Lecture 8 (k-NN, Decision Tree)

k-nearest neighbor

- *k*-Nearest Neighbors (*k*-NN) is an intuitive, simple, yet powerful **non-parametric** and **instance-based** learning algorithm widely used for both **classification** and **regression** tasks.
- Unlike many supervised learning algorithms that require training a model, *k*-NN stores the entire training dataset and makes predictions for new data points by comparing them directly to the stored instances. It is a **lazy learning algorithm**, meaning no explicit training occurs, and the predictions are made based on the proximity of new instances to existing ones.

k-nearest neighbor

- Given a query point x_q, the *k*-NN algorithm identifies the **k nearest neighbors** in the training set based on a distance metric:

Euclidean Distance

$$d(\boldsymbol{x}_q, \boldsymbol{x}_i) = \sqrt{\sum_{j=1}^{n} (x_{qj} - x_{ij})^2}$$

Manhattan Distance

$$d(\boldsymbol{x}_q, \boldsymbol{x}_i) = \sum_{j=1}^{n} |x_{qj} - x_{ij}|$$

k-nearest neighbor - Classification

- In classification tasks, the algorithm assigns a class label to the query point x_q based on the most frequent label (the mode) among its *k*-nearest neighbors.
- Once the *k*-nearest neighbors are identified, the class labels of the neighbors are collected, and the query point is classified by majority vote:

$$\hat{y}_q = mode([y_1, y_2, \cdots, y_k])$$

where $y_1, y_2, \cdots, y_k$ are the class labels of the *k*-nearest neighbors.

- **Example 1:**
- For a query point with neighbors' class labels [0, 1, 1, 0, 1], the **mode** is 1, so the predicted class is 1.
- **Example 2:**
- For a query point with neighbors' class labels [0, 1, 1, 0], the **mode** is ?. There is a **tie**

k-nearest neighbor - Classification

- In cases where multiple classes have the same frequency among the neighbors, a **tie** may occur. Possible strategies to handle ties include:
- **Random selection**: Randomly select one of the tied classes.
- **Preference to closest neighbor**: Choose the class label of the nearest neighbor in case of a tie.
- **Weighted voting**: Apply weights based on the distance of each neighbor, giving closer neighbors more influence in the decision.

Approach	**Advantages**	**Disadvantages**
Random Selection	Simple, neutral bias	Inconsistent, lacks interpretability
Nearest Neighbor Preference	Consistent, clear logic, interpretable	Sensitive to noise and scaling issues
Weighted Voting	Reliable, less noisy, reduces impact of ties	Extra computational cost, sensitive to weight choice

k-nearest neighbor - Regression

- In regression tasks, *k*-NN predicts the output based on the average of the target values of the *k*-nearest neighbors.
- For a query point x_q, the predicted value is the mean of the target values y_i of the nearest neighbors:

$$\hat{y}_q = \frac{1}{k}\sum_{i=1}^{k} y_i$$

- To improve performance, especially when some neighbors are much closer than others, weighted *k*-NN can be used. Here, each neighbor's influence is weighted by its distance from the query point:

$$\hat{y}_q = \frac{\sum_{i=1}^{k} w_i y_i}{\sum_{i=1}^{k} w_i}$$

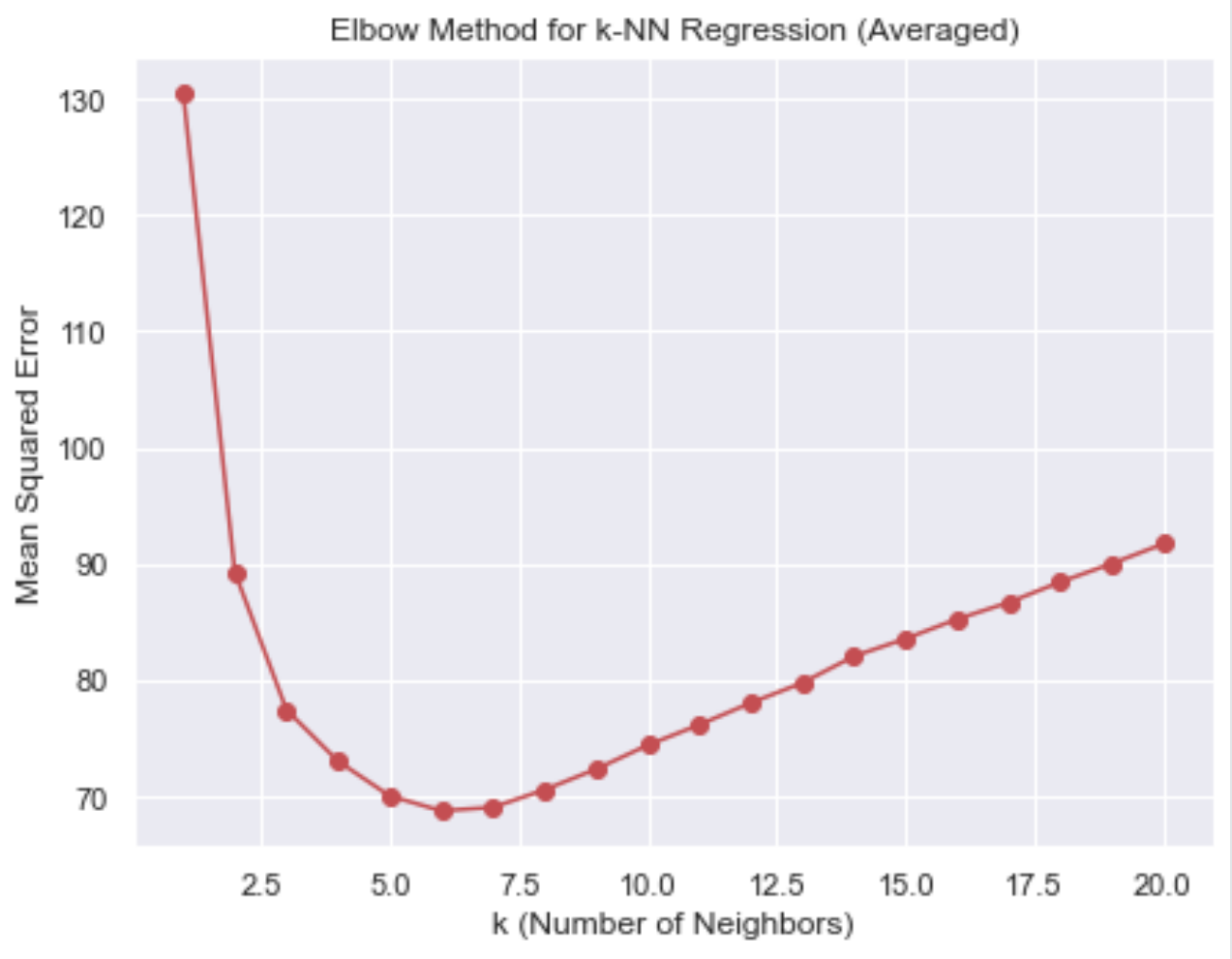
→ normalizes weights

where $w_i = \frac{1}{d(x_q, x_i)}$ is the weight assigned to neighbor *i* based on its distance to the query point.

k-nearest neighbor

- Selecting an appropriate value of *k* is crucial for *k*-NN's performance. Small values of *k* can lead to **overfitting**, while large values can cause **underfitting**.
- One common approach to choose *k* is the **elbow method**. This method involves plotting the error (classification error rate or mean squared error for regression) as a function of *k* and identifying the point where the error stops decreasing significantly (the "elbow" point).

k-nearest neighbor



Experiment 5

Limitations *k*-nearest neighbor

- **Curse of Dimensionality**: As the number of features increases, the distance between data points becomes less meaningful. In high-dimensional spaces, points tend to become equidistant from each other, reducing the effectiveness of distance-based methods like k-NN.

  Solution: *Dimensionality reduction techniques like* ***Principal Component Analysis (PCA)*** *or* ***Linear Discriminant Analysis (LDA)*** *can help reduce the number of features, improving the meaningfulness of distance computations.*

- **Computational Complexity**: For each query, *k*-NN requires computing the distance between the query point and all training points. This leads to a time complexity of $O(N \cdot n)$, where N is the number of training points and n is the number of features.

  Solution: *Data structures such as* ***KD-trees*** *and* ***ball trees*** *can reduce the computational burden, especially in low-dimensional data.*

- **Imbalance in Class Distribution**: When one class dominates the dataset, the nearest neighbors of any query point may frequently belong to the majority class, leading to biased predictions.

  Solution: *Use* ***distance-weighted voting***, *where closer neighbors are given more importance in the decision-making process, reducing bias toward majority classes.*

Decision Tree

- A decision tree is a simple model for supervised classification. It is used for classifying a single discrete target feature.
- Each internal node performs a Boolean test on an input feature (in general, a test may have more than two options, but these can be converted to a series of Boolean tests). The edges are labeled with the values of that input feature.
- Each leaf node specifies a value for the target feature.

Content adapted from Alice Gao

Decision Tree – Example 1 (All examples belong to the same class)

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Content adapted from Alice Gao

Decision Tree – Example 2 (No features left)

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No
15	Sunny	Mild	High	Weak	No
16	Sunny	Mild	High	Weak	Yes
17	Sunny	Mild	High	Strong	Yes

these all end up at one leaf node w/ no more features to split
↳ majority voting
↳ prob. distribution

Content adapted from Alice Gao

Decision Tree – Example 3 (No examples left)

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No
15	Sunny	Hot	High	Weak	No

Content adapted from Alice Gao

Decision Tree

Which feature we use at each step?

- Ideally, we would like to find the optimal order of testing features, which will minimize the size of our tree. Unfortunately, finding the optimal order is too expensive computationally. Instead, we will use a greedy approach.
- The greedy approach will make the best choice at each step without worrying about how our current choice could affect the potential choices in the future. More concretely, at each step, we will choose a feature that makes the biggest difference to the classification, or a feature that helps us make a decision as quickly as possible.

Content adapted from Alice Gao

Decision Tree

- Feature that reduces our uncertainty at much as possible will be chosen.
- To measure the reduction in uncertainty, we will calculate the uncertainty in the examples before testing the feature, and subtract the uncertainty in the examples after testing the feature. The difference measures the information content of the feature. Intuitively, testing the feature allows us to reduce our uncertainty and gain some useful information. We will select the feature that has the highest information content.
- How do we measure uncertainty?

Content adapted from Alice Gao

Decision Tree

entropy at a node

$$I\big(P(c_1),\cdots,P(c_k)\big) = -\sum_{i=1}^{k} P(c_i)\log_2(P(c_i))$$

- What is the entropy of the distribution (0.5,0.5)?

$$-0.5\log_2(0.5) - 0.5\log_2(0.5) = 1$$

This distribution has 1 bit of uncertainty.

- What is the entropy of the distribution (0.01,0.99)?

$$-0.01\log_2(0.01) - 0.99\log_2(0.99) = 0.08$$

This distribution has 0.08 bit of uncertainty.

- The entropy is maximized in the case of a uniform distribution
- Information gain will be the metric to be used to determine the feature to be used (ID3).

$$\text{InfoGain} = \text{I}_{\text{before}} - \text{I}_{\text{after}} = \text{I}_{\text{before}} - \sum_{i=1}^{k} \frac{p_i + n_i}{p+n} * I\left(\frac{p_i}{p+n}, \frac{n_i}{p+n}\right)$$

Content adapted from Alice Gao

Decision Tree – Example 4

There are 14 examples, 9 positive and 5 negative

What is the entropy of the examples before we select a feature for the root node of the tree?

$$I\left(\frac{9}{14},\frac{5}{14}\right) = -\left(\frac{9}{14}\log_2\frac{9}{14}+\frac{5}{14}\log_2\frac{5}{14}\right) \approx 0.94$$

What is the expected information gain if we select Outlook as the root node of the tree?

$$Outlook = \begin{cases} Sunny; 2\,Yes, 3\,No\,; 5\,Total \\ Overcast; 4\,Yes, 0\,No\,; 4\,Total \\ Rain; 4\,Yes, 2\,No\,; 5\,Total \end{cases}$$

$$Gain(Outlook) = 0.94 - \left(\frac{5}{14}.I\left(\frac{2}{5},\frac{3}{5}\right)+\frac{4}{14}.I\left(\frac{4}{4},\frac{0}{4}\right)+\frac{5}{14}I\left(\frac{3}{5},\frac{2}{5}\right)\right)$$

$$= 0.94 - \left(\frac{5}{14}(0.971)+\frac{4}{14}(0)+\frac{5}{14}(0.971)\right) = 0.94 - 0.694 = 0.247$$

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Content adapted from Alice Gao

Decision Tree – Example 4

What is the expected information gain if we select Humidity as the root node of the tree?

$$Humidity = \begin{cases} Normal\ ; 6\ Yes\ 1\ No; 7\ Total \\ High; 3\ Yes\ 4\ No; 7\ Total \end{cases}$$

$$Gain(Humidity) = 0.94 - \left(\frac{7}{14}.I\left(\frac{6}{7},\frac{1}{7}\right) + \frac{7}{14}.I\left(\frac{3}{7},\frac{4}{7}\right)\right)$$

$$= 0.94 - \left(\frac{7}{14}(0.592) + \frac{7}{14}(0.985)\right) = 0.94 - 0.789 = 0.151$$

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Content adapted from Alice Gao

Decision Tree – Example 4

What is the expected information gain if we select Wind as the root node of the tree?

$$Wind = \begin{cases} Weak\ ; 6\ Yes\ 2\ No; 8\ Total \\ Strong; 3\ Yes\ 3\ No; 6\ Total \end{cases}$$

$$Gain(Wind) = 0.94 - \left(\frac{8}{14}.I\left(\frac{6}{8},\frac{2}{8}\right) + \frac{6}{14}.I\left(\frac{3}{6},\frac{3}{6}\right)\right)$$

$$= 0.94 - \left(\frac{8}{14}(0.81) + \frac{6}{14}(1)\right) = 0.94 - 0.891 = 0.0485$$

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Content adapted from Alice Gao

Decision Tree – Example 4

What is the expected information gain if we select Temperature as the root node of the tree?

$Gain(Temperature) = 0.029$

Day	Outlook	Temp	Humidity	Wind	Tennis?
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cool	Normal	Weak	Yes
6	Rain	Cool	Normal	Strong	No
7	Overcast	Cool	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cool	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

Content adapted from Alice Gao

Decision Tree

- Information gain is not the only splitting criterion to be used in decision tree.
- The next criterion is the gini index (CART)

$$Gini(t) = 1 - \sum_{i=1}^{C} p_i^2$$

where

C is the number of classes

p_i is the proportion of instances belonging to class *i* at particular node t.

- Gini index is computationally efficient
- Information gain will be useful in the case of imbalanced datasets

Decision Tree

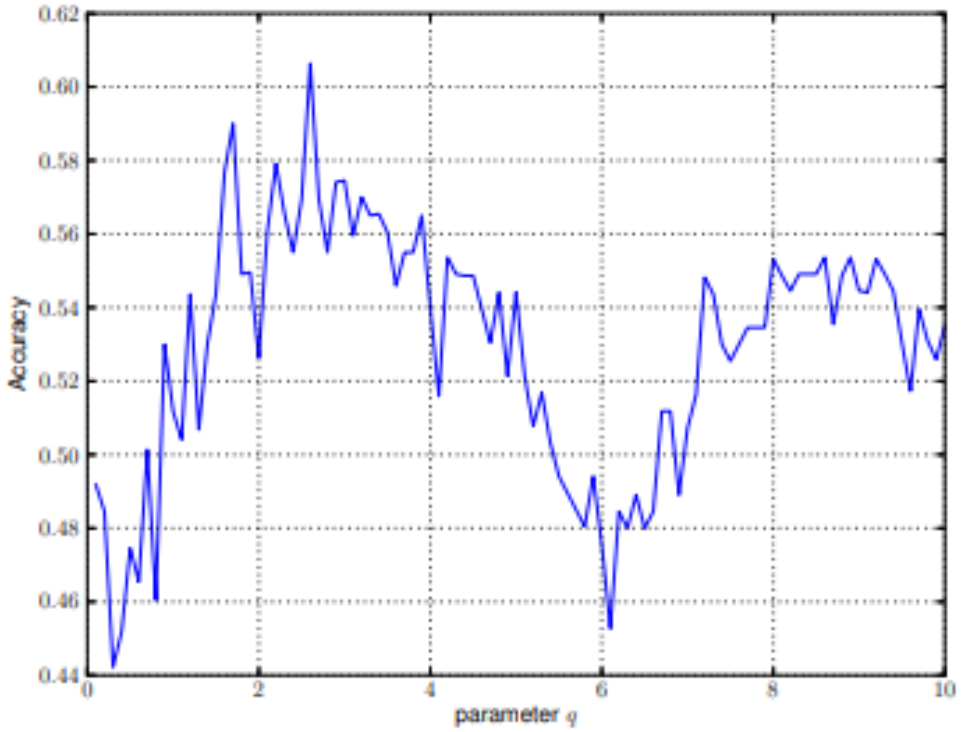
- The gini coefficient and the information gain (entropy) are different splitting criteria but are unified under what is known as the **Tsallis Entropy**.

$$S_q(X) = \frac{1}{1-q}\left(\sum_{i=1}^{n} p_i^q - 1\right), q \in \mathbb{R}$$

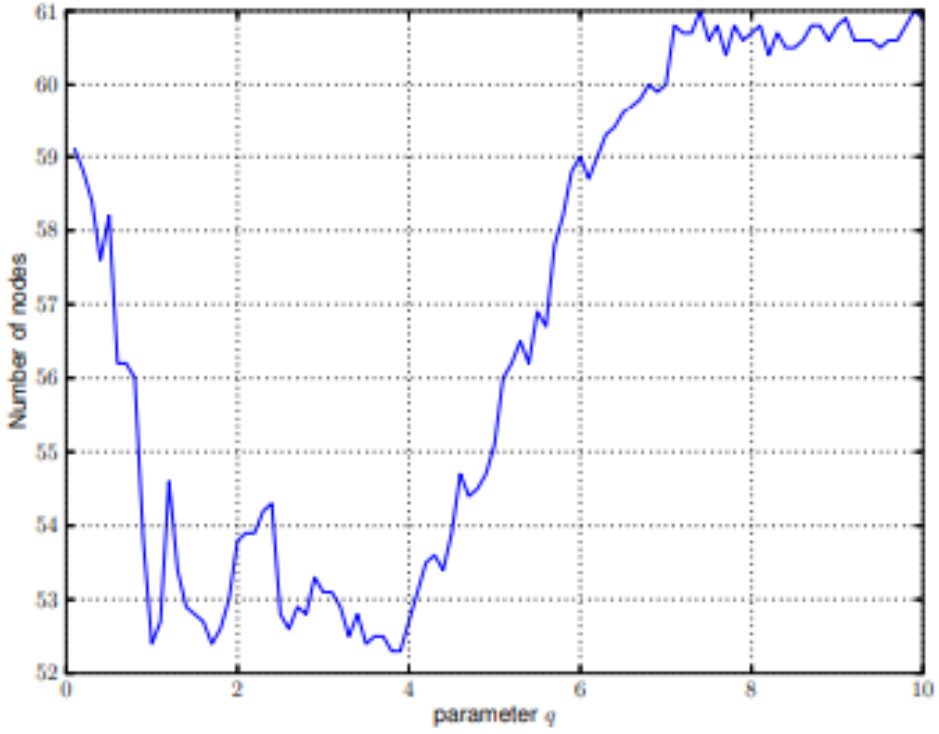
$$\lim_{q \to 1} S_q(X) = H(X)$$

$$S_2(X) = 1 - \sum_{i=1}^{n} p_i^2 = Gini\ Index$$

Decision Tree



(a) Accuracy with different values of q



(b) Tree complexity with different values of q

Wang, Yisen, Chaobing Song, and Shu-Tao Xia. "Unifying the Split Criteria of Decision Trees Using Tsallis Entropy." *arXiv preprint arXiv:1511.08136* (2016).

Decision Tree

Data Set	Shannon entropy (ID3)		Gini index (CART)		Tsallis entropy (TEC)		
	Accuracy (%)	No. of nodes	Accuracy (%)	No. of nodes	Accuracy (%)	No. of nodes	q
Yeast	52.8	199	51.8	196.6	**56.9**	**195.8**	1.4
Glass	51.2	52.4	52.6	53.8	**60.6**	52.6	2.6
Vehicle	71.7	103	70.2	**100**	**73.8**	111.0	0.6
Wine	92.9	12.0	90.0	12.0	**95.9**	9.6	3.1
Haberman	70.3	32.2	70.3	33.0	74.2	33.2	7.1
Car	98.2	106.4	98.1	106.8	98.3	**106.2**	0.8
Scale	75.9	97.6	76.1	97.2	78.2	93.1	3.1
Hayes	81.5	28.8	80.0	25.3	**82.3**	19.5	8.6
Monks	51.9	89.0	52.1	88.6	**57.3**	89.6	8.9
Abalone	25.4	89.2	25.0	85.8	**26.8**	86.2	0.8
Cmc	49.1	267.0	47.4	264.0	**52.0**	264.2	1.2

Wang, Yisen, Chaobing Song, and Shu-Tao Xia. "Unifying the Split Criteria of Decision Trees Using Tsallis Entropy." *arXiv preprint arXiv:1511.08136* (2016).

Choosing the optimal value of q is still an open question

Decision Tree

- It would be better to grow a smaller and shallower tree. The smaller and shallower tree may not predict all of the training data points perfectly but it may generalize to test data better.
- We have two options to prevent over-fitting when learning a decision tree
- Pre-pruning: stop growing the tree early
- Post-pruning: grow a full tree first and then trim it afterwards.

Content adapted from Alice Gao

Decision Tree

Pre-pruning

- If we decide not to split the examples at a node and stop growing the tree there, we may still have examples with different labels. At this point, we can decide to use the majority label as the decision for that leaf node. Here are some criteria we can use:
- **Maximum depth:** We can decide not to split the examples if the depth of that node has reached a maximum value that we decided beforehand.
- **Minimum number of examples at the leaf node:** We can decide not to split the examples if the number of examples remaining at that node is less than a predefined threshold value.
- **Minimum information gain:** We can decide not to split the examples if the benefit of splitting at that node is not large enough. We can measure the benefit by calculating the expected information gain. In other words, do not split examples if the expected information gain is less than the threshold.
- **Reduction in training error:** We can decide not to split the examples at a node if the reduction in training error is less than a predefined threshold value.

Content adapted from Alice Gao

Decision Tree

- Post-pruning is particularly useful when any individual feature is not informative, but multiple features working together is very informative.
- **Example:** Suppose we are considering post-pruning with the minimal information gain metric.

  First of all, we will restrict our attention to nodes that only have leaf nodes as its descendants. At a node like this, if the expected information gain is less than a predefined threshold value, we will delete this node's children which are all leaf nodes and then convert this node to a leaf node.

  There has to be examples with different labels at this node possibly both positive and negative examples. We can make a majority decision at this node.

Content adapted from Alice Gao