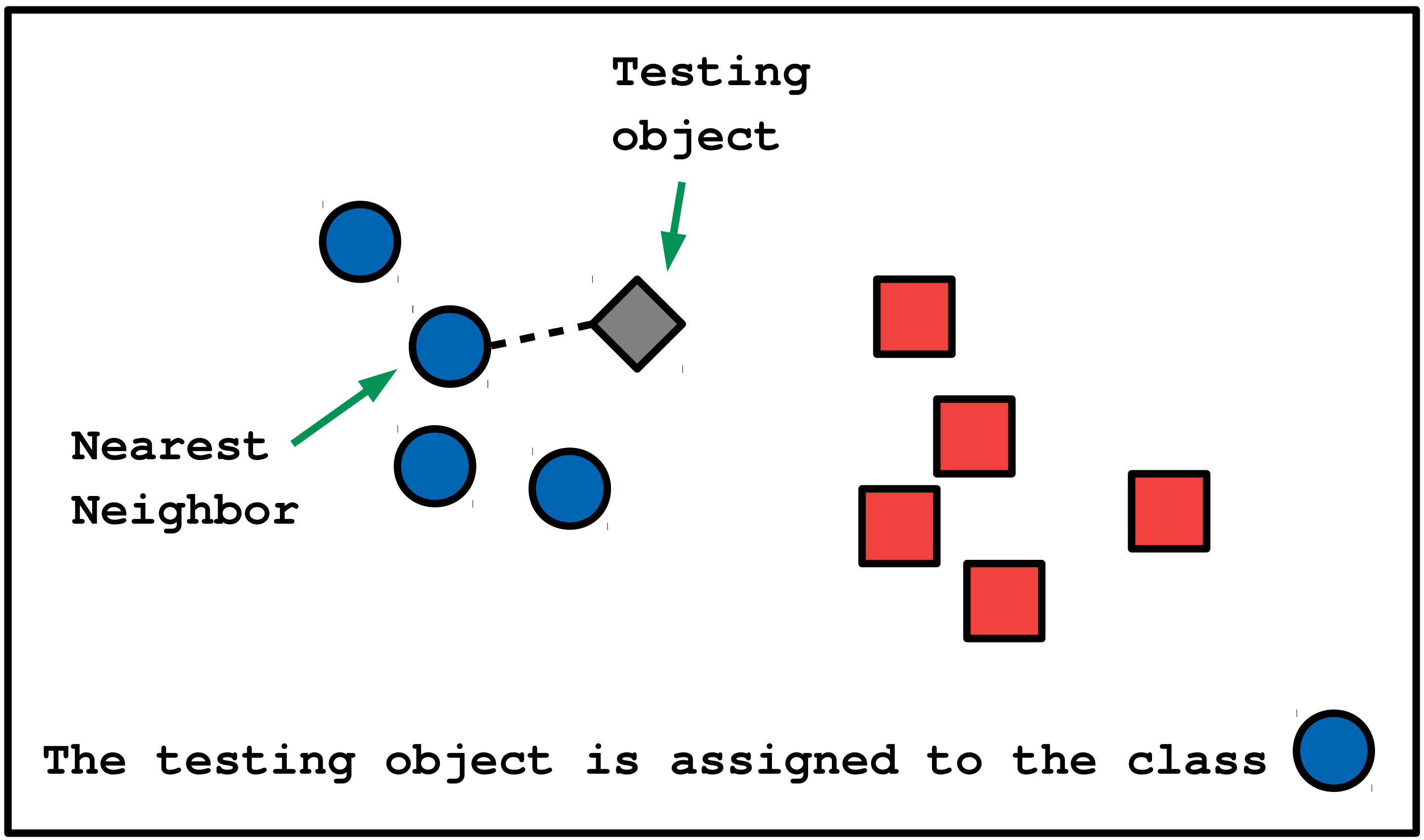
Int. Conf. on Pattern Recognition (ICPR) 2020

PowerHC: non linear normalization of distances for advanced nearest neighbor classification

Manuele Bicego, Mauricio Orozco-Alzate
University of Verona (Italy), Universidad Nacional de Colombia (Colombia)
manuele.bicego@univr.it, morozcoa@unal.edu.co

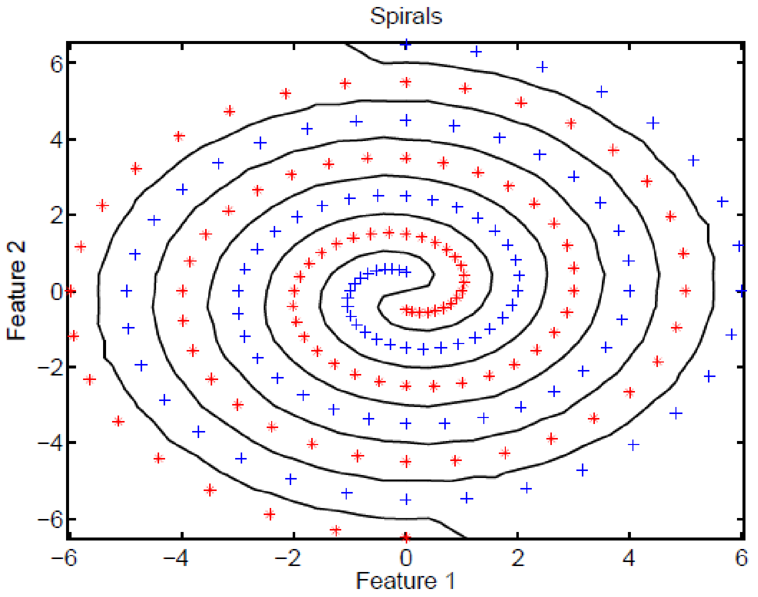
The nearest neighbor rule

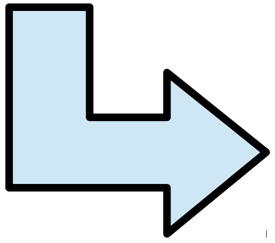
- The simplest classification technique: it assigns the testing object to the class of the **most similar object** of the training set (the **nearest neighbor**)



The nearest neighbor rule

- Many advantages:
  - Accurate: non linear classifier
  - No parameters
  - It works also for non vectorial data
  - Many theoretical results available
  - Simple and interpretable





Crucial need in recent years:

eXplainable Artificial Intelligence!

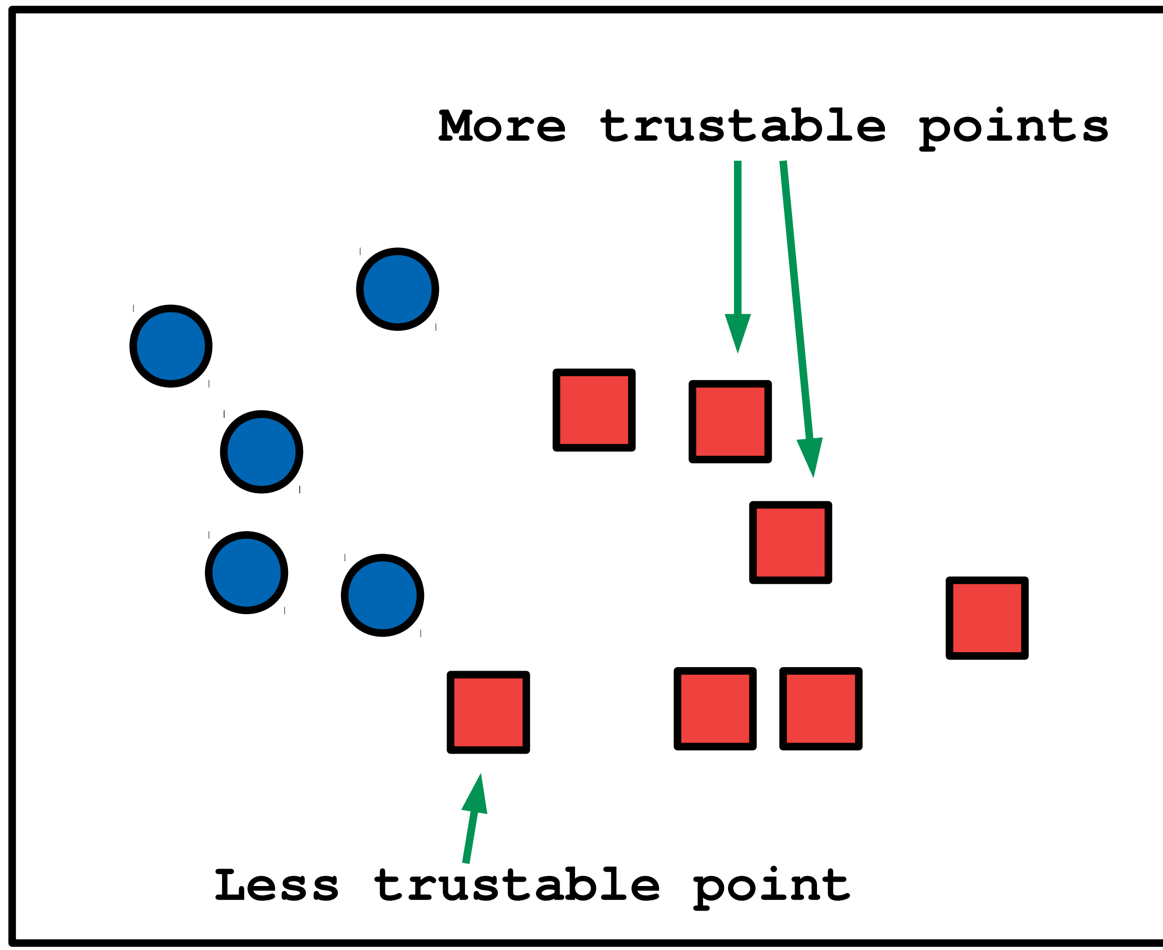
Variants of Nearest Neighbor

- Several variants have been proposed in literature to improve this technique (Condensing, Editing, Adaptation, Discriminative information...)
- Among others, there are two interesting variants based on a similar idea:
  - **ANN**: the Adaptive Nearest Neighbor rule (Wang et al. 2007)
  - **HC**: the Hypersphere Classifier (Lopes et al. 2015)

The idea behind ANN and HC

- ***Starting observation**: Some training objects are "better" than others for the NN rule*

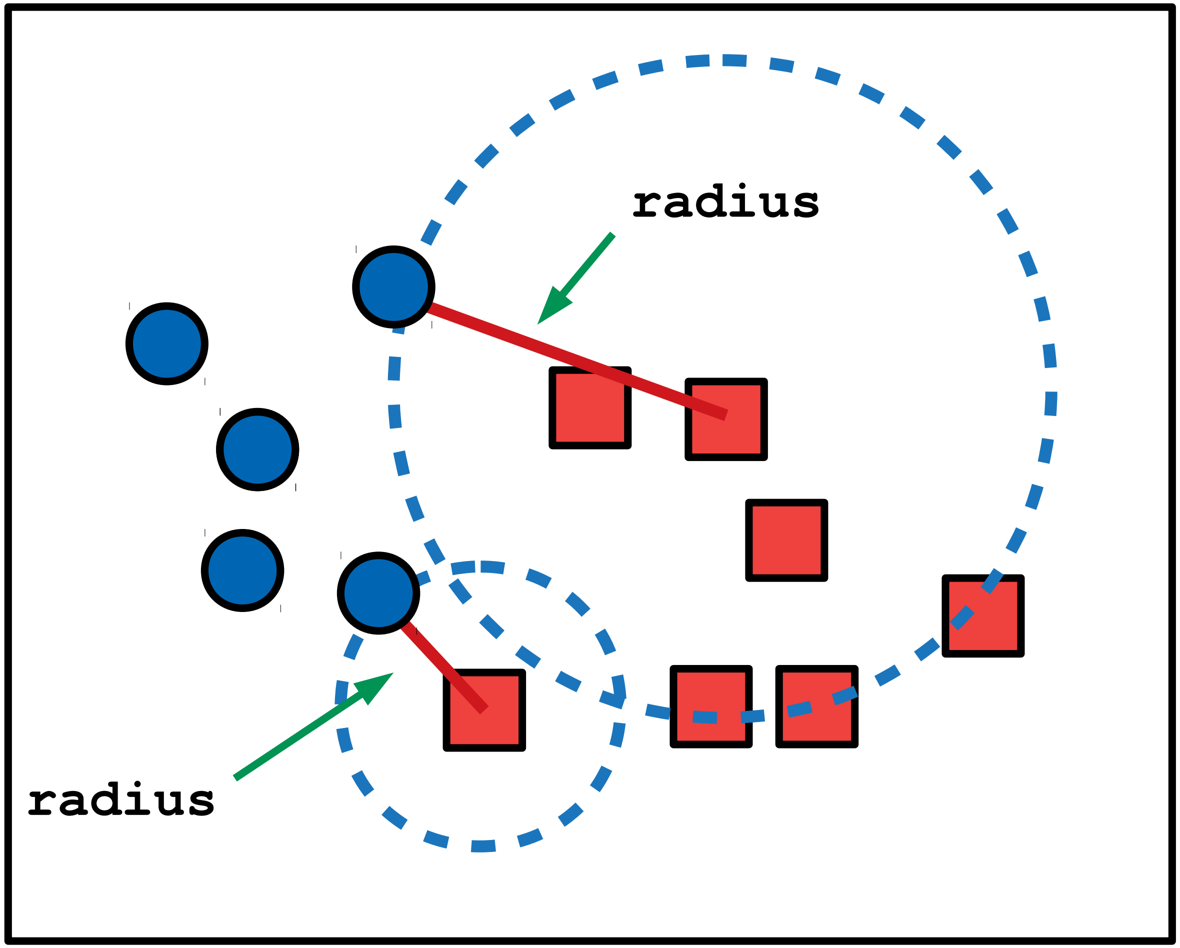
Points which are **far away from other classes** are more "trustable" that points which are not



The idea behind ANN and HC

- **ANN** and **HC** quantify this concept by defining the **radius of a training point**: “The distance from its nearest object of **another class**”

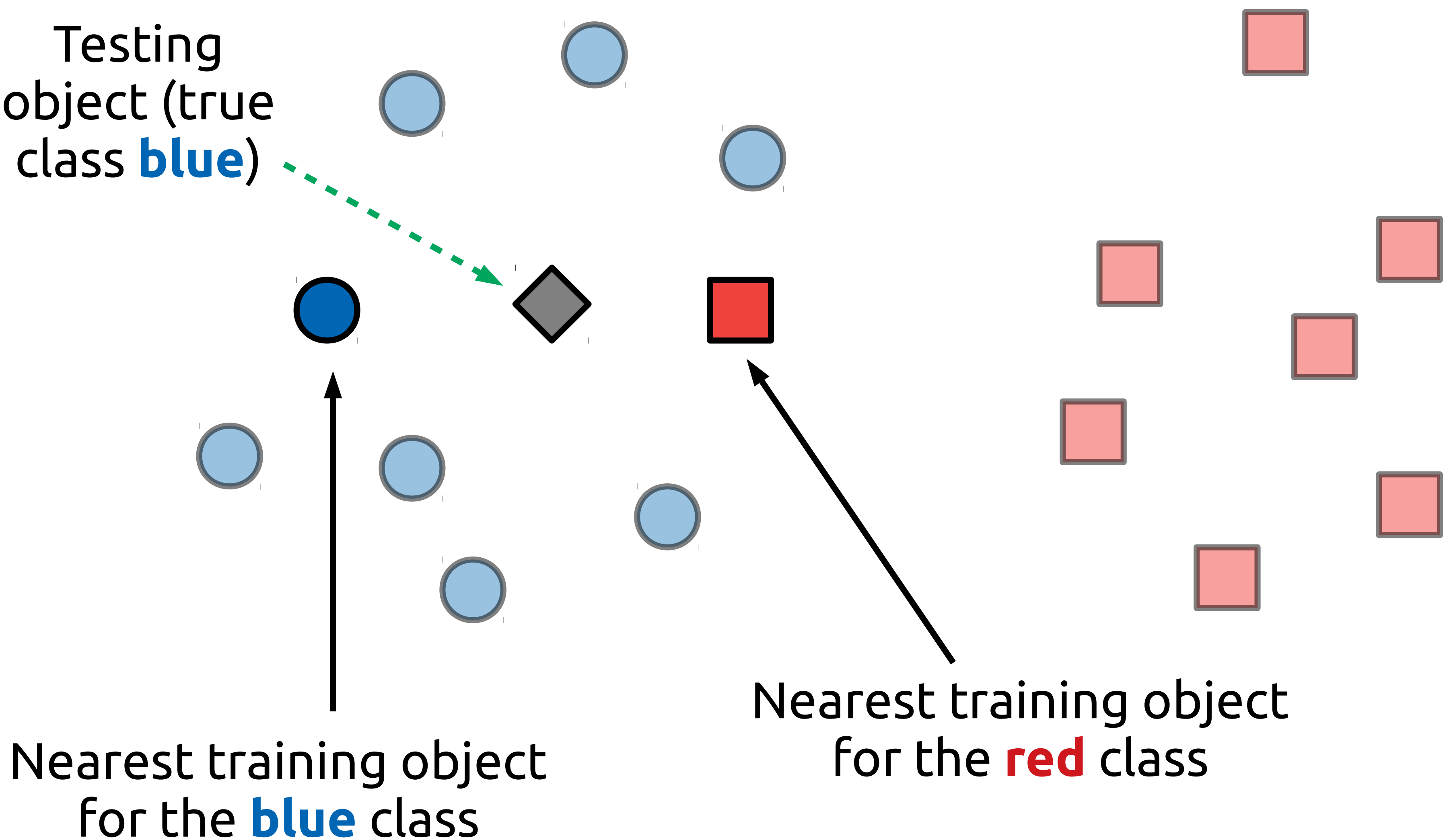
The larger the radius the more trustable (the better) the point

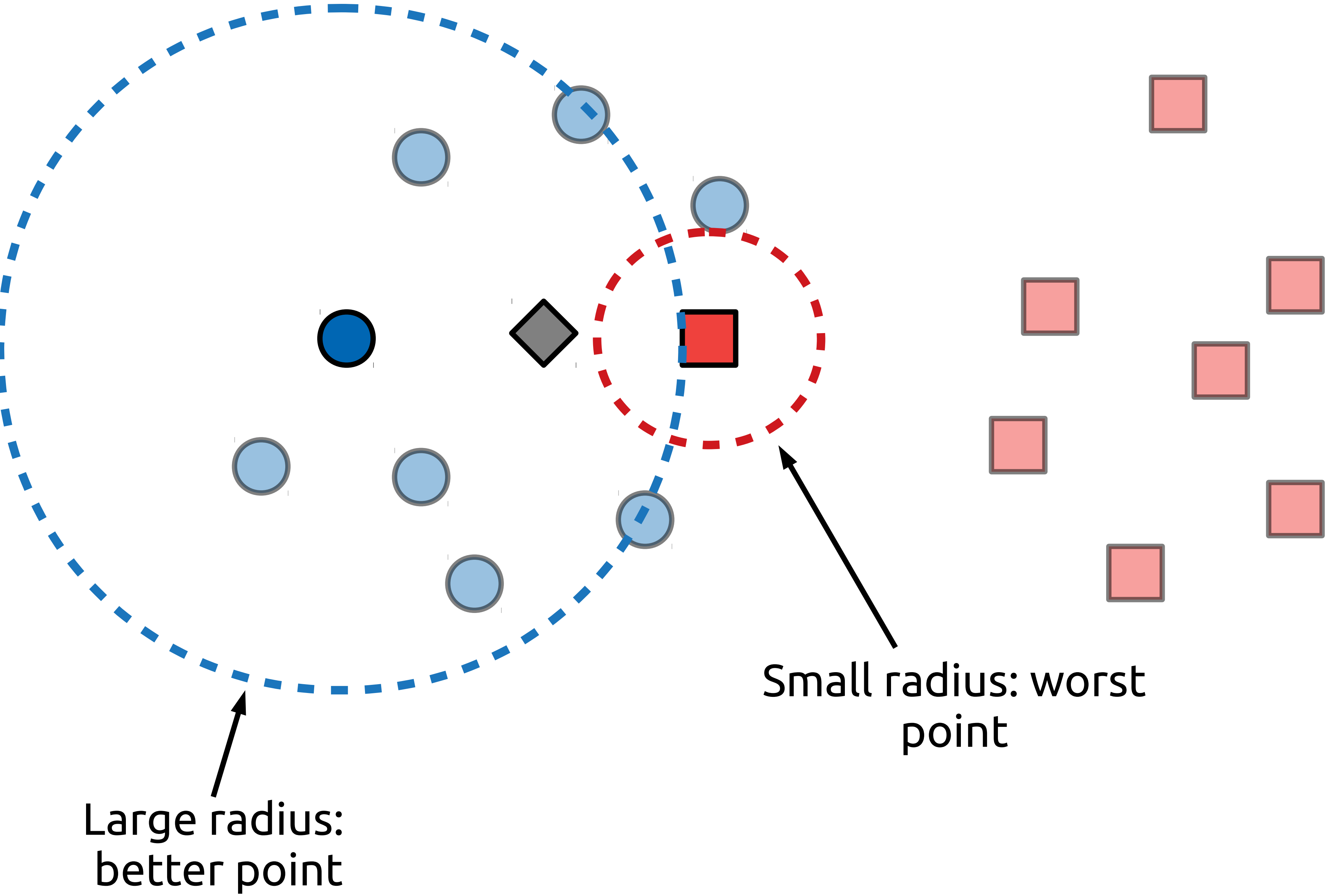


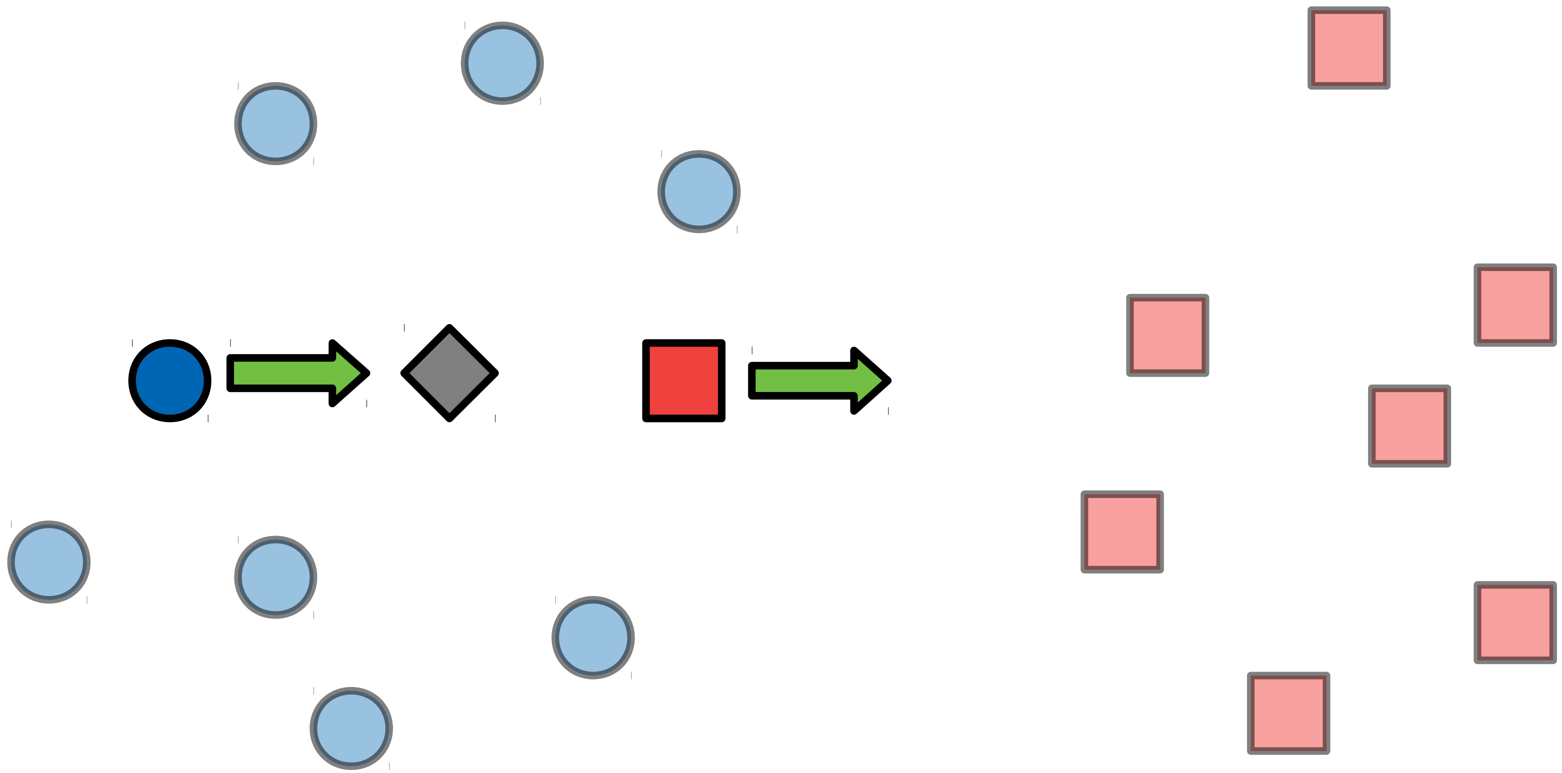
The idea behind ANN and HC

- ANN and HC employ a mechanism **to favour** in the NN rule “better” training points

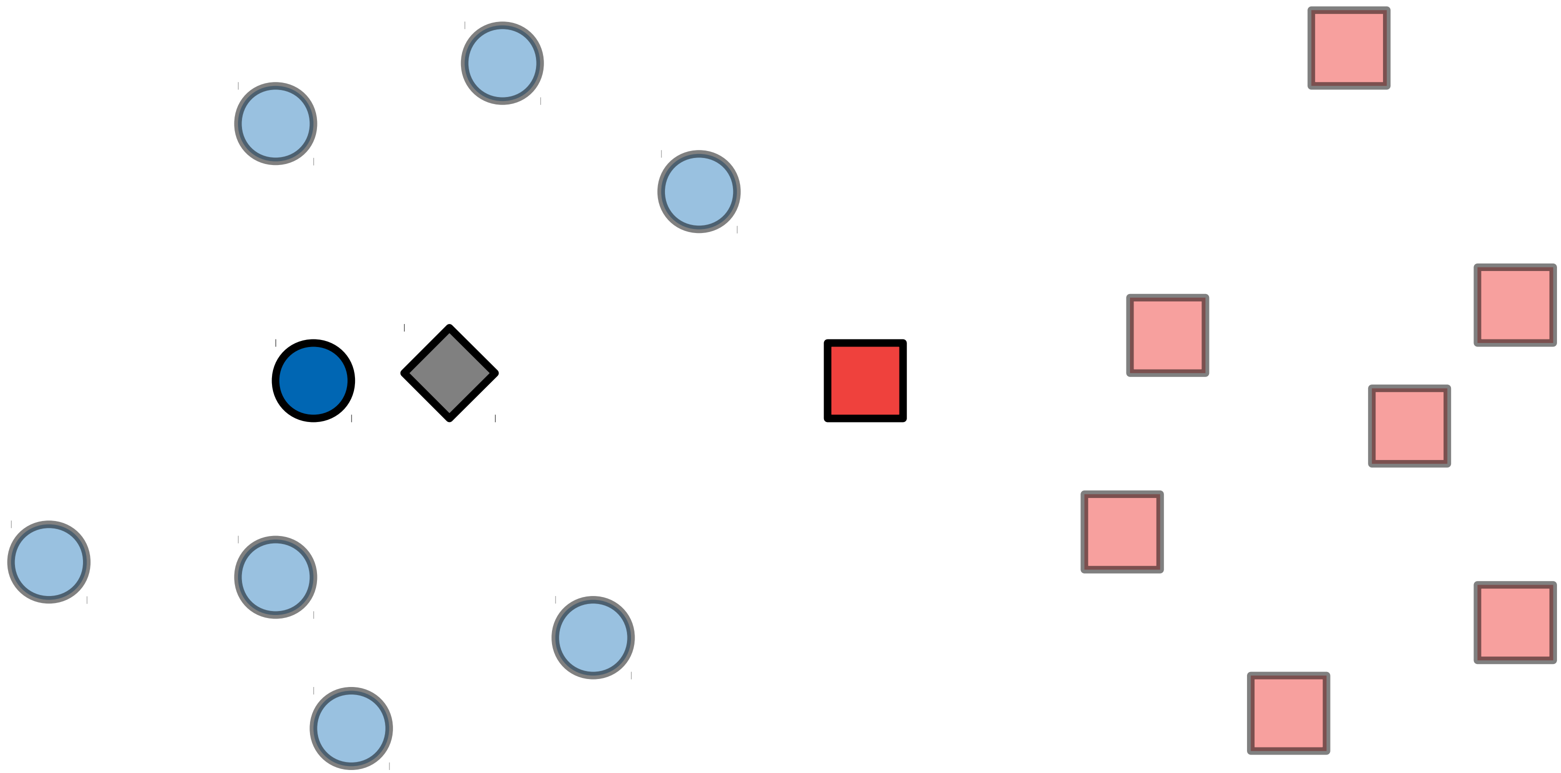
- In practice, HC and ANN use the radius to **correct** the distances from the testing object:
  - Better points become nearer to the testing objects than other points


Testing object (true class blue)
Nearest training object for the blue class
Nearest training object for the red class


Small radius: worst point
Large radius: better point



Better points → nearer
Worst points → farther



Better points → nearer
Worst points → farther

The idea behind ANN and HC

HC rule: $$d_{HC}(\mathbf{x}^{te}, \mathbf{x}_i^{tr}) = d(\mathbf{x}^{te}, \mathbf{x}_i^{tr}) - \mathrm{radius}(\mathbf{x}_i^{tr})$$

ANN rule: $$d_{ANN}(\mathbf{x}^{te}, \mathbf{x}_i^{tr}) = \frac{d(\mathbf{x}^{te}, \mathbf{x}_i^{tr})}{\mathrm{radius}(\mathbf{x}_i^{tr})}$$

The distance from points with large radius will be reduced more than that of points with a small radius (i.e. a training point with a larger radius will be preferred)

Relation between HC and ANN

- Recently it has been shown that the relation between HC and ANN is based on logarithms

> ANN is the HC rule applied to distances which have been **non linearly scaled** with a logarithm function

[Orozco-Alzate et al., ICIAP19]

Relation between HC and ANN

Non linear scaling: an alternative to linear scaling

Linear scaling

Z-score standardization

$$x^{new} = \frac{x - \mu_j}{\sigma_j}$$

Non linear scaling

Power $\quad x^{new} = x^{\rho} \quad (\rho > 0)$

Box-Cox $\quad x^{new} = \begin{cases} \frac{x^{\lambda}-1}{\lambda} & \text{if } \lambda \neq 0 \\ \log(x) & \text{otherwise} \end{cases}$

Logistic $\quad x^{new} = \frac{1}{1+e^{-x}}$

Logarithm $\quad x^{new} = \log(x)$

Relation between HC and ANN

- It has been shown that **non linear scaling of feature spaces** can be useful for classification
- Further, it has been shown that **non linear scaling of distances** can be useful for some distance-based classifiers

Carli et al ICCV2009W
Carli et al ICPR2010
Bicego et al Neurocomputing 2016
Duin et al S+SSPR2014
Orozco-Alzate et al S+SSPR2016

The PowerHC rule

- ANN rule is the HC rule applied to distances scaled with **logarithm**
- What about investigating **other non linear scalings**?
  - Logistic transformation
  - Power transformation

The PowerHC rule

- ANN rule is the HC rule applied to distances scaled with logarithm
- What about investigating other non linear scalings?
  - Logistic transformation
  - Power transformation

Best variant for non linear scaling of feature spaces

Almost never used for distances

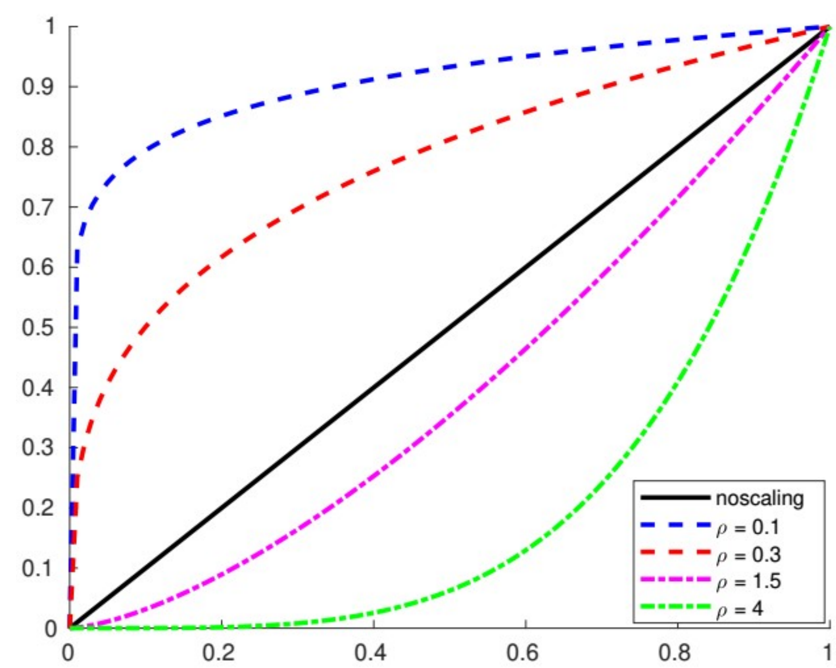
The Power-HC rule!

The powerHC rule

- The rule performs a non linear scaling of the distances using the **power transform** before applying the HC rule

$$\text{powerHC rule:} \quad d_{\rho HC}(\mathbf{x}^{te}, \mathbf{x}_i^{tr}) = d(\mathbf{x}^{te}, \mathbf{x}_i^{tr})^{\rho} - \text{radius}(\mathbf{x}_i^{tr})^{\rho}$$

In other words: distances are normalized via the **power transform**



All details in the paper!

Experimental evaluation

- We tested PowerHC using 24 standard UCI-ML datasets (of different dimensionality, cardinality and number of classes)

Moreover we tested PowerHC on a real world challenging problem (classification of **seismic-volcanic signals**)

We used DTW/Euclidean distances between spectrograms

Experimental evaluation

- We compared PowerHC with ANN and HC
  - PowerHC: the power ρ varies between 0.2 and 10 (step 0.2)
  - We report both accuracies for the best ρ and accuracies for automatically estimated ρ
- We used both NN and K-NN
- We evaluate statistical significance of differences with a statistical test

TABLE II

Accuracies, as percentages, along with standard errors for 50 repetitions and t-tests at 5% of significance for the compared methods when using NN for decision. Arrows point to the best method when differences are significant.

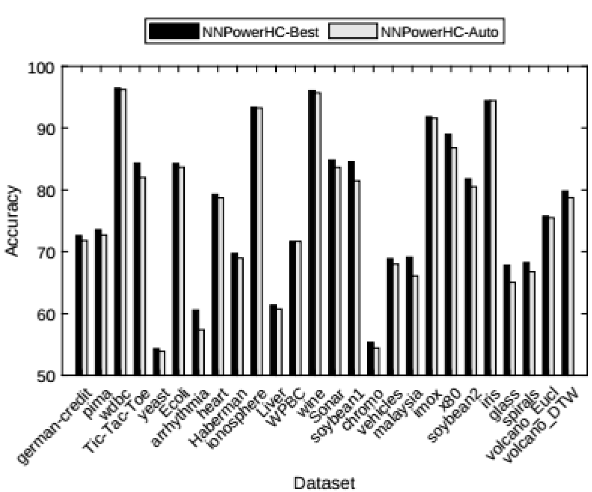
Dataset \ Method	A NN	B NN-ANN	C NN-HC	D NN-PowerHC	B vs D	C vs D
★ german-credit	68.72±0.29	71.32±0.29	71.58±0.29	72.59±0.28 ($\rho = 6.2$)	Reject ↗	Reject ↗
★ pima	69.70±0.33	72.43±0.32	72.70±0.32	73.54±0.32 ($\rho = 6.0$)	Reject ↗	Reject ↗
★ tic-tac-toe	79.52±0.26	80.86±0.25	83.04±0.24	84.36±0.23 ($\rho = 5.8$)	Reject ↗	Reject ↗
★ yeast	51.02±0.26					
★ arrhythmia	57.88±0.48					
★ heart	76.55±0.49					
★ haberman	66.32±0.54					
■ wdbc	95.06±0.18					
■ ecoli	81.79±0.42					
■ volcano_DTW	72.55±0.27					
■ glass	68.50±0.64					
▲ sonar	83.44±0.52					
▲ iris	93.33±0.41					
▲ liver	61.45±0.52					
▲ vehicles	69.11±0.32					
▲ malaysia	70.64±0.53					
◊ ionosphere	85.21±0.38					
◊ wpbc	65.59±0.68					
◊ wine	95.00±0.33					
◊ ch						
◊ vo						
◊ so						
◊ im						
◊ x8						
◊ so						
△ sp						

TABLE III

Accuracies, as percentages, along with standard errors for 50 repetitions and t-tests at 5% of significance for the compared methods when using KNN for decision. Arrows point to the best method when differences are significant.

Dataset \ Method	A KNN	B KNN-ANN	C KNN-HC	D KNN-PowerHC	B vs D	C vs D
★ german-credit	73.82±0.28	72.75±0.28	73.22±0.28	74.22±0.28 ($\rho = 8.8$)	Reject ↗	Reject ↗
★ tic-tac-toe	83.37±0.24	82.66±0.24	83.04±0.24	84.36±0.23 ($\rho = 5.8$)	Reject ↗	Reject ↗
★ arrhythmia	63.00±0.47	61.57±0.47	64.98±0.47	68.89±0.45 ($\rho = 1.8$)	Reject ↗	Reject ↗
★ haberman	75.03±0.49	74.64±0.50	74.76±0.50	75.26±0.49 ($\rho = 2.0$)	Reject ↗	Reject ↗
★ liver	63.74±0.52	62.48±0.52	64.15±0.52	65.08±0.51 ($\rho = 2.8$)	Reject ↗	Reject ↗
★ volcano_DTW	73.62±0.27	78.78±0.25	80.95±0.24	82.19±0.23 ($\rho = 2.2$)	Reject ↗	Reject ↗
▲ sonar	83.44±0.52	84.82±0.50	84.49±0.50	84.85±0.50 ($\rho = 0.2$)	Accept	Reject ↗
▲ malaysia	70.64±0.53	69.05±0.54	68.72±0.54	69.08±0.54 ($\rho = 0.2$)	Accept	Reject ↗
▲ iris	95.25±0.35	94.40±0.38	93.89±0.39	94.43±0.37 ($\rho = 0.2$)	Accept	Reject ↗
◊ wdbc	96.14±0.16	96.30±0.16	96.39±0.16	96.44±0.16 ($\rho = 1.6$)	Accept	Accept
			76.31±0.61	76.41±0.61 ($\rho = 8.2$)	Accept	Accept
			55.28±0.29	55.35±0.29 ($\rho = 0.8$)	Accept	Accept
			75.73±0.26	75.79±0.26 ($\rho = 3.0$)	Accept	Accept
			96.00±0.29	96.05±0.29 ($\rho = 0.6$)	Accept	Accept
			84.41±0.44	84.59±0.44 ($\rho = 1.6$)	Accept	Accept
			68.74±0.32	69.3±0.32 ($\rho = 9.8$)	Accept	Accept
			91.73±0.40	91.79±0.40 ($\rho = 0.8$)	Accept	Accept
			81.62±0.66	81.79±0.66 ($\rho = 1.2$)	Accept	Accept
			93.43±0.26	93.92±0.25 ($\rho = 0.2$)	↖ Reject	Reject ↗
			67.44±0.67	68.21±0.67 ($\rho = 0.2$)	↖ Reject	Reject ↗

All numbers are in the paper!



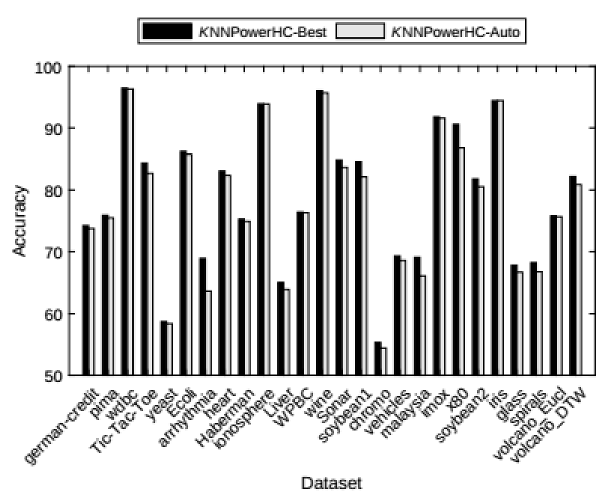


Fig. 2. Comparison of Optimal (Best) vs Automatic Approach for tuning ρ in: (left) NN-PowerHC and (right) KNN-PowerHC

Main findings

- Results with NN:
  - For a large group of datasets (16 over 26) PowerHC is better than ANN, HC (with a stastical significance)
  - In most of the other cases (9 over 26) there is not a statistically significant improvement
  - In 1 case PowerHC is outperformed by ANN

Main findings

- Similar results obtained with K-NN
- Automatic tuning of the parameter is satisfactory
- The best value for the parameter is always larger than 1 (concave transformation)
  - Different from what has been found for feature space!

Conclusions

- We proposed a novel variant of the Nearest Neighbor rule
  - Distances are non linearly scaled with the power tranformation before applying the HC rule
- Experiments show that non linear scaling are indeed useful
- To be investigated further: why concave transformation?

Thanks for your attention!