Supervised Learning

Week 8

KNN

Decision Tree (DT)

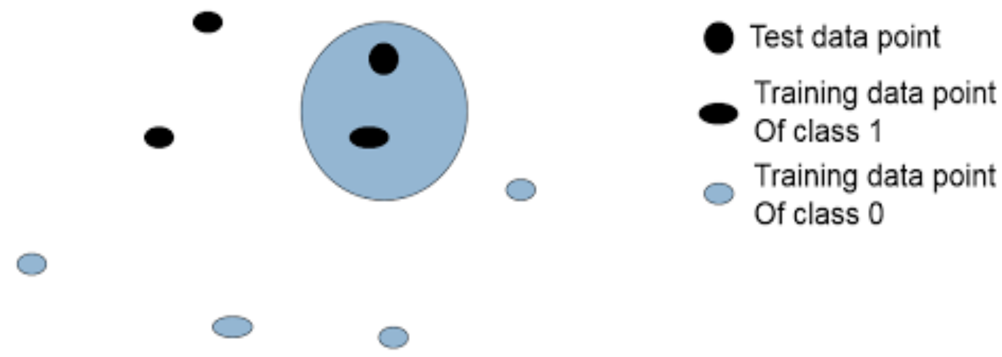
KNN

Algorithm & Variants

Best K

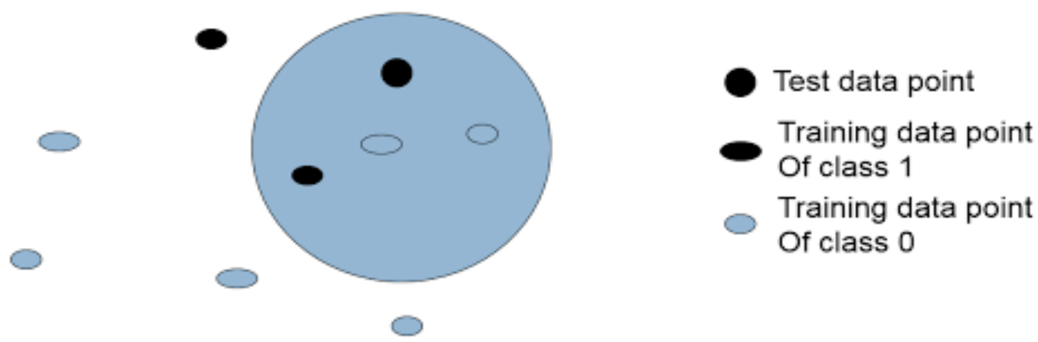
KNN
Algorithm & Variants

- For both classification and regression
  - A useful technique is to assign weights based on the neighbours.
  - The nearer neighbors contribute more to the average than the distant ones.
- The basic idea is to label the test data point same as the nearest neighbor.



KNN
Algorithm & Variants...

- How many neighbors?
  - K=?



- Let's say someone would like to check K nearest neighbours of the test point to make the decision.
  - Label a test instance same as the majority label of the K-nearest neighbours.
- The figure is an example a of 3−NN classification.

KNN
Algorithm & Variants...

- How to make the majority decisions?
  - Mode of the class labels
    - Discrete cases
  - Average or mean distances
    - Continuous cases
  - Distance-weighted nearest neighbour algorithm (Shepard's method)
    - Assign weights to the neighbours based on their distance from the test point.
    - Weight may be inverse square of the distances (*$1/D^2$*)
      - Higher the distance of the neighbour, lower its weight.

KNN
Best number of neighbors (K)

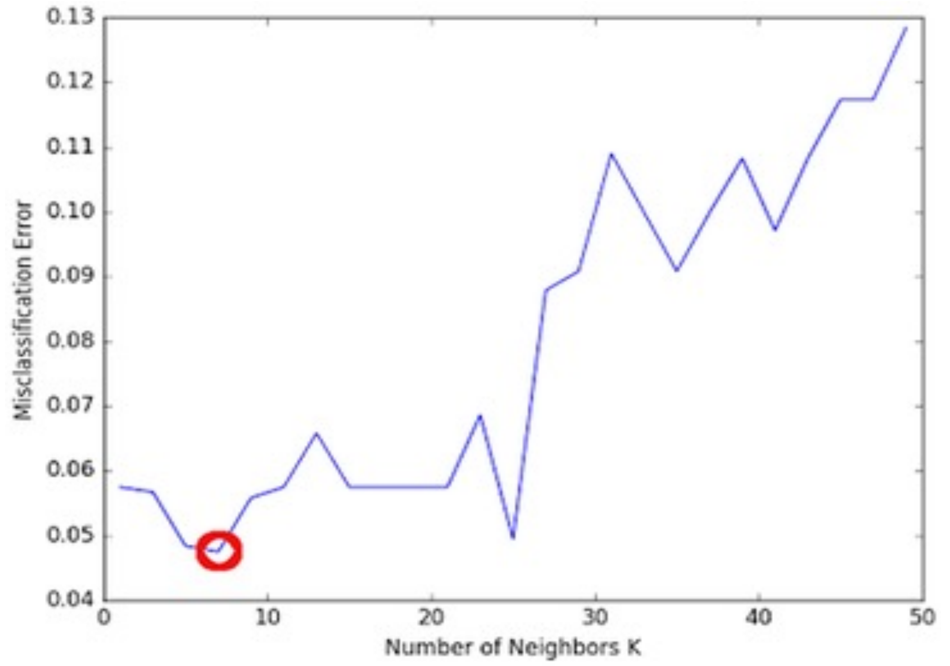
- What is the importance of the variable K?
  - K controls the shape of the decision boundary
- Small values of K
  - Restrains the region of a given prediction
  - Forces classifier to be more focused on the close regions and neighbours
    - This will result in a low bias and high variance
- Higher values of K
  - Asking for more information from distant training points
  - Smoother decision boundaries
    - Lower variance but increases bias

KNN
Best number of neighbors (K)...

- Finding the best K
  - There is no rule of thumb in selecting K_{max} since it depends on your desired rate of exploration for K
  - A simple and handy method
    - Cross-validation to partition your data into test and training samples
    - Evaluate model with different ranges of K values

$$K = 1, .., K_{max}$$

    - The misclassification error can be used as a measurement of performance



Remarks

- Learning is very simple (actually, no learning involved).

- Classification is very time consuming because we need to find distance with all the training instances.

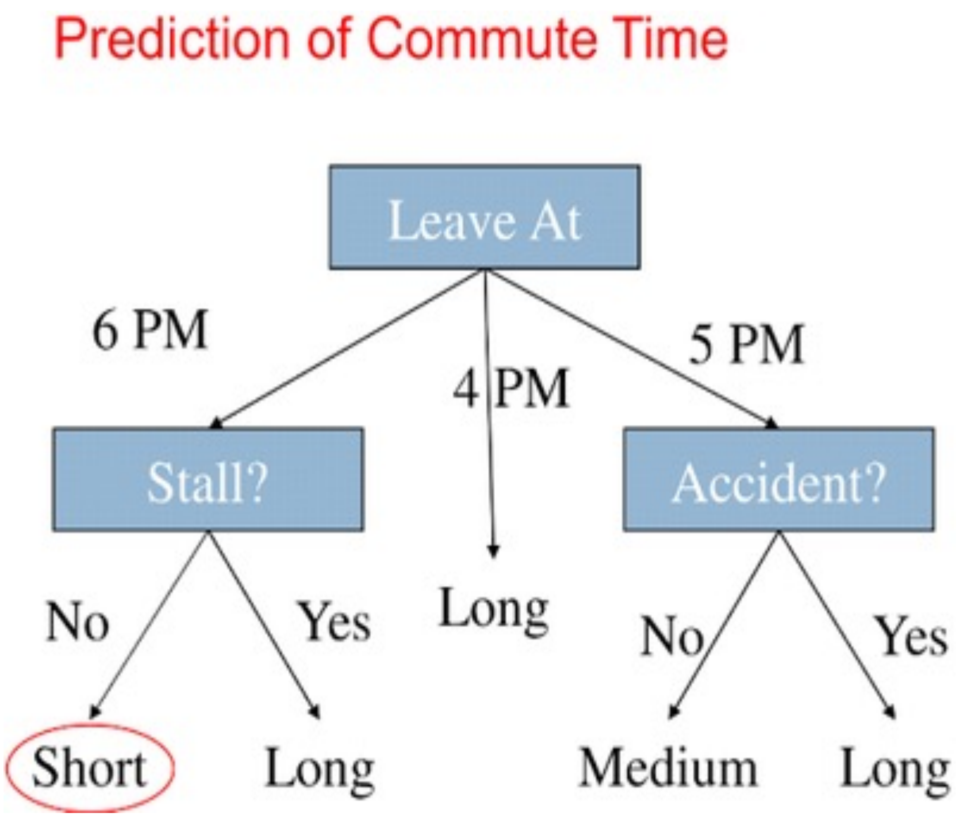
Decision Tree (DT)

Regression/classification trees

Decision tree algorithms

Model complexity & pruning

Decision Trees



If we leave at 6 PM and there are no cars stalled on the road, what will our commute time be?

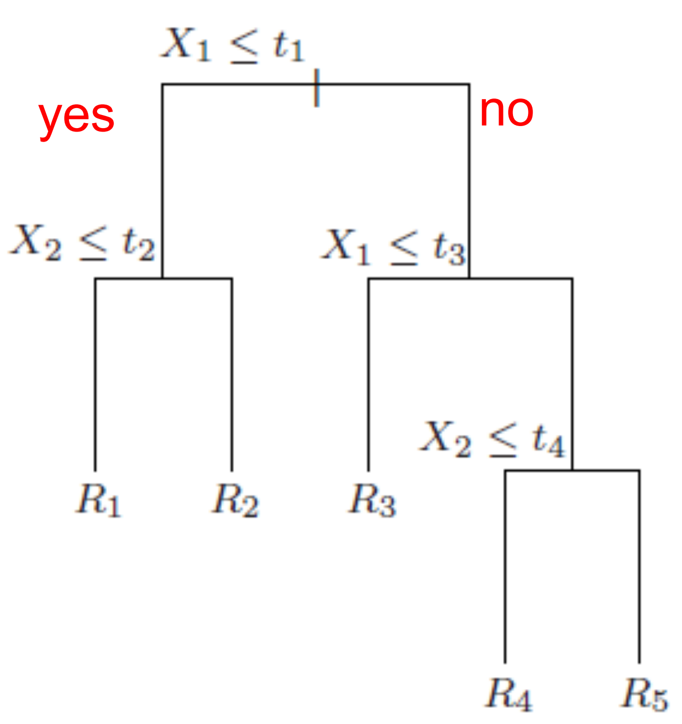
Decision Trees

- A decision tree is a map of the possible outcomes of a series of related options.
- It weighs possible actions against one another
  - Costs, probabilities, and benefits
- Typically starts with a single node
  - Branches into possible outcomes.

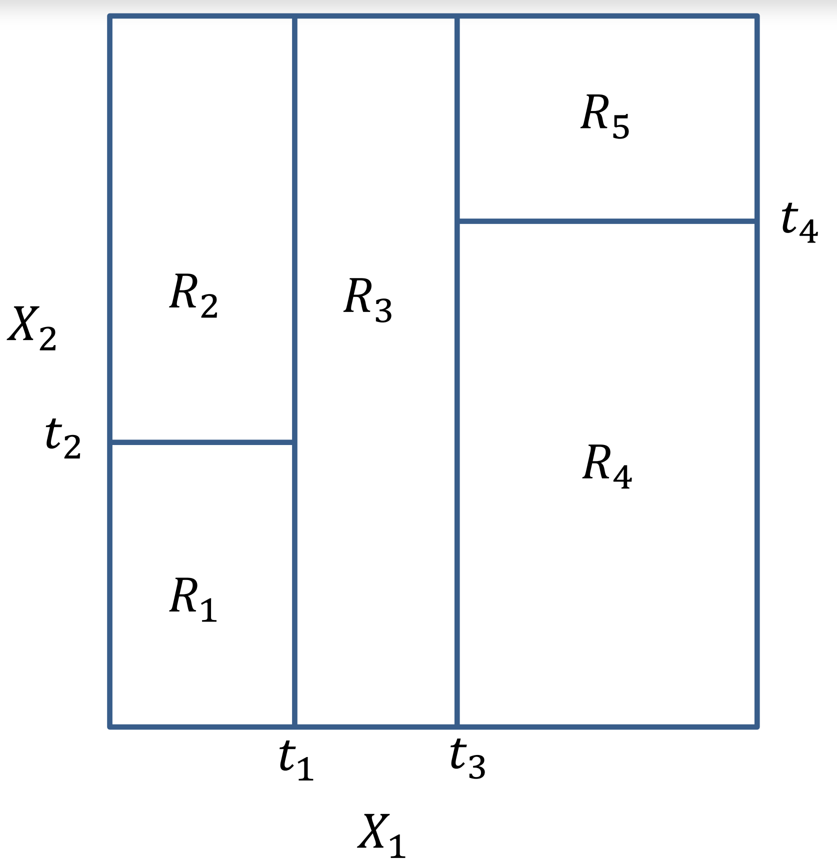
- Simple and easy to interpret!

- May not be competitive with the best supervised learning algorithms in terms of prediction accuracy.

Partition of Feature Space



[Figure taken from Hastie's ESL book]



Decision Trees

- After partitioning the feature space, we can fit a simple model in each sub-region (R_1, R_2 …).

- We can fit a regression model. Such decision trees are called regression trees.

- We can also fit a classification model. Such decision trees are called classification trees.

- Usually, extremely simple models such as majority (classification) or mean (regression) are used.

Process of building a DT

- Let's start with the procedure:
  - We divide the feature space, i.e., the set of possible values for $x_1, \ldots, x_d$ into J distinct and non-overlapping regions, $R_1, \ldots, R_\mathrm{J}$
  - For every instance that falls into region R_j, we make the same prediction, which is simply the mean (or mode) of response values for the training observations in R_j.

Formulation of Regression Trees

- The overall goal of regression trees is to find regions $R_1, R_2, \dots, R_J$ that minimize the training error:

$$\sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

where $\hat{y}_{R_j}$ is the mean of the target values of the training instances in the j^{th} region.

But how do we exactly perform these actions?

Solution

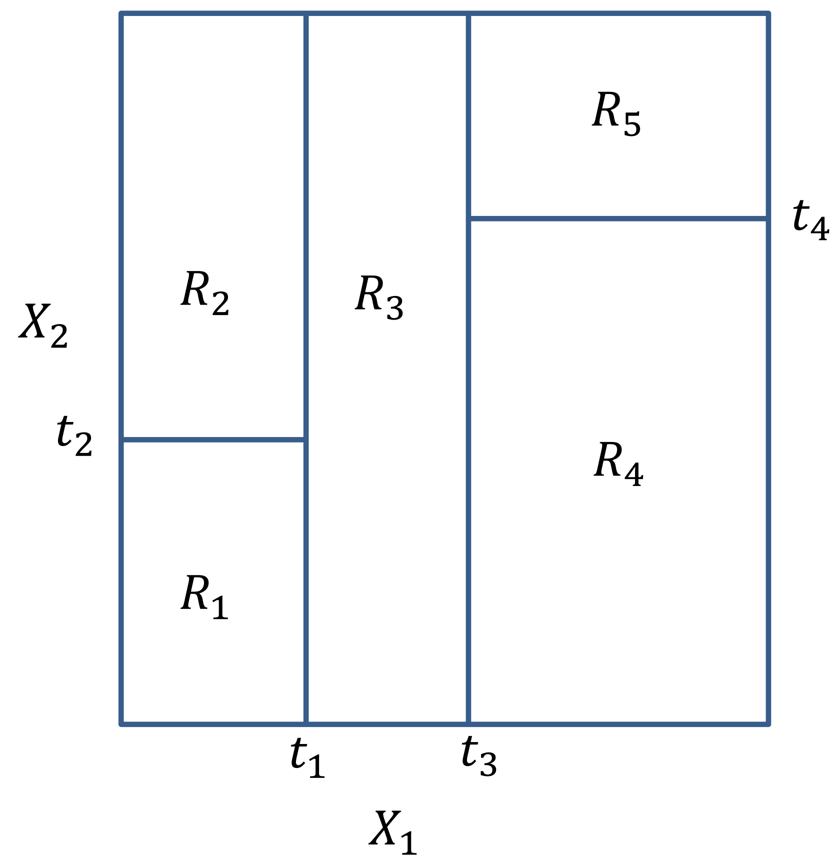
- It is computationally infeasible to consider every possible partition of the feature space into J regions.
- For this reason, a top-down, greedy approach is used
  - Known as recursive binary splitting.
- Rather using a brute-force solution, we would like to work in a heuristic way.

Solution

- How the heuristic method works?
  - Select a feature x_j and a threshold s such that
    - Split the feature space
      - Regions $\{x|x_j \leq s\}$ and $\{x|x_j \geq s\}$
        - leads to the best possible reduction in training error
  - Not going into the joint space of all features
    - Use independent feature form such as x_j with a threshold s .
  - Repeat the process
    - Looking for the best feature and the best threshold
      - Minimize the error in each of the resulting regions.
    - instead of splitting the entire feature space, we only split one of the two previously identified regions.
    - The splitting process continues until a stopping criterion is reached.

Prediction using DT

- We predict the response for a given test instance
  - using the mean (or mode) of the training instances in the region where the test observation falls.



Classification trees...

- In the classification setting
  - we replace the sum of square error by the classification error rate as a criterion for making the binary splits.
  - The classification error rate (E) is defined as the fraction of the training instances in that region that do not belong to the most common class.

$$E = 1 - \max_k \hat{p}_{jk}$$

$where\ \hat{p}_{jk}$ represents the proportion (fraction) of training instances in the j^{th} region that are from k^{th} class

$$CoD = \max_k \hat{p}_{jk}$$

- CoD (certainty of distribution) and close to 1
  - almost all the training points inside a region are voting for a certain class label.

Classification trees...

- Classification error is being less sensitive for tree-growing
- Alternative solution:
  - Gini index (G) is defined as

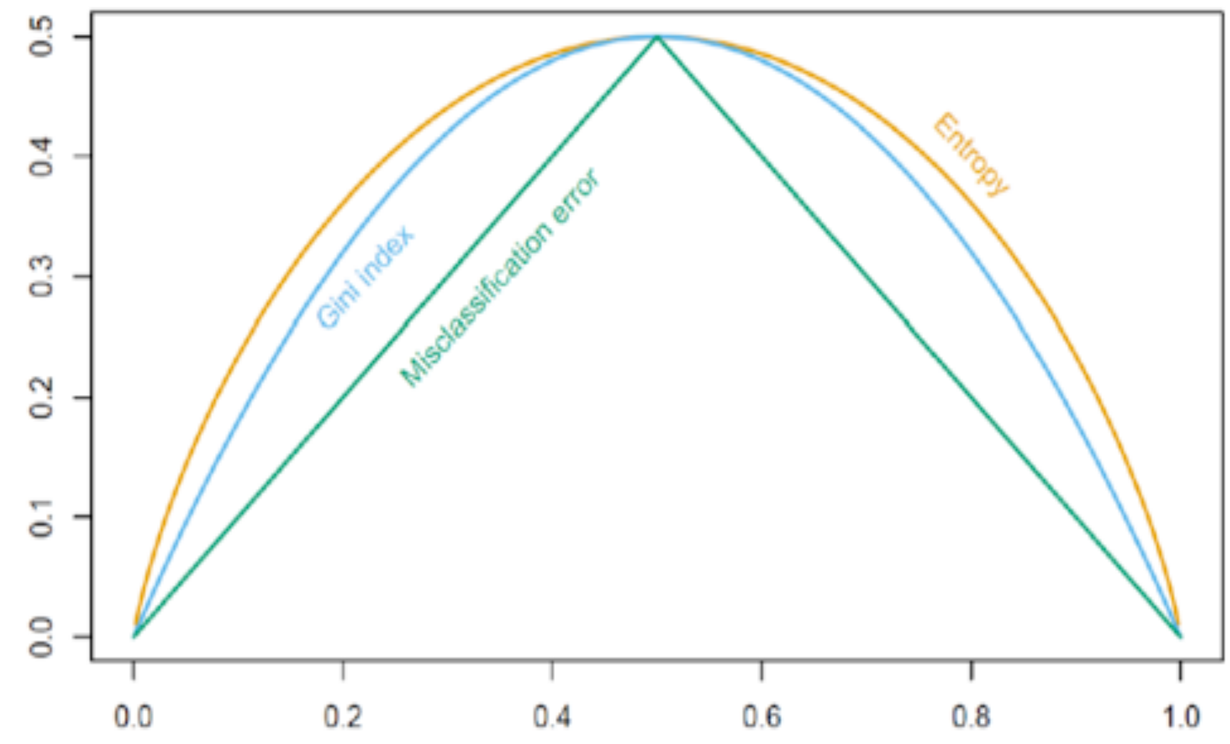
$$G = \sum_{k=1}^{K} \hat{p}_{jk}(1 - \hat{p}_{jk})$$

    - It is a measure of node purity. G becomes small as $\hat{p}_{jk}$ closes to either 0 or 1.
  - Entropy defined as:

$$D = -\sum_{k=1}^{K} \hat{p}_{jk} \log \hat{p}_{jk}$$

Classification trees...

- Pattern of Error, Gini Index and Entropy for different probabilities of class distributions:



Decision tree algorithms

- Three of the more popular ones are listed:
  - ID3 (Iterative Dichotomiser 3)
    - uses Entropy
  - *C4.5 (Successor of ID3)*
    - *slightly more advanced version of ID3 and uses Entropy*
  - *CART (Classification and Regression Tree)*
    - *uses Gini impurity*

Decision tree algorithms: ID3 Algorithm...

- Calculate the entropy of every feature using the data set S.
- Split the set S into subsets using the feature for which entropy is minimum.
  - So lesser values of entropy means it should be a good choice for selection of the attribute
- Make a "decision tree node" containing that feature.
- Recurse on subsets using remaining features.

Decision tree algorithms: ID3 Algorithm...

- If the tree is very deep
  - It partitions the feature space into small regions.
    - Small number of training points in sub-regions.
      - Increases variance and estimation becomes poor
- If the tree is shallow
  - Large regions
    - Small variance but large bias
- Need to find the sweet point
  - Depth of the decision tree
  - Cross-validation as discussed earlier (lecture – week 5)

Model complexity and pruning

- Pruning is a technique that reduces the size of decision trees
  - Removes sections of the tree
    - Little power to classify instances.
- The tree-building process
  - Overfit (creating deep trees)
  - Underfit (creating small number of regions)
- Generally, there are several ways of pruning trees:
  - Pre-pruning (forward pruning)
  - Post-pruning (backward pruning)

Model complexity and pruning: Pre-pruning

- In pre-pruning
  - Decision is made during the building process
    - Stop adding nodes (e.g., by looking at entropy).
- In case of Entropy
  - Check the amount of entropy reduction by selecting different features.
  - Stop splitting when the entropy reduction is not significant.
- Pre-pruning can be problematic
  - Sometimes attributes individually do not contribute much to a decision, but combined, they may have a significant impact.

Model complexity and pruning: Post-pruning

- Post-pruning waits until the full decision tree has been built
  - Then prunes the attributes by subtree replacement.
- Replace an entire subtree with a single region or node
  - It reproduces the smallest error.
- Select a subtree
  - Check – replacing it with a single node or feature incurs a small amount of change in Entropy.
  - If yes, trim the tree. If not, keep that subtree

Decision trees : Advantages/Disadvantages

- Advantages:
  - Decision trees are very easy to understand
  - Decision trees are capable of modelling nonlinear functions.
  - Decision tree can handle categorical variable
- Disadvantages:
  - Sensitive to small changes in the data.
    - Adding few data points or change some small values will change the DT
  - May overfit easily
    - Deep decision trees increases risk of overfitting and high variance model.
  - Only axis-aligned splits.
    - Considers each feature independently
    - No joint probabilities of features
  - Performance is not competitive
    - SVM, KNN or Neural network

Impact of distance metrics on KNN performance (Advanced topics)

- **KNN** classifies new data points according to their closeness
  - Neighbours
    - Distance measures
- Effectiveness of KNN
  - Selection distance metrics
  - Euclidean distance, Manhattan distance, and cosine similarity etc.

- Please use the following link for further explanation.

References:

1.Prasath, V. B., et al. "[Distance and Similarity Measures Effect on the Performance of K-Nearest Neighbor Classifier--A Review.]" arXiv preprint arXiv:1708.04321 (2017).

Feature importance of using Decision Trees (DT)

- **Decision Trees** uses feature selection to determine the most important classif ication features.
- DT operates by recursively segmenting the data into subsets based on the mo st informative features until a **stopping criterion** is reached.
    - Information gain or the Gini index
- At each node of the tree, the feature with **the highest score** is chosen as the s plitting criterion.
    - The significance of each feature can be determined by considering how much it **contributes**.

References:

1.Grabczewski, Krzysztof, and Norbert Jankowski. "**Feature selection with decision tree criterion**." Fifth International Conference on Hybrid Intelligent Systems (HIS'05). IEEE, 2005.

Thank You.