Verslag
Proximity graphs

Mathijs Vos, Ramon Janssen, Petra van den Bos

4 maart 2012

Inhoudsopgave

Samenvatting

Abstract hier

1 Inleiding

a short introduction about classification in general and kNN and their applications.

2 Het Probleem

say what is condensing and editing

2.1 Condensing en editing

Het kNN-principe kan op een simpele manier op alle soorten gegevens worden toegepast. Door een nieuw punt simpelweg met alle punten in de trainingset te vergelijken kan een goed resultaat worden verkregen. In de praktijk levert dit echter problemen op in performance, omdat een dataset al snel veel dimensies en punten bevat. Ook wordt ruis daarmee overgenomen, omdat van elk punt van de trainingset wordt aangenomen dat deze correct gelabeld is. De complexiteit voor het testen van nieuwe punten wordt dan al snel veel te groot en daarvoor zijn optimalisaties vereist.

Het doel van condensing is om de grootte van een dataset te reduceren door redundante informatie te schrappen. Dit richt zich voornamelijk op gebieden waar zich veel punten met dezelfde klasse bevinden: de punten in het midden van het gebied voegen niets toe ten opzichte van de punten op de rand. Alleen de randen worden dus bespaard. Hierbij zorgt ruis voor een probleem, omdat n (verkeerd gelabeld) punt in het midden van een gebied er anders voor zou zorgen dat het midden niet meer condensed kan worden.

Voor het weghalen van ruis is er editing. Dit haalt punten weg die nabij veel punten met een andere classificatie liggen, omdat dit soort punten meer kans hebben om ruis te zijn. Dit kan het beste vr het condensen gebeuren, zodat de condensing minder last heeft van ruis.

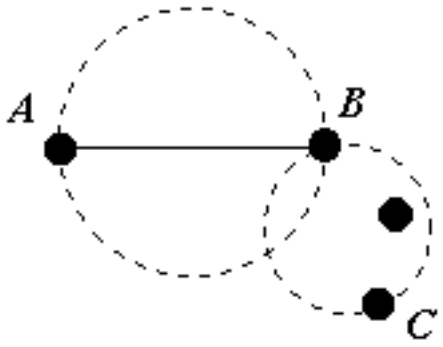
Voor zowel condensing als editing kan de precieze werking verschillen. Het is afhankelijk van het algoritme hoeveel informatie er verwijderd wordt. Als er meer informatie verwijderd wordt worden de resultaten gemiddeld slechter maar wordt de benodigde tijd voor het testen minder. Het verschilt ook per algoritme hoe effectief editing en condensing is onder verschillende omstandigheden, zoals de hoeveelheid dimensies en de vorm van de data.

The selected method: describe the method you selected: why this method, pseudo-code and characteristics you consider important

2.2 De methode

De titel van het artikel dat wij gekozen hebben is: Prototype selection for the nearest neighbour rulethrough proximity graphs. We hebben dit artikel gekozen omdat het ons wel leuk leek om grafen te gebruiken en omdat het artikel duidelijk was. Er worden twee soorten grafen beschreven: Gabriel Graphs en Relative Neighbourhood Graph. Beiden zijn grafen waarin punten verbonden worden door lijnen die elkaar niet snijden. Als een punt A verbonden is met punt B, is A een buur van B en B een buur van A.

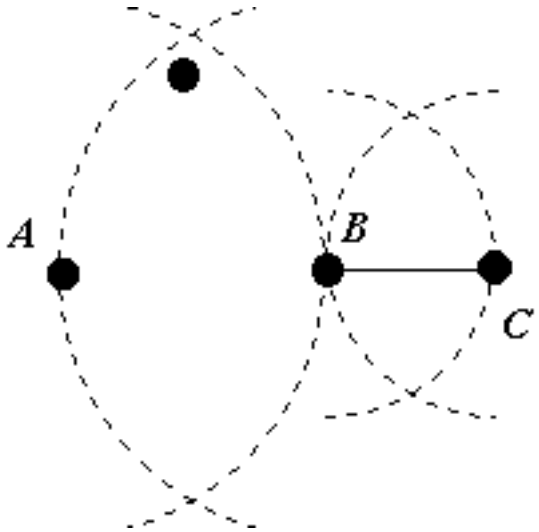
In een Gabriel Graph is er een verbinding tussen twee punten, als binnen de cirkel, waarvan de verbinding de diameter is, geen andere punten liggen. Met de euclidische afstand kan dit formeel uitgedrukt worden: A en B hebben een verbinding als voor alle andere punten X: afstand2(A,B)¡= afstand2(A,X) + afstand2(B,X). In Figuur 1 zijn A en B verbonden, maar B en C niet. In figuur 2 is een gehele Gabriel Graph te zien.



Figuur 1: Buren van punten in een Gabriel Graph

TODO:
Plaatje van gehele Gabriel Graph uit GraphVisualiser
Figuur 2

In een Relative Neighbourhood Graph hebben twee punten een verbinding als er geen andere punten binnen de lune van die twee punten liggen, zie figuur 3. Met de euclidische afstand kan dit formeel uitgedrukt worden: A en B hebben een verbinding als voor alle andere punten X: afstand(A,B) ¡= max{afstand(A,X),afstand(B,X)}. In Figuur 2 zijn B en C verbonden, maar A en B niet. In figuur 4 is een gehele Relative Neighbourhood Graph te zien.



Figuur 2: Buren van punten in een Relative Neighbourhood Graph

TODO:
Plaatje van gehele Relative Neighbourhood Graph uit GraphVisualiser
Figuur 4

Deze grafen kun je gebruiken om nieuwe data (de testset) te klassificeren aan de hand van data waarvan de classificatie al bekend is (de trainset). Een graaf bevat in eerste instantie dus alle punten uit de trainset. Om de performance te verbeteren en eventuele ruis te compenseren, wordt condensing en editing toegepast op de trainset. Hierbij wordt gekeken naar de punten die een verbinding hebben met een bepaald punt. Bij condensing wordt ieder punt weggehaald, dat alleen verbindingen naar punten met dezelfde klasse heeft. In het artikel werden twee soorten editing algoritmes besproken: een eerste orde editing algoritme en een tweede orde editing algoritme. Het eerste orde editing algoritme bepaald voor elk punt of de meerderheid van de buren van dat punt tot een van de andere klasses behoort. De punten waarvoor dit geldt, worden verwijderd. Het tweede orde algoritme verwijderd alle punten als geldt dat de meerderheid van de buren tot een van de andere klasses behoort, en als ook de meerderheid van de buren, van de buren die van dezelfde klasse als dat punt zijn, tot een van de andere klasses behoort.

De testset wordt geklassificeerd op de volgende manier: Eerst wordt de graaf van alle punten van de trainset geconstrureerd. Vervolgens wordt eerste of tweede orde editing toegepast. Daarna wordt condensing toegepast. Met de bewerkte graaf die overblijft wordt een klasse aan elk van de punten in de testset toegewezen. Dit gebeurt behulp van het k-nearest neighbour algoritme. Van elk punt in de testset wordt gekeken welke k punten het dichtste bij liggen. Deze k dichtstbijzijnde punten bepalen we ofwel door alleen naar de directe buren van het testsetpunt te kijken (eerste orde), ofwel naar de directe buren en buren van die directe buren te kijken. Als k groter is dan het aantal buren dat het testsetpunt heeft, worden alleen die buren gebruikt, waardoor k voor dat punt dus minder is.

3 Experimenten

Report results of experiments with the selected method on the considered datasets, in particular accuracy and storage reduction reduction.

We hebben onze methode met verschillende datasets getest:

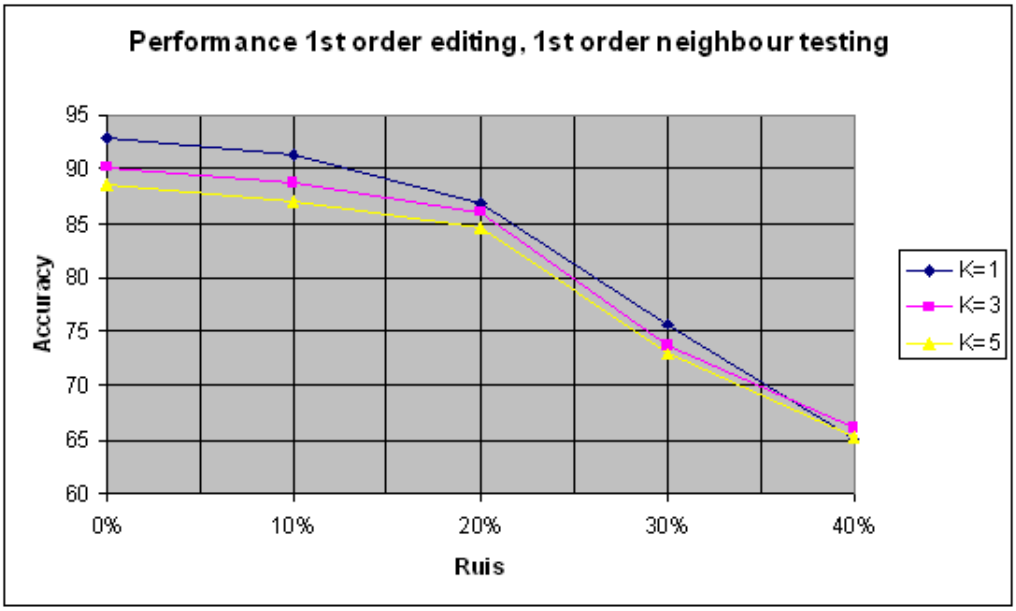
1. de XOR-dataset met 2 dimensies en 2 mogelijke klasses
2. MNIST-dataset met 196 dimensies en 10 mogelijke klasses
3. de diabetes-dataset met 8 dimensies en 2 mogelijke klasses

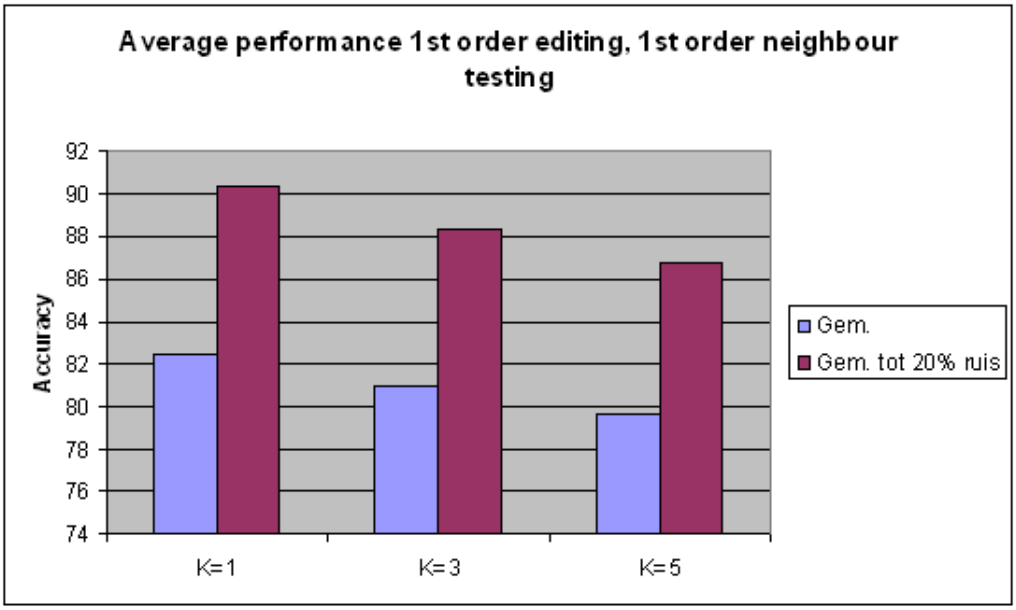
3.1 XOR

We hebben de gemiddelde nauwkeurigheid en reductie voor elke 'error-rate' bepaald, door het gemiddelde te nemen van de score van alle testsets (dat waren er 40), toegepast op alle trainsets (er waren 10 trainsets). Hierbij hebben we Gabriel Graphs gebruikt. We hebben deze tests gedaan voor eerste orde en tweede orde editing, in combinatie met eerste orde en tweede orde (k-nearest-neighbour) tests op de punten van de testset. Hieronder staat de testresultaten weergegeven in een tabel en daarbij twee grafieken die de resultaten uit de tabel weergeven.

3.1.1 Eerste orde editing, eerste orde tests

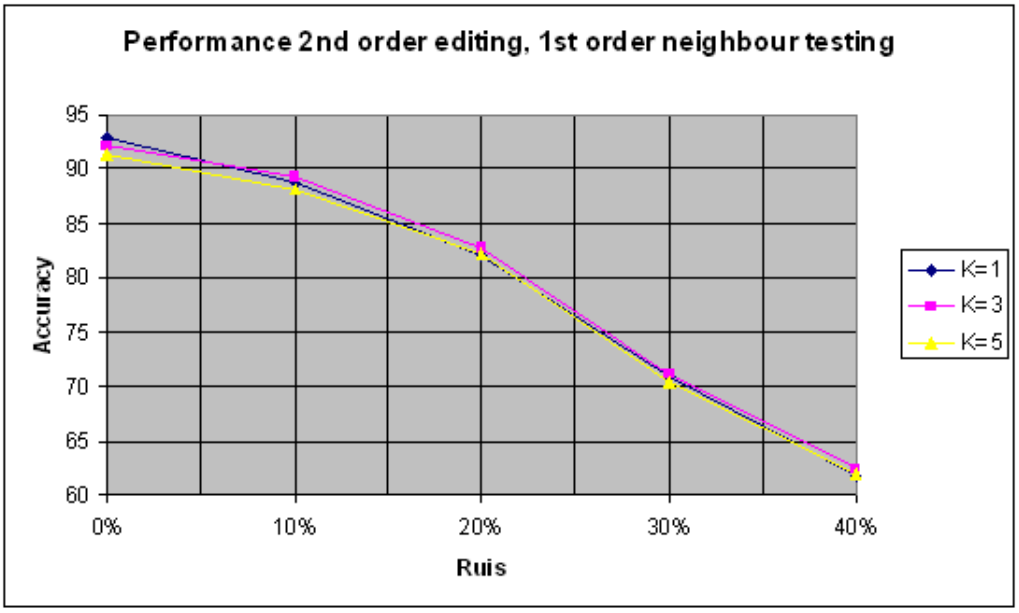
Ruis	K=1	K=3	K=5
0%	92,9975	90,17	88,58
10%	91,37	88,6925	86,945
20%	86,7875	86,07	84,625
30%	75,7175	73,7675	72,9125
40%	65,185	66,11	65,31
Gem.	82,4115	80,962	79,6745
Gem. tot 20% ruis	90,385	88,31083333	86,71666667

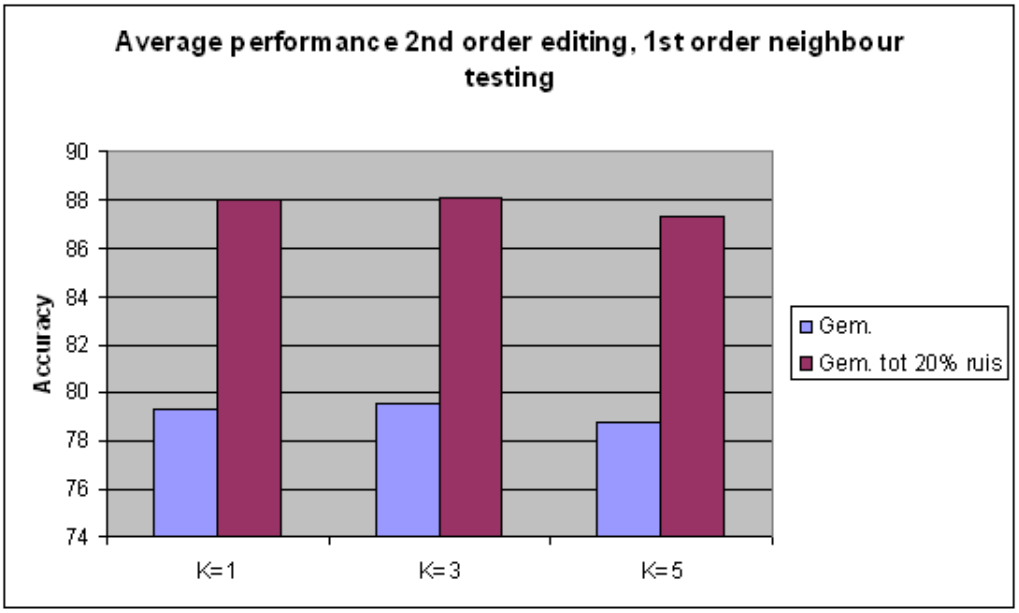




3.1.2 Tweede orde editing, eerste orde tests

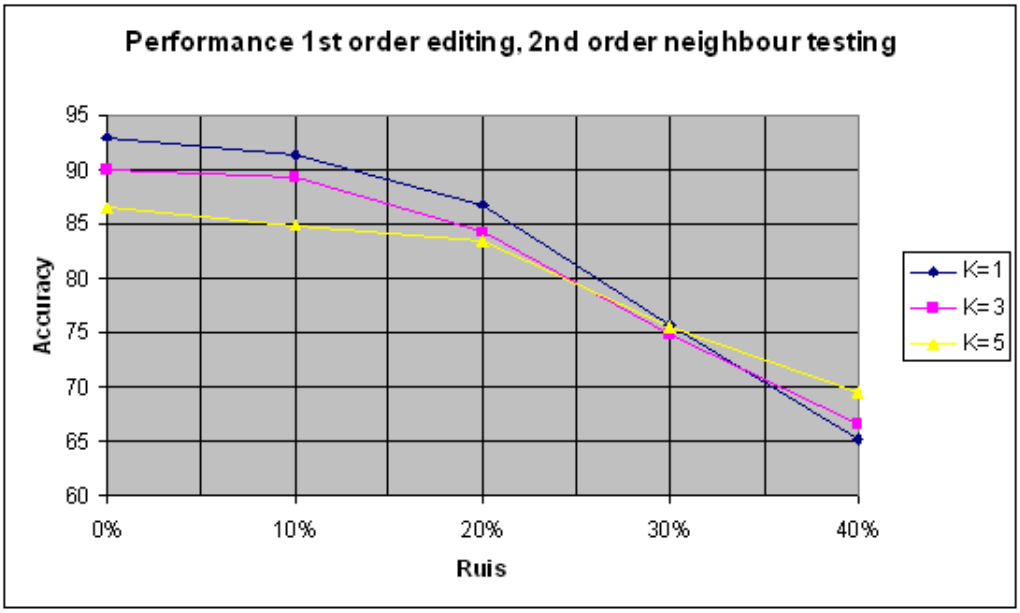
Ruis	K=1	K=3	K=5
0%	93,005	92,1775	91,3025
10%	88,7825	89,205	88,2175
20%	82,085	82,7275	82,34
30%	70,8475	71,06	70,4
40%	61,8275	62,33	61,9375
Gem.	79,3095	79,5	78,8395
Gem. tot 20% ruis	87,9575	88,03666667	87,28666667

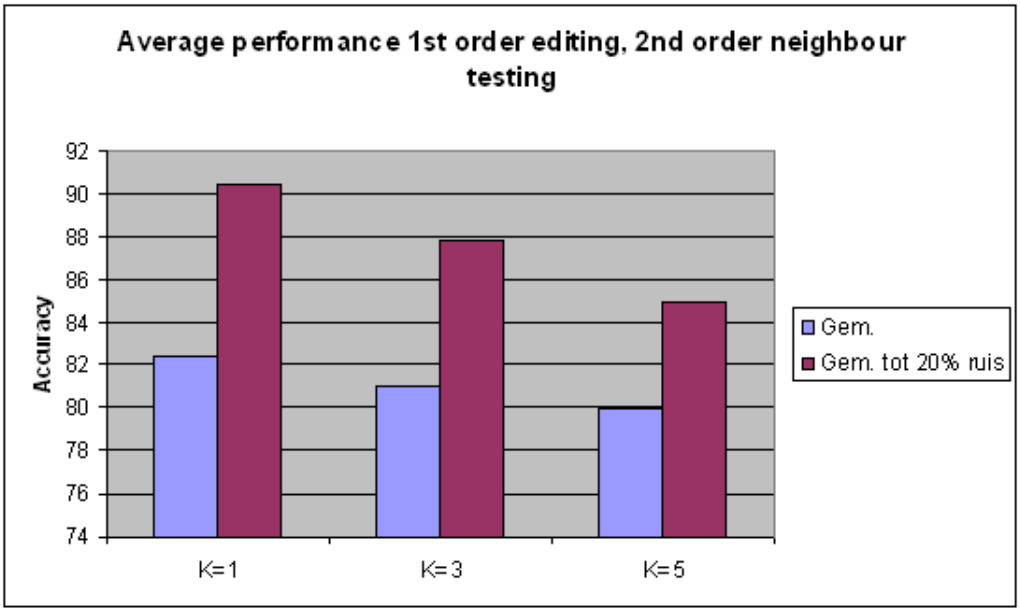

Performance 2nd order editing, 1st order neighbour testing
95
90
85
80
75
70
65
60
Accuracy
0%
10%
20%
30%
40%
Ruis
K=1
K=3
K=5


Average performance 2nd order editing, 1st order neighbour testing
90
88
86
84
82
80
78
76
74
Accuracy
K=1
K=3
K=5
Gem.
Gem. tot 20% ruis

3.1.3 Eerste orde editing, tweede orde tests

Ruis	K=1	K=3	K=5
0%	92,9975	89,9475	86,515
10%	91,37	89,29	84,865
20%	86,7875	84,17	83,49
30%	75,7175	74,83	75,4975
40%	65,185	66,7225	69,54
Gem.	82,4115	80,992	79,9815
Gem. tot 20% ruis	90,385	87,8025	84,95666667

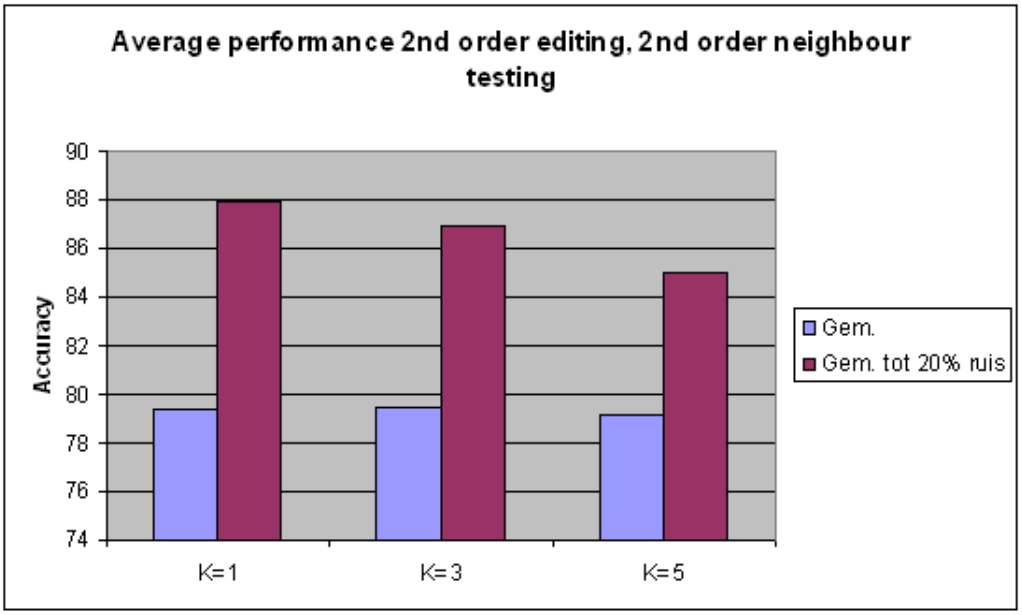




3.1.4 Tweede orde editing, tweede orde tests

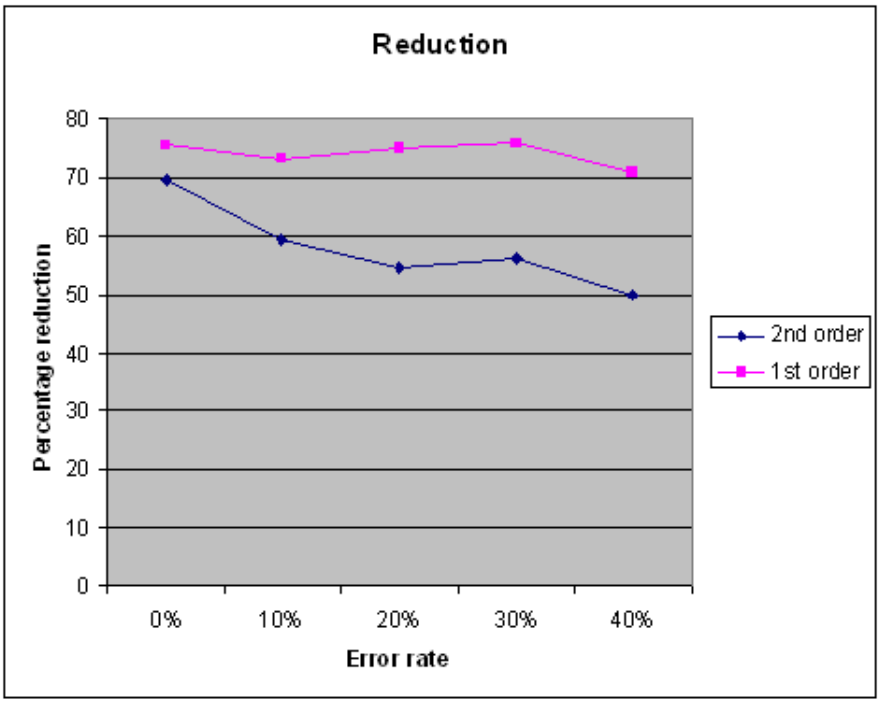
Ruis	K=1	K=3	K=5
0%	93,005	90,065	86,7675
10%	88,7825	88,25	85,98
20%	82,085	82,4625	82,4
30%	70,8475	73,93	75,735
40%	61,8275	62,2375	64,6
Gem.	79,3095	79,389	79,0965
Gem. tot 20% ruis	87,9575	86,92583333	85,04916667


Performance 2nd order editing, 2nd order neighbour testing
95
90
85
80
75
70
65
60
Accuracy
0%
10%
20%
30%
40%
Ruis
K=1
K=3
K=5


Average performance 2nd order editing, 2nd order neighbour testing
90
88
86
84
82
80
78
76
74
Accuracy
K=1
K=3
K=5
Gem.
Gem. tot 20% ruis

3.1.5 Reductie

	0%	10%	20%	30%	40%
2nd order editing	69,7	59,3	54,6	56	50
1st order editing	75,6	73,1	74,9	75,9	70,9



3.1.6 Conclusies

Nauwkeurigheid van het klassificeren:
Uit de staaf grafieken blijkt dat voor de xor-dataset gemiddeld het beste k=1 gebruikt kan worden voor het k-nearest-neighbour algoritme, als er weinig ruis in de dataset zit. Als er veel ruis in de dataset zit, levert k=5 gemiddeld een zelfde of beter resultaat. Eerste orde editing levert gemiddeld een iets beter resultaat, vooral voor k=1. De orde van testen heeft niet zo veel invloed, dit kan misschien verklaard worden doordat het aantal buren van een punt vaak groter is dan k, waardoor tweede orde testen ook alleen tests doet met alleen de directe buren.

Reductie van het aantal punten in de trainset:
Zeker bij meer ruis, verwijderd eerste orde editing meer punten. Dat is ook wel logisch, want tweede orde editing test of het punt verwijderd zou worden met eerste orde editing, waarna het punt niet altijd verwijderd wordt als eerste orde editing dat zou ddoen, door de extra conditie van tweede orde editing.

3.2 MNIST

3.3 Diabetes

Discuss the performance of the selected method.

4 Conclusie

Summarize the content of your report and provide some conclusive remarks and observation about the effectiveness of the selected method and possible future work (improvements).