Chapter 7 k-Nearest Neighbors

Associate Professor
Yachai Limpiyakorn, Ph.D.

Classifier

ตัวเรียนรู้กระตือรือร้น (Eager Learner) ทำการประมาณค่าฟังก์ชันเป้าหมาย (target function) ณ ขณะเวลาเรียนรู้ด้วยชุดตัวอย่างสอน ค่าประมาณฟังก์ชันเป้าหมายที่ได้เป็นแบบ Global Approximation ซึ่งเป็นค่าประมาณที่ใช้จำแนกประเภทตัวอย่างทั้งหมดในปริภูมิตัวอย่าง รวมทั้งตัวอย่างค้นถามในอนาคต ได้แก่ Decision Tree, Naïve Bayes, Neural Networks

ตัวเรียนรู้เกียจคร้าน (Lazy Learner) จะรีรอไม่ดำเนินการเรียนรู้ด้วยชุดตัวอย่างสอน จนกว่าจะมีตัวอย่างใหม่ที่ต้องการจำแนกประเภทเข้ามา โดยจะสร้างค่าประมาณฟังก์ชันเป้าหมายแบบท้องถิ่น (local) ซึ่งค่าประมาณจะแตกต่างกันไปขึ้นอยู่กับค่าตัวอย่างละแวกใกล้เคียง (neighbors) กับตัวอย่างค้นถาม X_q (query instance)

ไม่สน model ในหัว

Instance-based/ Memory-based Learning

lazy learning

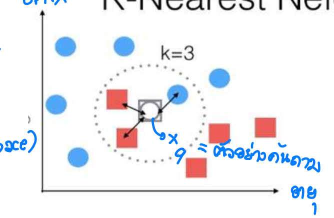
K - NN

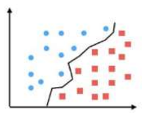
- memorizes the training observations for classifying the unseen test data, instead of performing explicit generalization.
- also termed as *lazy learning*, as it does not learn anything during the training phase like decision tree, neural networks, and so on
- Instead, it starts working only during the evaluation phase to compare the querying observations with nearest training observations, consuming significant time in comparison.
- the technique is thus not efficient on big data ไม่เหมาะกับ big data
- also, performance does deteriorate when the number of variables is high due to the *curse of dimensionality*. performance ด้อยลง ถ้า ตัวแปรมีเยอะ

  feature variable

Non-Parametric Classifiers: K-Nearest Neighbor

object
< $a_1, a_2, \ldots, a_n$ >
จุดในปริภูมิ n มิติ
(dimensional space)

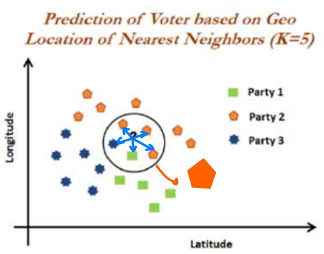



e.g., k=1

- Simple
- Non-Parametric means no assumption for underlying data distribution

k-NN Voter Example

predict the party for which voter will vote based on their neighborhood, precisely geolocation (latitude and longitude)



- The querying voter will vote for Party 2. As within the vicinity, one neighbor voted for Party 1 and the other voter voted for Party 3. But three voters voted for Party 2.
- kNN solves any given **classification** problem with *Majority vote*.
- While **regression** problems are solved by taking *mean* of its neighbors within the given circle or vicinity or k-value.

Two parts of Classification/ Regression issues:

- First, how to measure "close"
  - Minkowski distance หาค่าความใกล้ของเพื่อนบ้านด้วยระยะห่างระหว่างจุด 2 จุด

$$d(i,j)=\sqrt[q]{(|x_{i_1}-x_{j_1}|^q+|x_{i_2}-x_{j_2}|^q+\ldots+|x_{i_p}-x_{j_p}|^q)}$$

    - Manhattan distance (q=1)
    - Euclidean distance (q=2) is the most common one
- Second, how many close observations (K) neighbors

Circumventing Scaling Issues

- Similarity measured with Distance function can be dominated by those large-scaled attributes. ถูกครอบงำด้วย attribute ที่มีค่ามาก เป็น attribute ที่ชี้ขาดว่าเป็น class อะไร
- Data Standardization/ Normalization; for example, Min-Max Normalization, Scale each feature to zero mean and unit variance.

$$x_{i,\text{std}} = \frac{x_i - \mu_i}{\sigma_i}$$ Z - Score

Note: A scale is an ordered set of values, continuous or discrete, or a set of categories to which an attribute is mapped. Type of scale: Ratio numeric, Interval numeric, Nominal, Ordinal.

k-NN Algorithm

Training Algorithm:

- For each training example <x, f(x)>, add the example to the list *training_examples*

Classification Algorithm:

- Given a query instance x_q to be classified,
  - Let x_1 .... x_k denote k instances from *training_examples* nearest to x_q
  - CASE real-valued target function: RETURN

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$ กรณีเป็นตัวเลข

  - CASE discrete-valued target function: RETURN กรณี categorial

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\operatorname{argmax}} \sum_{i=1}^{k} \delta(v, f(x_i))$$

orange blue green
(3, 1, 1)

where $\delta(a,b) = 1$ if a=b and $\delta(a,b) = 0$ otherwise.
ถ้า == ถ้า ≠

K - fold = $\frac{\sum_{i=1}^{5} \text{Accuracy}_i}{5}$ X เพราะบาง fold อาจมีข้อมูลไม่เท่ากัน
แต่ถ้าข้อมูลเท่ากัน ใช้ได้

Weighted k-NN Algorithm (improvement โดยการถ่วงน้ำหนัก)

ใครใกล้เรา ได้น้ำหนักเยอะ

- Regression

1 เท่ากันหมดจะใช้สูตรนี้ได้ $\frac{\Sigma f(x_i)}{5}$

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}$$

→ summation weight

Intuitively, give more weight to the nearby points, and less weight to the farther away points. The simple function used is the inverse distance function.

$$w_i \equiv \frac{1}{d(x_q, x_i)^2}$$

แปรผกผัน

d น้อย w เยอะ

มีโอกาสเป็น 0 ถ้าพิกัดเดียวกัน

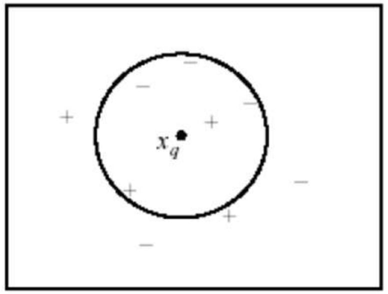
แต่ถ้าพิกัดเดียวกันแต่มีโอกาสต่าง class ได้ ต้อง handle programming เพิ่ม

- Classification

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\text{argmax}} \sum_{i=1}^{k} w_i \delta(v, f(x_i))$$

where $\delta(a,b) = 1$ if a=b and $\delta(a,b) = 0$ otherwise.

ประเด็นค่า k



K=1, $\hat{f}(x_q)$ = +

K=5, $\hat{f}(x_q)$ = −

Choosing k

- Goal is generalization: pick k (called a hyper-parameter) that performs best[1] on unseen (future) data.
- Split the dataset D:

Training set $\mathcal{D}_T$	Validation set $\mathcal{D}_V$

Learning set $\mathcal{D}_L$	Test set $\mathcal{D}_t$

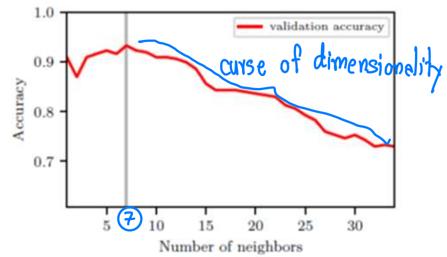
$\mathcal{D}$

Hyper-parameter tuning procedure

- Learn the model using the training set
- Evaluate performance with different k on the *validation set* picking the best k
- Report final performance on the test set.[2]

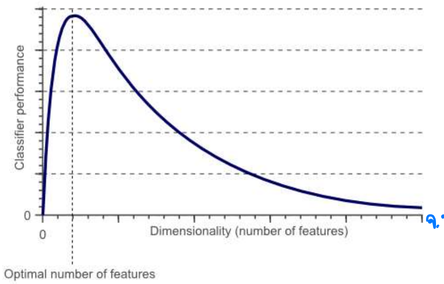
[1] In terms of some predefined metric, i.e. accuracy

Using validation set to choose k



We choose $k = 7$.

Curse of Dimensionality



จ.น. มิติข้อมูล

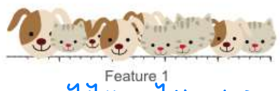
Figure 1. As the dimensionality increases, the classifier's performance increases until the optimal number of features is reached. Further increasing the dimensionality without increasing the number of training samples results in a decrease in classifier performance.

 https://www.visiondummy.com/2014/04/curse-dimensionality-affect-classification/

Curse of Dimensionality and Overfitting

- Given 10 training images of cats and dogs
- Start by a single feature (Fig.2)
- Add another feature (Fig.3)

เพิ่มเป็น 2 feature ไม่สามารถใช้ 1 เส้น ทำนายหมาแมวได้ เพราะมีจำนวน 2 มิติ



ทำไม่ได้ เพราะไม่มีจุดแบ่ง จำแนกได้

Figure 2. A single feature does not result in a perfect separation of our training data.

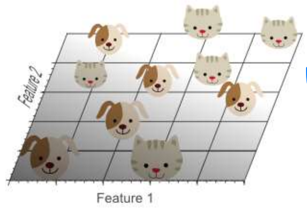


Figure 3. Adding a second feature still does not result in a linearly separable classification problem: No single line can separate all cats from all dogs in this example.

Curse of Dimensionality and Overfitting

- Add a third feature, yielding a three-dimensional feature space:

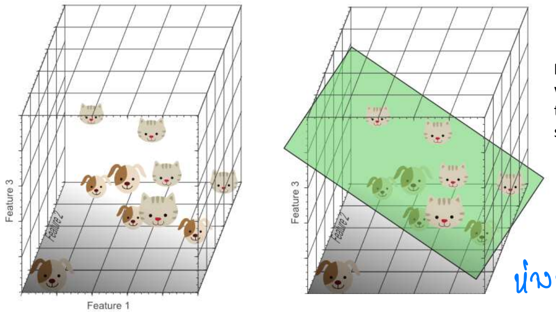


Figure 5. The more feature we use, the higher the likelihood that we can successfully separate the classes perfectly.

Figure 4. Adding a third feature results in a linearly separable classification. A plane exists that perfectly separates dogs from cats.

ห่างขึ้นๆ เบาบาง
sparse matrix ข้อมูลน้อย ในพื้นที่เยอะ
(attribute เยอะ)

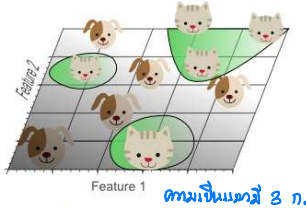
Curse of Dimensionality and Overfitting

- Keep adding features, the dimensionality of the feature space grows, and becomes sparser and sparser. ยิ่งเพิ่ม f ยิ่งเข้าใกล้ overfit
- In the 1D case (figure 2), 10 training instances covered the complete 1D feature space, the width of which was 5-unit intervals. Therefore, in the 1D case, the sample density was 10/5=2 samples/interval. ค.หนาแน่น 1 มิติ = 2 sample/interval (2 ตัวต่อ 1 ช่วง)
- In the 2D case however (figure 3), we still had 10 training instances, which now cover a 2D feature space with an area of 5×5= 25-unit squares. Here, the sample density was 10/25 = 0.4 samples/interval.
- Finally, in the 3D case, the 10 samples had to cover a feature space volume of 5x5x5= 125-unit cubes. Therefore, the sample density was 10/125 = 0.08 samples/interval. เพิ่มมิติ แต่ sample เท่าเดิม ค.หนาแน่นยิ่งน้อย
- N has to grow exponentially with the number of features.
- By using less features, the *curse of dimensionality* was avoided such that the classifier did not *overfit* the training data.

หลีกเลี่ยง curse of dimensionality โดยการลด จ.น. feature

Curse of Dimensionality and Overfitting

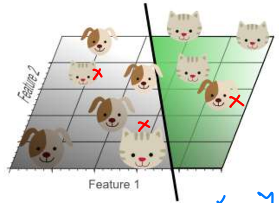
- Fig. 6 shows 3D classification results, projected onto a 2D feature space. Whereas the data was linearly separable in the 3D space, this is not the case in a lower dimensional feature space.



ความเป็นแมวมี 3 กฎ ซึ่ง specific เกินไป

Figure 6. Using too many features results in overfitting. The classifier starts learning exceptions that are specific to the training data (by coincidence) and do not generalize well when new data is encountered.

- Fig. 7 shows the result of a linear classifier that has been trained using only 2 features instead of 3.



แม้จะทายไม่ถูกเป๊ะ แต่ generalize กว่า รองรับ unseen data

Figure 7. Although the training data is not classified perfectly, this classifier achieves better results on unseen data than the one from figure 5.

Issues of kNN

- Distance measures → Normalization/ Standardization
- Expense of searching for nearest neighbors as we need to compare with the entire of training data stored in the repository
  - ➢Solution: use tree-based search structures, such as *k-d tree search* for efficient (approximate) NN on high-dimensional data
- Sensitive to Curse of Dimensionality
  - Remove irrelevant attributes ลบ features ไม่เกี่ยวข้อง
  - Suggest large datasets ใช้ข้อมูลเยอะ ๆ
  - Suggest the number of attributes not more than 20 $f \leq 20$

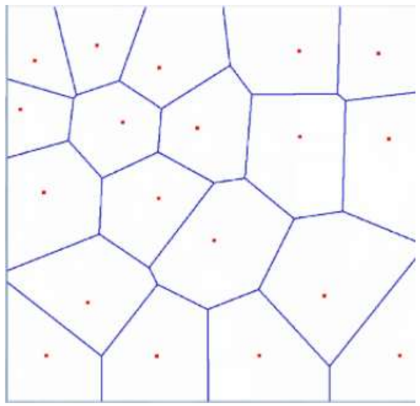
Pros & Cons & Constraints

ไม่มีผลกับ outlier เพราะอาศัยเพื่อนบ้าน ใกล้สุด

- *Pros*: high accuracy, insensitive to outlier (not using all points), no assumption about data distriution
- *Cons*: computationally expensive when classifying; susceptible to curse of dimensionality ช่องไหน
- *Constraints*: all features need to be standardized before fitting the model

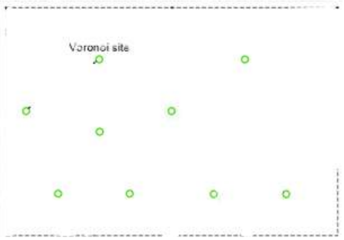
Voronoi Diagram [1]

- method decomposes a set of objects in a spatial space to a set of polygonal partitions.



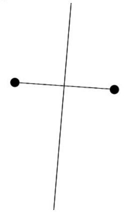
Voronoi Diagram 2

ถ้าเลือก region แล้วดูว่าตกพื้นที่ไหน
แสดงว่าเป็น class พื้นที่นั้น

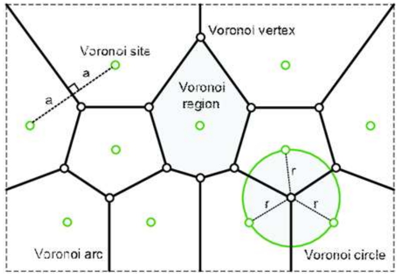


The points $p_1, \ldots, p_n$ are called **Voronoi sites**. The Voronoi diagram for two sites p_i and p_j can be easily constructed by drawing the perpendicular bisector of line segment $\overline{p_i p_j}$.

ตั้งฉาก



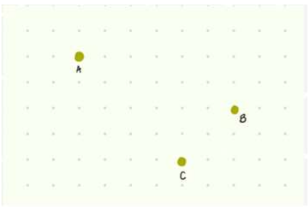
Voronoi Diagram 3



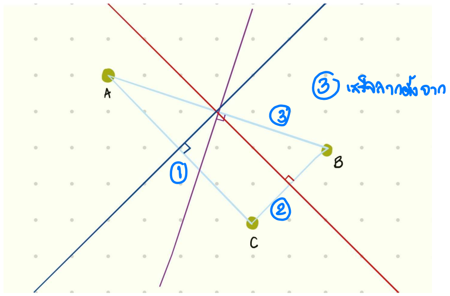
- Given a set of S points $p_1, \ldots, p_n$ in the plane, a Voronoi diagram divides the plane into n Voronoi regions with the following properties:
- Each point pi lies in exactly one region.
- If a point $q \notin S$ lies in the same region as p_i, then the Euclidian distance from p_i to q will be shorter than the Euclidian distance from p_j to q, where p_j is any other point in S.

Example

- Given Voronoi sites, p_i =A, B, C



- Half plane , $H(p_i p_j)$



Example cont.

- Voronoi regions, V(pi)

$V(p_i) = H(p_i p_1) \cap H(p_i p_2) \cap \ldots \cap H(p_i p_n)$

- **Voronoi edges** = line segments form the boundaries of Voronoi regions
- **Voronoi vertice** = intersection of adjacent edges.