keuzes die gemaakt dienen te worden. Ten eerste waar de eerste decision rule wordt geplaatst. Als bijvoorbeeld op figure 1d de decision rules verschoven zouden worden langs het hyperplane dan zouden de zones die de decision rules anders voorspellen dan het hyperplane ook verschuiven. Dit zorgt voor een set van decision rules die andere voorspellingen zouden doen dan de eerste set. Het voorspellen welke beginpositie de beste resultaten zal opleveren is zeer moeilijk. In deze paper is er gekozen om de eerste decision rule op te stellen rond de minimum x-waarde van de training set. De eerste decision rule zal dus gecentreerd zijn op de minimum x-waarde.

Vervolgens is een zeer belangrijke keuze de breedte van de decision rules. Dit zal niet enkel bepalen hoe groot de zones zijn die anders voorspeld worden maar ook hoeveel decision rules er nodig zijn om de feature space te beschrijven. Een kleine breedte zal zorgen voor veel decision rules maar een kleinere oppervlakte die anders voorspeld wordt. Een grote breedte zal zorgen voor weinig decision rules maar een grotere oppervlakte die anders voorspeld wordt. In sectie 3.4 wordt een methode getoond die een breedte berekent gebaseerd op de marge van het Support Vector Machine model. De impact van de breedte wordt in sectie 4.1 en 5.1 nog besproken.

Tenslotte rest er nog de mogelijkheid om de decision rules vanuit een andere as op te stellen. In het voorbeeld van afbeelding 1a tot 1d is er gekozen om de decision rules te tekenen vanuit de x-as. Het was ook mogelijk om de decision rules vanuit de y-as op te stellen. Dan hadden de decision rules horizontaal gelegen in plaats van verticaal. Ook deze keuze beïvloedt de positie van de zones die de decision rules anders voorspellen dan het hyperplane. In deze paper is er steeds gekozen om de decision rules op te stellen vanuit de eerste parameter.

De keuze van de positie van de decision rules en vanuit welke as de decision rules worden opgesteld, worden in deze paper niet verder besproken. Sectie 4.1 en 5.1 gaan wel verder in op de impact van de decision rule breedte op de accuracy en het aantal decision rules.

3.4 Algoritme voor 2 dimensionele input data

In dit onderdeel wordt het algoritme besproken dat de decision rules opstelt vanuit het Support Vector Machine model. Dit algoritme beschrijft enkel 2 dimensionale input data, het algoritme voor hogere dimensies wordt in de volgende sectie besproken. Het opstellen van dit algoritme gebeurde in Python en het Support Vector Machine model is getraind aan de hand van Sklearn.SVM. Elke input die gebruikt wordt in het algoritme is af te leiden uit het Scikit-learn Support Vector Machine model.

Ten eerste heeft het algoritme de vergelijking van het hyperplane nodig. Dit maakt het mogelijk om voor een bepaalde x-waarde de y-waarde van het hyperplane te berekenen. Vervolgens is de *marge* van het Support Vector Machine model ook nodig om te gebruiken als basis voor de breedte van de decision rules. De marge is de afstand tussen de datapunten die gemaximaliseerd wordt om het separating hyperplane op te stellen (Kecman, 2005).

Het algoritme is te volgen in algorithm 1: Extractie decision rules. Er zijn enkele regels die extra uitleg kunnen gebruiken. Regel 3 roept de functie aan die de breedte van de decision rules

bepaalt. De breedte wordt berekend aan de hand van de richtingscoëfficiënt van het hyperplane en de marge. De breedte wordt gevisualiseerd op figure 2. Hier beeldt de volle zwarte rechte het SVM hyperplane uit en de 2 stippellijnen de lijnen door de support vectors. Deze gestippelde lijnen liggen beide een halve marge verwijderd van het hyperplane. Als breedte van de decision rules is de x-afstand tussen de twee stippellijnen gekozen, de vette horizontale lijn op figure 2. De afstand wordt berekend aan de hand van de basisverhoudingen uit de goniometrie. Op deze manier wordt ervoor gezorgd dat de door de decision rules anders voorspelde gebieden enkel tussen de twee stippellijnen en dus de support vectors vallen. Zo worden de gebieden buiten de stippellijnen volledig hetzelfde voorspeld door het Support Vector Machine model en de decision rules. Tenslotte dient de structuur van de DecisionOnder en DecisionBoven array die op lijn 6 en 7 wordt aangegeven ook toegelicht te worden. Elk object binnen de array beschrijft een rechthoek en bevat 4 waardes die de grenzen van de rechthoek beschrijven. De eerste twee waardes beschrijven de minimum en maximum x-waardes van de rechthoek. De laatste twee waardes zijn de minimum en maximum y-waardes van de rechthoek, false duidt hier aan dat er geen boven of ondergrens is.

Algorithm 1: Extractie decision rules

Input: SVM hyperplane vergelijking
Richtingscoëffiënt hyperplane
Marge SVM model

Output: Array DecisionOnder
Array DecisionBoven `// Decision rules onder en boven hyperplane`

Data: 2 dim SVM training set z

```
1  minX = minimum x waarde in z
2  maxX = maximum x waarde in z
3  breedte = Breedte(rico, marge)
4  while curX < maxX+breedte/2 do
5      curY = Y-waarde van curX op SVM hyperplane
6      DecisionOnder.append([curX-breedte/2, curX+breedte/2, curY, False])
7      DecisionBoven.append([curX-breedte/2, curX+breedte/2, False, curY])
8      curX+= breedte
9  Function Breedte(rico, marge):
10     overstaande=marge
11     hoek= arctan(rico)
12     schuine= abs(overstaande/ sin(hoek))
13     return schuine
```

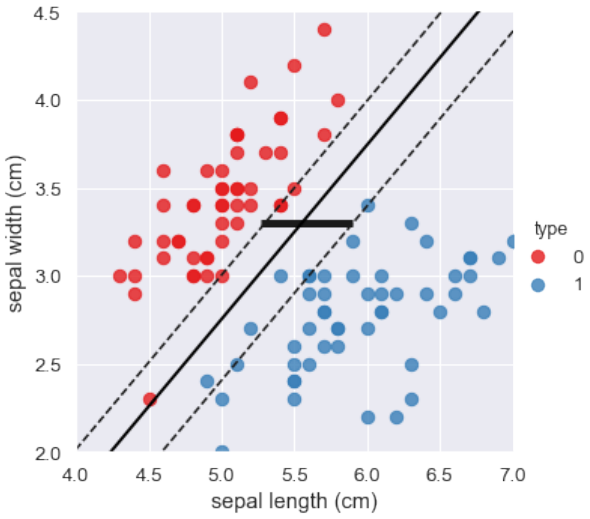


Figure 2: Visualisatie van de breedte van een decision rule

3.5 Algoritme in hogere dimensies

Dit algoritme is ook uitbreidbaar naar hogere dimensies mits enkele aanpassingen en is weergegeven in Algorithm 2. Indien er gebruik gemaakt wordt van n-dimensionele data dienen bepaalde stappen binnen het algoritme n-1 keren uitgevoerd te worden. Zo zal er voor n-1 dimensies de minimum waarde, de maximum waarde en de breedte berekend worden. Deze berekening wordt weergegeven op lijn 1 tot en met 4. Vervolgens dienen er ook n-1 geneste while loops aanwezig te zijn, waarbij een loop overeenkomt met een dimensie. Tenslotte worden de arrays die aan DecisionOnder en DecisionBoven worden toegevoegd ook groter, deze bevatten nu $n * 2$ elementen, een boven en ondergrens voor elke dimensie.

Algorithm 2: Extractie decision rules in n dimensies

Input: SVM hyperplane vergelijking
Richtingscoëffiënt hyperplane per dimensie
Marge SVM model

Output: Array DecisionOnder
Array DecisionBoven `// decision rules onder en boven hyperplane`

Data: n dim SVM training set z

1 **for** $i \leftarrow 1$ **to** $n-1$ **do**
2 min[i] = minimum waarde in dimensie i
3 max[i] = maximum waarde in dimensie i
4 breedte[i] = Breedte($rico_i$, marge)
5 **while** *min[1]< max[1]+*$breedte_1/2$ **do**
  ⋮
6 **while** *min[n-1] < max[n-1]+*$breedte_{n-1}/2$ **do**
7 curN = waarde in dim n op SVM hyperplane behorende aan min[1] tot min[n-1]
8 DecisionOnder.append([min[1]-$breedte_1/2$, min[1]+$breedte_1/2$, ...,
  min[n-1]−$breedte_{n-1}/2$,min[n-1]+$breedte_{n-1}/2, curN, False$])
9 DecisionBoven.append([min[1]-$breedte_1/2$, min[1]+$breedte_1/2$, ...,
  min[n-1]−$breedte_{n-1}/2$,min[n-1]+$breedte_{n-1}/2, False, curN$])
10 min[n-1]+= $breedte_{n-1}$
11 min[n-1]= minimum waarde in dimensie $n-1$
  ⋮
12 min[1]+= $breedte_1$
13 **Function** `Breedte(`*rico, marge*`):`
14 overstaande=marge
15 hoek= arctan(*rico*)
16 schuine= abs(overstaande/ sin(*hoek*))
17 **return** schuine

3.6 Vertaling gevonden decision rules naar DMN Decision Table

Voor het opstellen van een DMN decision table werd gebruik gemaakt van het Python package xml.etree.ElementTree (*The ElementTree XML API documentation*, n.d.). Dit package staat toe op een gebruiksvriendelijke manier XML code te genereren. Een high-level algoritme voor de creatie van de decision table wordt weergegeven in Algorithm 3. Dit algoritme neemt als input de DecisionOnder en DecisionBoven array uit Algorithm 1 en 2 en het aantal dimensies van de inputdata. Ten eerste wordt er een XML decision table gecreëerd aan de hand van verschillende xml.etree.ElementTree functies. Vervolgens wordt op regel 3 per dimensie van de

data een input aan de decision table toegevoegd. Regel 4 voegt ook een output toe. Vervolgens wordt er geïtereerd over de DecisionOnder array en wordt per element een nieuwe decision rule aangemaakt. Vervolgens wordt op regel 10 voor elk rule over al de inputs geïtereerd. Hier wordt dan uit het overeenkomstig DecisionOnder element de grenzen voor dimensie k gehaald en in de rule geplaatst. Tenslotte wordt de output van de rule op 1 gezet zodat de decision rule gelinkt is aan de juiste decision class. De for loop op regel 8 is identiek aan de voorgaande, enkel wordt er nu geïtereerd over de DecisionBoven array en wordt de output van de regel op 0 gezet.

Voorbeelden van bekomen DMN decision tables zijn te vinden in figure 4 tot en met 11 in de Appendix sectie.

Algorithm 3: Opstellen DMN decision table vanuit hypercube arrays

Input: DecisionOnder
DecisionBoven
aantal dimensies data n

Output: DMN decision table

```
1  DecisionTable = new decisiontable                    // met hitpolicy UNIQUE
2  for i ← 0 to n do
3      DecisionTable.input[n]= new input
4  DecisionTable.output = new output
5  for j ← 0 to length(DecisionOnder) do
6      DecisionTable.rule[j]= new rule
7      for k ← 0 to n do
8          DecisionTable.rule[j].input[k]= grenzen voor dim k uit DecisionOnder[j]
9      DecisionTable.rule[j].output = 1
10 for l ← 0 to length(DecisionBoven) do
11     DecisionTable.rule[l]= new rule
12     for m ← 0 to n do
13         DecisionTable.rule[l].input[m]= grenzen voor dim m uit DecisionBoven[l]
14     DecisionTable.rule[l].output = 0
15 return DecisionTable
```

4 Validatie van de methode in 2 dimensies

Deze sectie beschrijft een analyse van de methode. De focus van de analyse ligt op de accuracy van de bevonden decision table en het aantal decision rules dat deze table bevat.

4.1 Accuracy

Zoals reeds in de methodologie beschreven, wordt in deze sectie de accuracy van de bekomen DMN decision table vergeleken met de accuracy van de Support Vector Machine waarop de

Accuracy	Iris	Wine	Breast cancer Wisconsin
Decision Tree	0.92	0.97	0.88
Support Vector Machine	1	0.97	0.87
Decision Table	1.00 (0.10,22)	0.94 (0.10,22)	0.87 (0.10,22)
	0.96 (0.30,8)	0.94 (0.30,8)	0.87 (0.30,8)
	0.88 (0.58,6)	**0.94 (0.58,6)**	0.89 (0.50,6)
	0.84 (0.80,4)	0.91 (0.80,4)	0.92 (0.80,4)
	0.80 (1.00,4)	0.97 (1.00,4)	**0.81 (1.08,4)**
	0.52 (1.50,4)	0.67 (1.50,4)	0.80 (1.50,4)
	0.40 (2.00,2)	0.42 (2.00,2)	0.80 (2.00,2)

Table 1: Accuracy analyse 2d

decision table gebaseerd is en een Decision Tree. Voor deze analyse zijn drie tweedimensionale datasets gebruikt.

De accuracies van de modellen op de testsets zijn terug te vinden in table 1: Accuracy analyse 2d. Hier stellen de kolommen de gebruikte datasets voor en de rijen het gebruikte model. Naast de decision table accuracy staan twee waardes tussen haakjes. Deze twee waardes stellen respectievelijk de breedte van de decision rules en het aantal decision rules voor. De decision table voor de iris dataset behaalt dus een accuracy van 1 wanneer de breedte van de decision rules 0.1 bedraagt en de decision table bestaat uit 22 decision rules.

Zoals eerder aangegeven in Algorithm 1 berekent de methode de breedte van de decision rules zelf aan de hand van de marge van de Support Vector Machine. Op deze manier werd getracht de accuracy van het Support Vector Machine Model te benaderen en tegelijkertijd geen groot aantal decision rules te hebben. Het is echter interessant om te analyseren welke impact de breedte van de decision rules heeft op de accurary van de decision table en op het aantal decision rules. Daarom is er gekozen om de breedtes 0.1, 0.3, 0.5, 0.8, 1, 1.5 en 2 te testen. De motivatie hierachter is dat aangezien de data genormaliseerd zijn, elke waarde boven 2 dezelfde accuracy als 2 zal geven. Dit komt doordat bij een breedte van 2 de eerste 2 decision rules al de hele breedte van 0 tot 1 beschrijven. Verder zijn er meer waardes onder 1 gekozen omdat deze interessantere resultaten zullen geven dan elke waarde tussen 1 en 2 die altijd 4 decision rules zal geven. De door het algoritme berekende breedtes zijn vet gedrukt. Aangezien deze vaak dichtbij de vooropgestelde breedtes vallen, vervangen de berekende breedtes die waardes. Zo vervangt bij de iris dataset de breedte van 0.58 de breedte van 0.5. De datasets en de bijhorende Support Vector Machine hyperplanes zijn gevisualiseerd in figure 3a tot en met 3c.

De bekomen decision table volgt bij de Iris dataset een verwacht patroon. Naarmate de breedte van de decision rules kleiner wordt en dus het aantal decision rules stijgt, benadert de decision table de accuracy van de Support Vector Machine. Indien de breedte van de decision rules naar

nul zal neigen, zal de oppervlakte van de zones die anders voorspeld worden ook naar nul neigen en zal dezelfde accuracy als de Support Vector machine bereikt worden. Uit de iris dataset is wel op te merken dat met een breedte van 0.30 en 8 decision rules, de accuracy van het decision tree model reeds overtroffen wordt. Volgens de methode van Etinger bevat de Decision Tree echter slechts 5 decision rules. De hogere accuracy wordt dus enkel bereikt met een hoger aantal decision rules.

In de wine dataset is dit patroon minder aanwezig. De decision table met een breedte van 1 en slechts 4 decision rules heeft de hoogste accuracy van al de geanalyseerde breedtes. Dit kan er op wijzen dat deze decision rules een cluster van punten opvangen die andere decision rules missen. Op figure 3b bevindt zich een kleine zone rode punten onder het hyperplane, deze cluster wordt door de decision table met een breedte van 1 als behorende tot klasse 0 geclassificeerd.

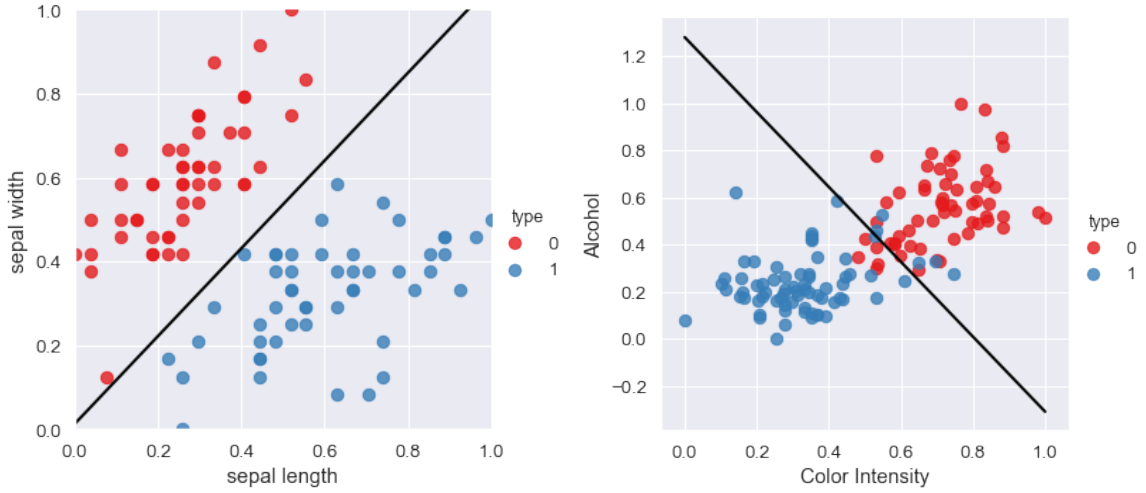
Tenslotte is er bij de Breast Cancer Winsconsin wel het patroon aanwezig dat de accuracy van de decision table de accuracy van de Support Vector Machine benadert naarmate de breedte kleiner wordt. Maar de decision tables die het beste presteren hebben een breedte van 0.5 en 0.8 met respectievelijk een accuracy van 0.89 en 0.92. Deze decision tables hebben dus een hogere accuracy dan het Support Vector Machine model waaruit ze afgeleid zijn. Dat wijst erop dat deze decision tables een patroon in de data beschrijven dat het Support Vector Machine niet gecapteerd heeft. Op figure 3c zijn de Breast Cancer Wisconsin data gevisualiseerd, hier bevindt zich duidelijk een groep van rode punten onder het hyperplane die door het Support Vector Machine model fout geclassificeerd worden. Op figure 3d staan de decision rules uit de decision table met een accuracy van 0.92, gevisualiseerd. Hier is duidelijk te zien dat de decision rule rechtsboven op de plot wel de cluster rode punten onder het hyperplane juist classificeert. Ook de zone boven het hyperplane in de decision rule linksonder bevat nog een blauw punt en slechts één rood. De hogere accuracy geeft dus aan dat de trapstructuur van de decision rules patronen in de data kan beschrijven dat een lineair Support Vector Machine model niet kan. Echter is het vinden van deze patronen zeer afhankelijk van de breedte van de decision rules. En het is niet mogelijk op voorhand te weten welke breedte de juiste patronen kan opvangen.

Uit deze analyse kan er besloten worden dat het mogelijk is voor de bevonden decision tables om de accuracy van het Support Vector Machine model en de Decision Tree te benaderen. In sommige gevallen kan de accuracy zelfs overtroffen worden. Iris toont een patroon dat de accuracy stijgt naarmate de breedte kleiner wordt en er meer decision rules zijn. Wine toont ook een stijgend patroon met een uitzondering bij een breedte van 1. Bij Breast Cancer Wisconsin is dit patroon minder aanwezig, een kleinere breedte zorgt voor een betere benadering van de accuracy van het Support Vector Machine model. Maar grotere breedtes zorgen voor accuracies die hoger zijn dan de accuracy van het Support Vector Machine model zelf.

Iris toont aan dat de breedte van de decision rules eerder een design keuze kan zijn met een trade-off tussen accuracy en simpliciteit van de decision table. Bijvoorbeeld in de iris dataset wordt een accuracy van 0.96 reeds met 8 decision rules bereikt. Een accuracy van 1 wordt bereikt met 22 decision rules, dit zijn 14 extra rules voor slechts een verhoging in de accuracy van 0.04. Indien er waarde wordt gehecht aan de simpliciteit van de decision table kan er best voor de 8 decision rules gekozen worden, maar indien de accuracy heel belangrijk is, kan er gekozen worden

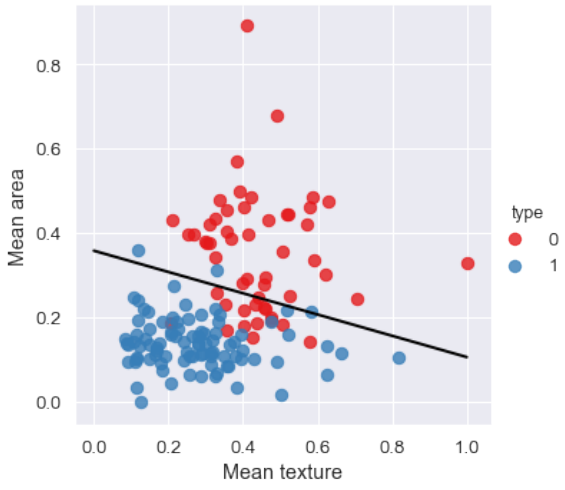
voor een decision table met 22 decision rules.

Wine en Breast Cancer Wisconsin tonen echter aan dat de breedte die zorgt voor de beste accuracy niet direct de kleinste is. Maar dat het vinden van de breedte met de beste accuracy eerder een trial and error proces is.

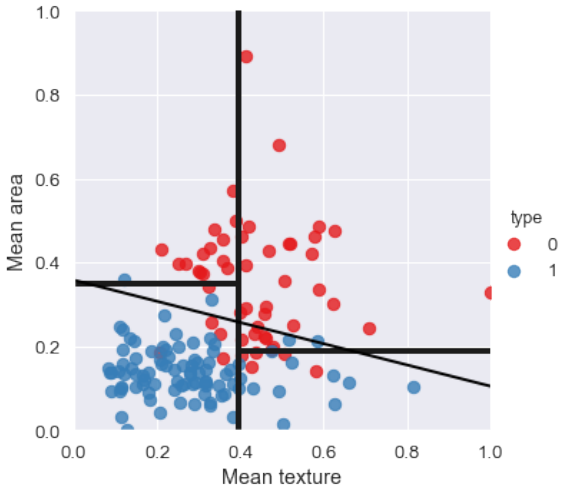


(a) Iris data

(b) Wine data



(c) Beast Cancer Wisconsin data



(d) Breast Cancer Wisconsin met decision rules

Figure 3: Visualisatie datasets

4.2 Aantal decision rules

Verder is het waardevol om te analyseren hoe dit algoritme presteert in vergelijking met andere algoritmes die hetzelfde trachten te bereiken. Voor deze analyse wordt de methode van Etinger gebruikt die decision tables afleidt uit Decision Trees ter vergelijking. Deze methode creëert een decision table waarbij elke endnote of elk pad door de boom overeenkomt met een decision rule (Etinger et al., 2019). Deze bevonden decision table zal exact dezelfde accuracy hebben als de onderliggende Decision Tree. Dat is bij de methode in deze paper niet het geval. De reden hiervoor is dat de methode van Etinger de reeds transparante beslissingslogica van een Decision Tree slechts hoeft te vertalen. De methode in deze paper zal uit een niet-transparante Support Vector Machine transparante beslissingslogica afleiden.

Indien de methode van Etinger toegepast wordt op de decision trees getraind op de Iris, Wine en Breast Cancer Wisconsin datasets, bekomen we decision tables met een accuracy en aantal decision rules beschreven in table 2.

Deze koppels van accuracy en aantal decision rules kunnen nu vergeleken worden met de koppels gevonden door de methode voorgesteld in deze paper. Uit table 1 is af te lezen dat voor de Iris dataset de methode uit deze paper de accuracy niet kan evenaren met hetzelfde aantal decision rules, namelijk 5. Met 1 decision rule meer behaalt de decision table verkregen door de methode in deze paper een accuracy van 0.88, dit is nog steeds lager dan de accuracy bereikt door de methode van Etinger. Echter met 8 decision rules, dus 3 meer dan de decision table van Etinger, wordt wel een hogere accuracy bereikt, namelijk 0.96.

Voor de wine dataset bereikt de methode uit deze paper dezelfde accuracy met 1 decision rule minder, namelijk een accuracy van 0.97 met 4 decision rules. Voor de Breast Cancer Wisconsin dataset wordt zelfs een hogere accuracy bereikt met de helft van de decision rules, namelijk een accuracy van 0.92 met 4 decision rules.

Uit deze analyse kan afgeleid worden dat de methode in deze paper vergelijkbaar is aan een reeds eerder voorgestelde methode. Echter biedt Etinger slechts een vertaling van een reeds transparante Decision Tree naar een decision table aan. De methode voorgesteld in deze paper gaat een stap verder en vormt een niet-transparante Support Vector Machine om naar een transparante decision table.

Een tekortkoming van deze analyse is wel dat er meerdere decision tables berekend zijn volgens de methode uit deze paper en de methode van Etinger slechts op 1 Decision Tree model gebaseerd is. Moesten er verschillende Decision Trees getraind zijn en de decision table gebaseerd op de beste tree gekozen zijn, kon de methode van Etinger misschien betere resultaten behalen.

Dataset	Accuracy	Aantal decision rules
Iris	0.92	5
Wine	0.97	5
Breast Cancer Wisconsin	0.88	8

Table 2: Aantal decision rules methode Etinger

4.3 Decision table

Ter afsluiting van deze sectie worden er enkele voorbeelden van bevonden decision tables gegeven. De XML bestanden voor deze decision tables zijn opgesteld met gebruik van xml.etree.ElementTree en zijn gevisualiseerd in Camunda Modeler. Voor elk van de 3 datasets worden 2 decision tables getoond. De eerste table van een dataset bevat de decision rules die verkregen worden aan de hand van de breedte berekend door het algoritme. De tweede decision table is de table met de hoogste accuracy. De decision tables zijn terug te vinden in figure 4 tot en met 9 in de Appendix sectie.

5 Validatie van de methode in 3 dimensies

Deze sectie beschrijft een analyse van de methode in drie dimensies: ten eerste om aan te tonen dat de methode effectief uitbreidbaar is naar hogere dimensies en ten tweede om mogelijke problemen te identificeren indien er uitgebreid wordt naar hogere dimensies. De opbouw van deze sectie is gelijkaardig aan de voorgaande.

5.1 Accuracy

Voor deze analyse is enkel de Bank Note Authentication dataset gebruikt en hierop zijn een Support Vector Machine en een Decision Tree getraind. Verder is ook Algorithm 2 gebruikt om decision rules uit de Support Vector Machine af te leiden.

De bevonden accuracy van het Decision tree model, het Support Vector Machine model en de decision table is af te lezen in table 3. Deze table heeft dezelfde structuur als table 1. Enkel staan er nu 3 getallen tussen haakjes bij de accuracy van de decision table. Aangezien er nu in 3 dimensies wordt gewerkt, dienen er nu 2 breedtes berekend te worden die de breedte (en diepte) van de decision rules bepalen. Het eerste getal is de breedte voor de genormaliseerde variance parameter, het tweede de breedte van de genormaliseerde skewness parameter en het derde is opnieuw het aantal decision rules.

Verder is de analyse van de bevonden decision table ook opgesplitst in wanneer de twee breedtes hetzelfde zijn en wanneer deze verschillen. De vetgedrukte waardes beschrijven opnieuw de breedtes van de decision rules die gevonden zijn door het algoritme. Het is opmerkelijk dat deze twee waardes zeer dicht bij elkaar liggen. Dit is echter te verklaren door het feit dat de

Accuracy	Bank Note Authentication
Decision Tree	0.93
Support Vector Machine	0.98
Decision Table	0.98 (0.10,0.10,242)
Gelijke Breedtes	**0.94 (0.25,0.26,50)**
	0.93 (0.30,0.30,32)
	0.87 (0.50,0.50,18)
	0.73 (0.80,0.80,8)
	0.85 (1.00,1.00,8)
	0.78 (1.50,1.50,8)
	0.40 (2.00,2.00,2)
Decision Table	0.40 (0.25,2.00,10)
Verschillende Breedtes	0.40 (2.00,0.26,10)
	0.65 (0.25,1.00,20)
	0.84 (1.00,0.26,20)
	0.67 (0.50,1.50,12)
	0.63 (1.50,0.50,12)

Table 3: Accuracy analyse 3d

absolute waarde van de richtingscoëfficiënt in beide diminensies gelijkend is. Dit betekent dat de hellingsgraad van het Support Vector Machine hyperplane in beide dimensies ongeveer hetzelfde is, bij andere datasets kunnen deze verschillen. Als extra waardes voor de breedtes zijn dezelfde waardes als in table 1 gekozen met dezelfde motivatie. Voor de verschillende breedtes is er gekozen om de extremen met elkaar te vergelijken.

De rij met gelijke breedtes toont aan dat de accuracy van de Support Vector Machine geëvenaard wordt indien de 2 breedtes 0.10 bedragen, dit brengt wel 242 decision rules met zich mee. De acccuracy van de Decision Tree wordt geëvenaard bij breedtes van 0.3 met 32 decision rules. Het is opmerkelijk dat bij breedtes van 1 met slechts 8 decision rules een accuracy van 0.85 wordt behaald, aangezien breedtes van 0.8 slechts een accuracy van 0.73 teweeg brengen. Dit kan er op wijzen dat de decision rules met breedte 1 een zone correct classificeren die de decision rules met breedtes 0.8 niet correct capteren.

Voor de decision rules met verschillende breedtes vallen de resultaten eerder tegen. De beste accuracy die hier behaald wordt is 0.84 met 20 decision rules. De verschillende breedtes kunnen dus niet de hoogste accuracy van 0.98 bij gelijke breedtes evenaren.

5.2 Aantal decision rules

Zoals voorheen wordt het aantal decision rules verkregen via de methode in deze paper vergeleken met het aantal decision rules verkregen door de methode van Etinger. De methode van Etinger verliest geen accuracy ten opzichte van de Decision Tree, dus zijn decision table haalt een accuracy van 0.93. De decision table van Etinger bestaat uit 8 decision rules.

De accuracy van de decision table van Etinger kan geëvenaard worden met 32 decision rules. Dit betekent dat de bevonden decision table volgens de methode in deze paper 4 keer zo groot is als de decision table van Etinger. De beste accuracy die de methode in deze paper kan halen met hetzelfde aantal decision rules als de decision table van Etinger, namelijk 8, is 0.85. Dit wijst op een vermindering in accuracy van slechts 0.08 ten opzichte van de decision table van Etinger. Indien er geen rekening gehouden wordt met het aantal decision rules behaalt de methode in deze paper een accuracy van 0.98.

Etinger beschrijft in zijn paper een tekortkoming van zijn algoritme. Deze tekortkoming is dat machine learning algoritmen zeer goed omkunnen met hoog dimensionele data, maar dat de presentatie van die attributen in een decision table een bottleneck kan zijn (Etinger et al., 2019). Dit wordt ook bevestigd door Hasic, een groot aantal inputvariablen is hoog gecorreleerd met de perceptie van complexiteit van een decision table (Hasić & Vanthienen, 2019).

Naast het aantal variabelen is het ook interessant om het aantal decision rules te bekijken. Zoals blijkt uit de analyse is er bij de verplaatsing van 2 naar 3 dimensies een vrij grote stijging in decision rules. Zo bekomt een breedte van 0.1 in 2 dimensies slechts 22 decision rules terwijl breedtes van 0.1 in 3 dimensies 242 decision rules bekomen. Naarmate de data hogere dimensies aannemen, zal het aantal decision rules exponentiëel groeien. Indien we er van uitgaan dat de data genormaliseerd zijn en de breedte van de decision rules in elke dimensie hetzelfde is, kan het aantal decision rules beschreven worden door volgende formule. Waarbij $\lfloor \rceil$ de afronding naar de dichtsbijzijnde integer voorstelt en n het aantal dimensies van de inputdata.

$$2\lfloor rangedata/breedterule + 1 \rceil^{n-1}$$

Indien de ranges en de breedtes van de rules verschillen, kan het aantal decision rules beschreven worden door volgende formule.

$$2 * \lfloor rangedata_0/breedterule_0 + 1 \rceil * \lfloor rangedata_1/breedterule_1 + 1 \rceil ...$$

$$* \lfloor rangedata_{n-1}/breedterule_{n-1} + 1 \rceil$$

De paper van Hasic toont echter aan dat een groot aantal decision rules weinig gecorreleerd is met de perceptie van complexiteit van een decision table (Hasić & Vanthienen, 2019). Hierdoor is de impact op complexiteit van een groot aantal decision rules lager dan de impact van een groot aantal inputparameters.

5.3 Decision table

Net zoals voorheen worden er twee bekomen decision tables getoond. Figure 10 beschrijft de decision table gebaseerd op de breedtes berekend door het algoritme, hier worden slechts 27 van de 50 decision rules getoond. Figure 11 beschrijft de decision table met de hoogste accuracy, ook hier worden slechts de eerste 27 van de 242 decision rules getoond. Figure 10 en 11 zijn terug te vinden in de Appendix.

6 Conclusie

Aan de ene kant brengt de opkomst van Machine Learning algoritmen veel nieuwe mogelijkheden met zich mee. Nog nooit zijn er modellen geweest die zo goede voorspellingen of beslissingen kunnen maken als de modellen van vandaag de dag. Aan de andere kant missen deze modellen vaak echter transparantie en werken ze als een black box.

Deze paper stelde een methode voor om decision rules uit lineaire binaire Support Vector Machines te ontdekken die dan in DMN decision tables verzameld kunnen worden. Deze DMN decision tables hebben een hoge graad van transparantie en staan toe gebruikt te worden in gestandaardiseerde bedrijfsprocessen. De validatie van deze methode toonde aan dat het mogelijk is voor de decision tables om de accuracy van de Support Vector Machines te evenaren en tegelijkertijd een hogere transparantie aan te bieden. In een bepaalde case kon de accuracy van de onderliggende Support Vector Machine zelfs overtroffen worden. Het aantal decision rules waaruit de decision table bestond was ook vergelijkbaar met het aantal decision rules bevonden door de methode van Etinger die zich baseert op Decision Trees. De analyse toonde wel aan dat zowel het aantal decision rules als de accuracy van de decision table zeer afhankelijk zijn van de breedte van de rules. De breedte vinden die een goede balans geeft tussen het aantal decision rules en accuracy is een trial and error proces en is afhankelijk van de use case. Een tekortkoming verbonden aan dit algoritme is dat hoog dimensionele data moeilijk voor te stellen zijn in decision tables en dat dit ook een grote impact heeft op de complexiteit van de decision table. Verder is er ook aangetoond dat het aantal decision rules exponentiëel stijgt met het aantal dimensies van de inputdata. Tenslotte is deze methode slechts gevalideerd in 2 en 3 dimensies.

Toekomstig onderzoek kan focussen op het vinden van een optimale breedte van de decision rules voor bepaalde use cases. Verder is het mogelijk om de bevonden decision tables te decomposeren om het aantal decision rules te verkleinen. Tenslotte kan er geprobeerd worden om deze methode uit te breiden naar multi-class classification of Support Vector Machines die gebruik maken van de polynomial en radial kernel.

Referenties

Biard, T., Le Mauff, A., Bigand, M., & Bourey, J.-P. (2015). Separation of Decision Modeling from Business Process Modeling Using New "Decision Model and Notation" (DMN) for Automating Operational Decision-Making. In L. M. Camarinha-Matos, F. Bénaben, & W. Picard (Eds.), *Risks and Resilience of Collaborative Networks* (pp. 489–496). Cham: Springer International Publishing. doi: 10.1007/978-3-319-24141-8_45

Calvanese, D., Dumas, M., Laurson, , Maggi, F. M., Montali, M., & Teinemaa, I. (2018, November). Semantics, Analysis and Simplification of DMN Decision Tables. *Information Systems*, *78*, 112–125. Retrieved 2021-05-09, from `https://www.sciencedirect.com/science/article/pii/S0306437916306330` doi: 10.1016/j.is.2018.01.010

Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning - ICML '06* (pp. 161–168). Pittsburgh, Pennsylvania: ACM Press. Retrieved 2021-04-18, from `http://portal.acm.org/citation.cfm?doid=1143844.1143865` doi: 10.1145/1143844.1143865

Dua, D., & Graff, C. (2017). *UCI Machine Learning Repository*. University of California, Irvine, School of Information and Computer Sciences. Retrieved from `http://archive.ics.uci.edu/ml`

The ElementTree XML API documentation. (n.d.). Retrieved 2021-05-17, from `https://docs.python.org/3/library/xml.etree.elementtree.html#module-xml.etree.ElementTree`

Etinger, D., Simić, S., & Buljubašić, L. (2019, May). Automated decision-making with DMN: from decision trees to decision tables. In *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)* (pp. 1309–1313). (ISSN: 2623-8764) doi: 10.23919/MIPRO.2019.8756694

Figl, K., Mendling, J., Tokdemir, G., & Vanthienen, J. (2018). What we know and what we do not know about DMN. *Enterprise Modelling and Information Systems Architectures – International Journal of Conceptual Modeling*, *13*(2), 1–16. Retrieved 2021-05-17, from `https://lirias.kuleuven.be/retrieve/504891` (Publisher: Gesellschaft für Informatik) doi: 10.18417/emisa.13.2

Hasić, F., & Vanthienen, J. (2019, July). Complexity metrics for DMN decision models. *Computer Standards & Interfaces*, *65*, 15–37. Retrieved 2021-05-17, from `https://linkinghub.elsevier.com/retrieve/pii/S0920548918303647` doi: 10.1016/j.csi.2019.01.006

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, *9*(3), 90–95. doi: 10.1109/MCSE.2007.55

Kecman, V. (2005, 05). Support vector machines – an introduction. In (Vol. 177, p. 605-605). doi: 10.1007/10984697_1

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research*, *12*(Oct), 2825–2830.

Waskom, M. L. (2021). seaborn: statistical data visualization. *Journal of Open Source Software*, *6*(60), 3021. Retrieved from `https://doi.org/10.21105/joss.03021` doi: 10.21105/joss.03021

7 Appendix

Decision Table | Hit Policy: Unique

	When Sepal width double	And Sepal length double	Then Class integer	Annotations
1	<0.289	<=0.013	1	
2	<0.289	>0.013	0	
3	[0.289..0.867 [	<=0.618	1	
4	[0.289..0.867 [	>0.618	0	
5	>=0.867	<=1.222	1	
6	>=0.867	>1.222	0	
+	-	-		

Figure 4: Decision Table Iris berekende breedte (acc. 0.88)

Decision Table | Hit Policy: Unique

	When Sepal width double	And Sepal length double	Then Class integer	Annotations
1	<0.05	<=0.013	1	
2	<0.05	>0.013	0	
3	[0.05..0.15 [	<=0.118	1	
4	[0.05..0.15 [	>0.118	0	
5	[0.15..0.25 [	<=0.222	1	
6	[0.15..0.25 [	>0.222	0	
7	[0.25..0.35 [	<=0.327	1	
8	[0.25..0.35 [	>0.327	0	
9	[0.35..0.45 [	<=0.431	1	
10	[0.35..0.45 [	>0.431	0	
11	[0.45..0.55 [	<=0.536	1	
12	[0.45..0.55 [	>0.536	0	
13	[0.55..0.65 [	<=0.64	1	
14	[0.55..0.65 [	>0.64	0	
15	[0.65..0.75 [	<=0.745	1	
16	[0.65..0.75 [	>0.745	0	
17	[0.75..0.85 [	<=0.849	1	
18	[0.75..0.85 [	>0.849	0	
19	[0.85..0.95 [	<=0.954	1	
20	[0.85..0.95 [	>0.954	0	
21	>=0.95	<=1.058	1	
22	>=0.95	>1.058	0	
+	-	-		

Figure 5: Decision Table Iris beste accuracy (acc. 1.00)

Decision Table | Hit Policy: Unique

	When Color intensity (double)	And Alcohol (double)	Then Class (integer)	Annotations
1	<0.288	<=1.28	1	
2	<0.288	>1.28	0	
3	[0.288..0.865 [	<=0.364	1	
4	[0.288..0.865 [	>0.364	0	
5	>=0.865	<=-0.552	1	
6	>=0.865	>-0.552	0	
+	-	-		

Figure 6: Decision Table Wine berekende breedte (acc. 0.94)

Decision Table | Hit Policy: Unique

	When Color intensity (double)	And Alcohol (double)	Then Class (integer)	Annotations
1	<0.5	<=1.28	1	
2	<0.5	>1.28	0	
3	>=0.5	<=-0.309	1	
4	>=0.5	>-0.309	0	
+	-	-		

Figure 7: Decision Table Wine beste accuracy (acc. 0.97)

Decision Table | Hit Policy: Unique

	When mean texture (double)	And mean area (double)	Then Class (integer)	Annotations
1	<0.541	<=0.358	1	
2	<0.541	>0.358	0	
3	>=0.541	<=0.084	1	
4	>=0.541	>0.084	0	
+	-	-		

Figure 8: Decision Table Breast Cancer Wisconsin berekende breedte (acc. 0.81)

Decision Table | Hit Policy: Unique

	When	And	Then	
	mean texture (double)	mean area (double)	Class (integer)	Annotations
1	<0.4	<=0.358	1	
2	<0.4	>0.358	0	
3	>=0.4	<=0.156	1	
4	>=0.4	>0.156	0	
+	-	-		

Figure 9: Decision Table Breast Cancer Wisconsin beste accuracy (acc. 0.92)

Decision Table | Hit Policy: Unique

	When	And	And	Then	
	Variance (double)	Skewness (double)	Curtosis (double)	Class (integer)	Annotations
1	<0.126	<0.131	<=1.351	1	
2	<0.126	<0.131	>1.351	0	
3	[0.126..0.379 [	<0.131	<=1.096	1	
4	[0.126..0.379 [	<0.131	>1.096	0	
5	[0.379..0.632 [	<0.131	<=0.841	1	
6	[0.379..0.632 [	<0.131	>0.841	0	
7	[0.632..0.885 [	<0.131	<=0.586	1	
8	[0.632..0.885 [	<0.131	>0.586	0	
9	>=0.885	<0.131	<=0.332	1	
10	>=0.885	<0.131	>0.332	0	
11	<0.126	[0.131..0.394 [	<=1.105	1	
12	<0.126	[0.131..0.394 [	>1.105	0	
13	[0.126..0.379 [	[0.131..0.394 [	<=0.851	1	
14	[0.126..0.379 [	[0.131..0.394 [	>0.851	0	
15	[0.379..0.632 [	[0.131..0.394 [	<=0.596	1	
16	[0.379..0.632 [	[0.131..0.394 [	>0.596	0	
17	[0.632..0.885 [	[0.131..0.394 [	<=0.341	1	
18	[0.632..0.885 [	[0.131..0.394 [	>0.341	0	
19	>=0.885	[0.131..0.394 [	<=0.086	1	
20	>=0.885	[0.131..0.394 [	>0.086	0	
21	<0.126	[0.394..0.657 [	<=0.86	1	
22	<0.126	[0.394..0.657 [	>0.86	0	
23	[0.126..0.379 [	[0.394..0.657 [	<=0.605	1	
24	[0.126..0.379 [	[0.394..0.657 [	>0.605	0	
25	[0.379..0.632 [	[0.394..0.657 [	<=0.35	1	
26	[0.379..0.632 [	[0.394..0.657 [	>0.35	0	
27	[0.632..0.885 [	[0.394..0.657 [	<=0.095	1	

Figure 10: Decision Table Bank Note Authentication berekende breedtes (acc. 0.94)

Decision Table | Hit Policy: Unique

	When	And	And	Then	
	Variance (double)	Skewness (double)	Curtosis (double)	Class (integer)	Annotations
1	<0.05	<0.05	<=1.351	1	
2	<0.05	<0.05	>1.351	0	
3	[0.05..0.15 [	<0.05	<=1.25	1	
4	[0.05..0.15 [	<0.05	>1.25	0	
5	[0.15..0.25 [	<0.05	<=1.149	1	
6	[0.15..0.25 [	<0.05	>1.149	0	
7	[0.25..0.35 [	<0.05	<=1.049	1	
8	[0.25..0.35 [	<0.05	>1.049	0	
9	[0.35..0.45 [	<0.05	<=0.948	1	
10	[0.35..0.45 [	<0.05	>0.948	0	
11	[0.45..0.55 [	<0.05	<=0.847	1	
12	[0.45..0.55 [	<0.05	>0.847	0	
13	[0.55..0.65 [	<0.05	<=0.746	1	
14	[0.55..0.65 [	<0.05	>0.746	0	
15	[0.65..0.75 [	<0.05	<=0.645	1	
16	[0.65..0.75 [	<0.05	>0.645	0	
17	[0.75..0.85 [	<0.05	<=0.544	1	
18	[0.75..0.85 [	<0.05	>0.544	0	
19	[0.85..0.95 [	<0.05	<=0.443	1	
20	[0.85..0.95 [	<0.05	>0.443	0	
21	>=0.95	<0.05	<=0.343	1	
22	>=0.95	<0.05	>0.343	0	
23	<0.05	[0.05..0.15 [	<=1.258	1	
24	<0.05	[0.05..0.15 [	>1.258	0	
25	[0.05..0.15 [	[0.05..0.15 [	<=1.157	1	
26	[0.05..0.15 [	[0.05..0.15 [	>1.157	0	
27	[0.15..0.25 [	[0.05..0.15 [	<=1.056	1	

Figure 11: Decision Table Bank Note Authentication beste accuracy (acc. 0.98)