K-NN – METHOD SUMMARY

REGRESSION | CLASSIFICATION | NONPARAMETRIC | WHITE-BOX

General idea

- **similarity** in feature space (w.r.t. certain **distance metric** $d(\mathbf{x}^{(i)}, \mathbf{x})$) $\rightsquigarrow$ similarity in target space
- **Prediction** for **x**: construct k-**neighborhood** $N_k(\mathbf{x})$ from k points closest to **x** in $\mathcal{X}$, then predict
  - (weighted) mean target for **regression**: $\hat{y} = \frac{1}{\sum_{i:\mathbf{x}^{(i)} \in N_k(\mathbf{x})} w_i} \sum_{i:\mathbf{x}^{(i)} \in N_k(\mathbf{x})} w_i y^{(i)}$ with $w_i = \frac{1}{d(\mathbf{x}^{(i)}, \mathbf{x})}$ non weighted: $\hat{y} = \frac{1}{k} \sum y^{(i)}, w_i = 1 \; \forall$ obs. i

    $\rightarrow$ optional: higher weights w_i for close neighbors
  - most frequent class for **classification**: $\hat{y} = \arg\max_{\ell \in \{1,\dots,g\}} \sum_{i:\mathbf{x}^{(i)} \in N_k(\mathbf{x})} \mathbb{I}(y^{(i)} = \ell)$

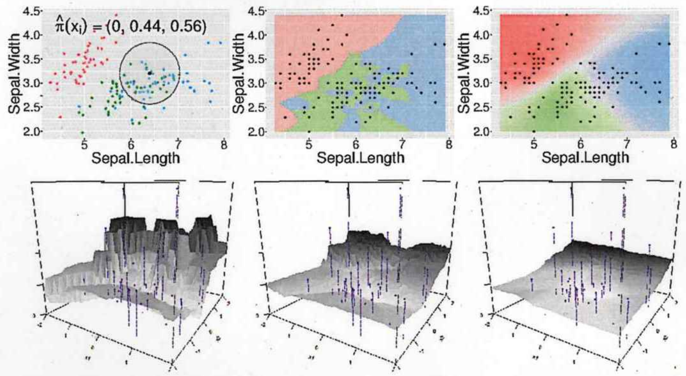
    $\Rightarrow$ Estimating posterior probabilities as $\hat{\pi}_\ell(\mathbf{x}^{(i)}) = \frac{1}{k} \sum_{i:\mathbf{x}^{(i)} \in N_k(\mathbf{x})} \mathbb{I}(y^{(i)} = \ell)$ $\nearrow$ $\hat{h}(x^{(i)}) = \arg\max_{\ell \in \{1,\dots,g\}} \hat{\pi}_\ell(x^{(i)})$
- **Nonparametric** behavior: parameters = training data; no compression of information
- Not immediately interpretable, but inspection of neighborhoods can be revealing

- Features should be standardized or normalized (e.g. between 0 and 1) so that they all "live" on same comparable scale

  Most distances place higher importance on feature with larger range scale $\Rightarrow$ imbalances, distortion of feature influence
- Domain knowledge may makes us manually up(down) weight certain features inside chosen distance metric

K-NN – METHOD SUMMARY

Hyperparameters Neighborhood **size** k (locality), **distance** metric (next page)



Classification
Left: Neighborhood for exemplary observation in iris, $k = 50$
Middle: Prediction surface for $k = 1$
Right: Prediction surface for $k = 50$

Regression
Left: Prediction surface for $k = 3$
Middle: Prediction surface for $k = 7$
Right: Prediction surface for $k = 15$

- Small $k \Rightarrow$ very local, "wiggly" decision boundaries tiny k is prone to overfitting (e.g. $k=1$ almost memorizes train data)!
- Large $k \Rightarrow$ rather global, smooth decision boundaries

K-NN – METHOD SUMMARY

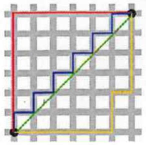
Popular distance metrics

$\hookrightarrow$ best set upfront as hyperparam.-tuning is inconvenient in kNN

\# insert (w_j) if we want importance-weighted features

- Numerical feature space:

  $\Rightarrow$ Typically, **Minkowski** distances $d(\mathbf{x}, \tilde{\mathbf{x}}) = \|\mathbf{x} - \tilde{\mathbf{x}}\|_q = \left(\sum_j^{\#} |x_j - \tilde{x}_j|^q\right)^{\frac{1}{q}}$
  - $q = 1$: **Manhattan** distance $\rightarrow d(\mathbf{x}, \tilde{\mathbf{x}}) = \sum_j^{\#} |x_j - \tilde{x}_j|$
  - $q = 2$: **Euclidean** distance $\rightarrow d(\mathbf{x}, \tilde{\mathbf{x}}) = \sqrt{\sum_j^{\#} (x_j - \tilde{x}_j)^2}$
  - Visualization: Manhattan (red, blue, yellow) vs. Euclidean (green)



- Mixed feature space:
  - **Gower distance** can handle numerical and categorical features, and missing data:

    \- numerical: $d(x_j, \tilde{x}_j) = \frac{|x_j - \tilde{x}_j|}{\max(x_j) - \min(x_j)}$ across j-th feature range(x_j) $\in [0,1]$

    \- categorical: $d(x_j, \tilde{x}_j) = \begin{cases} 1, \text{if } x_j \neq \tilde{x}_j \\ 0, \text{if } x_j = \tilde{x}_j \end{cases}$

    \- Gower distance as average over individual scores

    $d_{gower}(\mathbf{x}, \tilde{\mathbf{x}}) = \frac{\sum_{j=1}^{p} \delta_j \, d(x_j, \tilde{x}_j)}{\sum_{j=1}^{p} \delta_j}$ mixed features

    $\delta_j = \begin{cases} 0 \text{ if } j\text{-th feature has NA(s)} \text{ or if } x_j = \tilde{x}_j = 0 \text{ for binary feature (asymmetric, "1" more important vs. "0")} \\ 1, \text{ else} \end{cases}$
- Optional **weighting** to account for beliefs about varying feature importance

Figure Source: https://es.m.wikipedia.org/wiki/Archivo:Manhattan_distance.svg

K-NN – PROS & CONS

, Very simple learner that is rarely used in practice

Advantages non parametric model, lazy classifier

\+ Algorithm **easy** to explain and implement
\+ No distributional or functional **assumptions**
 $\rightarrow$ able to model data of **arbitrary complexity**
\+ No **training** or **optimization** required (lazy)
\+ **local model** $\rightarrow$ **nonlinear** decision boundaries
\+ Easy to **tune** (few hyperparameters)
 $\rightarrow$ number of neighbors k, distance metric
\+ **Custom** distance metrics can often be easily designed to incorporate domain knowledge

Disadvantages

– Sensitivity w.r.t. **noisy** or **irrelevant** features and outliers due to dependency on distance measure
– Heavily affected by **curse of dimensionality**
– Bad performance when feature **scales** are not consistent with feature relevance (standardize features!)
– Poor handling of data **imbalances** (worse for more global model, i.e., large k)