## SL00: Supervised Learning

In a supervised learning setting, the learning algorithm will be provided with instances and labels as training data.

- Classification: The classification task can be summarized as taking input x (e.g. images of traffic signs) and assign it to a label y that belongs to a discrete number of labels Y (e.g. STOP, Speed Limit, Keep Right, etc.).
- Regression: The regression task can be summarized as taking input x (e.g. employee's years of experience) and mapping it to a real continuous value (e.g. expected salary).

### Terminology:

- Instances: Inputs (Features).
- Concept: A set of functions that make good candidates to map inputs to outputs.
- Target concept: The actual function that can map inputs to outputs. In other words, the concept that we're willing to model.
- Hypothesis class: A subset of the concept set of functions that is easier to search through for the target concept.
- Sample (Training set): A set of instances with correct labels.
- Candidate: The best approximation of the target concept.
- Testing set: A set of instances with correct labels that was not visible to the learning algorithm during the training phase. It's used to determine the algorithm performance on novel data.

## SL01: Decision Trees:

A sequence of decision making points (nodes) applied to every instance to assign it to a specific class.
Example: Classifying vehicles into Sedans and Trucks.

### Representation vs Algorithm:

- A decision tree is a structure of nodes, edges and leaves that can be used to represent the data.
    1. Nodes represent attributes (vehicle length, vehicle height, number of doors, etc.).
    2. Edges represent values. (A specific value for each attribute)
    3. Leaves represent the output (A classification decision).
- A decision tree algorithm is a sequence of steps that will lead you to the desired output.
    1. Pick the best attribute.
    2. Ask a question about this attribute.
    3. Follow the correct answer path.
    4. Loop back to (1) till you narrow down the possibilities to one answer (the output).

    Note that the usefulness of a question depends upon the answers we got to previous questions.

## Expressiveness:

- Decision trees can basically express any function using the input attributes.
- For boolean functions (AND, OR, etc.), the truth table row will be translated into a path to a leaf.
  Add boolean functions decision trees here

## ID3 Algorithm:

- Pseudocode:

```
1. Pick the best attribute (A).
2. Assign (A) as a decision attribute for a node.
3. For each value of (A), create a descendent of node.
4. Sort training examples to leaves.
5. If (examples perfectly classified) {
           Stop
   } else {
           Iterate over leaves
   }
```

- Finding the best attribute:
  We can find the best attribute by calculating the information gain of each particular attribute, which is the amount of information gained by picking this attribute.
  Mathematically, the information gain quantifies the reduction in randomness over the labels we have with a set of data, based upon knowing the value of a particular attribute.
  Add information gain formula here

  - S is the set of training examples.
  - A is an attribute.
  - H is the entropy (Randomness) defines as:

  Add entropy formula here
  So ID3 is an algorithm of maximizing over the information gain.
- ID3 Bias:
  We have two kind of biases:
  1. Restrictive bias: Describes the hypothesis set space *H* that we will consider for learning.
  2. Preference bias: Describes the subset of hypothesis *n*, from the hypothesis set space *H (n belongs to H)*, that we prefer.

  The ID3 algorithm will prefer these decision trees:
  1. Good splits at the top of the tree.
  2. Trees that gives the correct answers.
  3. Prefers shorter trees vs. longer trees.

## Other considerations:

- How to handle continuous attributes?
  - Use intervals. Split the attribute range into equal intervals.
  - A modified ID3 can be used to find the best split.
- Does it make sense to repeat an attribute along a path in the tree?
  - No if this attribute has a finite value.
  - Continuous attributes can be tested with different questions.

- When do we stop?
  - When everything is classified correctly.
  - No more attributes.
  - Don’t overfit:
    - ➔ Not too big tree.
    - ➔ We can use cross-validation to prevent overfitting.
    - ➔ Use a validation set and stop when the error is low enough.
    - ➔ Build the whole tree, the prune it. Pruning means removing sections of the tree that provide little power in classifying instances.