Data Science and Analytics (SW-326)

MACHINE LEARNING – SUPERVISED (CLASSIFICATION)

Dr. Areej Fatemah Meghji areej.fatemah@faculty.muet.edu.pk

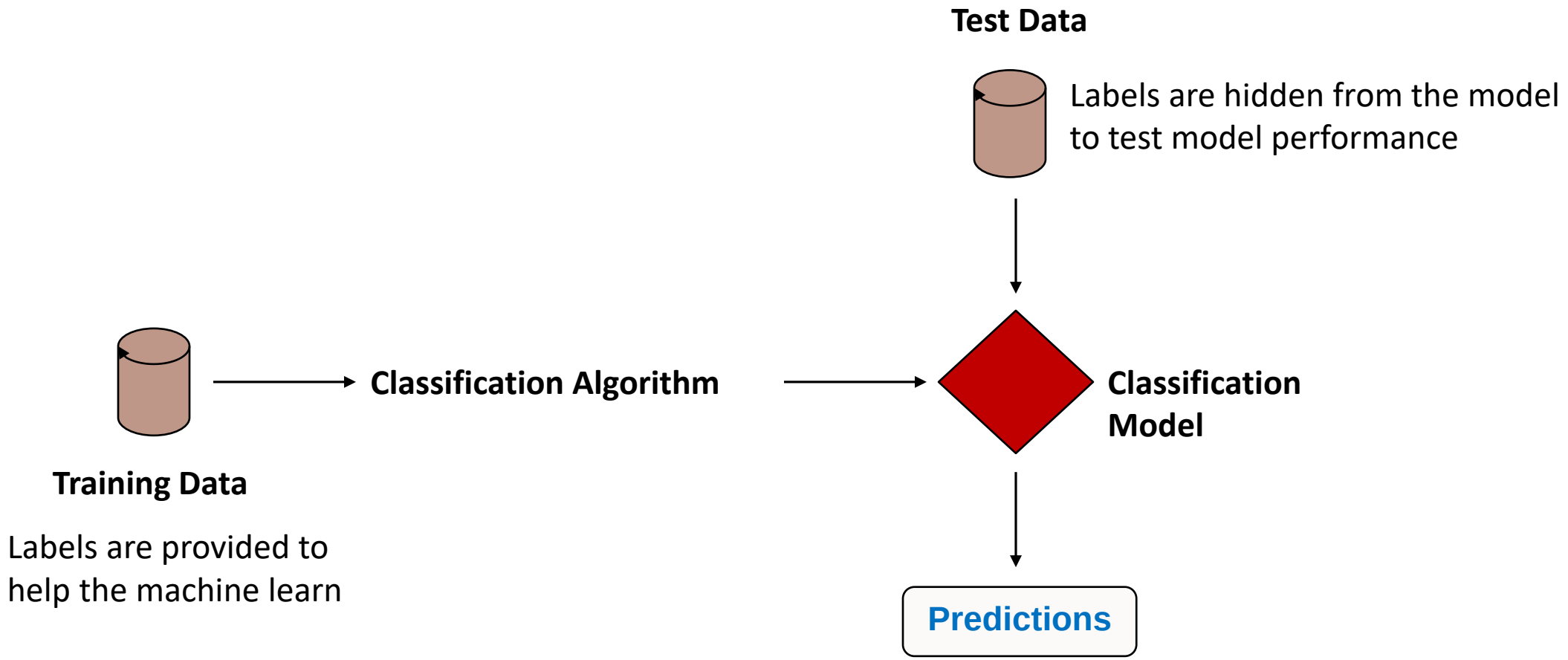
Machine Learning

- Machine learning is a method of data analysis that automates analytical model building.
- It is based on the idea that systems, with minimal human intervention, can :
  - ***learn from data***,
  - ***identify patterns,*** and
  - ***make decisions***
- What is required to create good machine learning systems?
  - Data preparation capabilities.
  - Algorithms – basic and advanced.
  - Automation and iterative processes.
  - Scalability.
  - Ensemble modeling.

Supervised Machine Learning

- In this type of machine learning, data scientists supply algorithms with labeled training data and define the variables they want the algorithm to assess for correlations.
  - Both the input and the output of the algorithm is specified.
- Supervised machine learning algorithms can apply what has been learned in the past to new data using labeled examples to predict future events.
- Starting from the analysis of a known training dataset, the learning algorithm produces an inferred function (model) to make predictions about the output values. The system is able to provide targets for any new input after sufficient training.
- The learning algorithm can also compare its output with the correct, intended output and find errors in order to modify the model accordingly.

Supervised Machine Learning
Classification



Supervised Machine Learning

- Supervised machine learning requires the data scientist to train the algorithm with both labeled inputs and desired outputs.
- Supervised learning algorithms are good for the following tasks:
  - Binary classification: Dividing data into two categories.
  - Multi-class classification: Choosing between more than two types of answers.
  - Regression modeling: Predicting continuous values.
  - Ensembling: Combining the predictions of multiple machine learning models to produce an accurate prediction.

- **Datasets in Supervised Machine Learning:**
  - Training set: set of examples used for learning, where the target value is known
  - Validation set: set of examples used to tune the architecture of a classifier and estimate the error.
  - Test set: used only to assess the performances of a classifier. It is never used during the training process so that the error on the test set provides an unbiased estimate of the generalization error.

Supervised Learning
Classification

- Everything begins with training a machine-learning model.
  - Machine Learning Model: a mathematical function capable of repeatedly modifying how it operates until it can make accurate predictions when given fresh data.
- Before training begins, you first have to choose which data to gather and decide which features of the data are important.
  - An important point to note is that the ***data has to be balanced***.
- Before training gets underway there will generally also be a data-preparation step, during which processes such as duplication, normalization and error correction will be carried out.

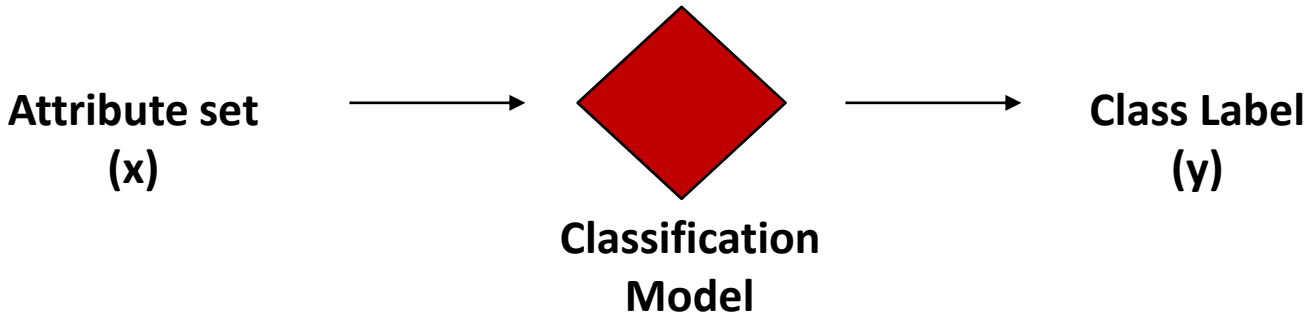
Supervised Learning
Classification

- The gathered data is then split
  - larger proportion for training, about 70%, and a smaller proportion for evaluation, the remaining 30%.
  - This evaluation (test) data allows the trained model to be tested, to see how well it is likely to perform on real-world data.

- The next step will be choosing an appropriate machine-learning algorithm from the wide variety available.
- Each have strengths and weaknesses depending on the type of data, for example some are suited to handling images, some to text, and some to purely numerical data.

Supervised Learning
Classification Algorithms

- Rule-based
- Decision Trees
- Bayesian Classifiers (Probabilistic)
- K-Nearest Neighbor (Memory based – lazy)
- Neural Networks



Classification Algorithms
Rule-Based

- The term rule-based classification can be used to refer to any classification scheme that make use of **IF-THEN** rules for class prediction.
- These rules are easily interpretable and thus these classifiers are generally used to generate ***descriptive models***.
- Rule notation: (Condition) ⟶ y
  - Where: Condition is a conjunctions of attributes and y is the class label
- Rule-based classification schemes typically consist of the following components:
  1. ***Rule Induction Algorithm:*** This refers to the process of extracting relevant IF - THEN rules from the data.
  2. ***Rule Ranking Measures:*** This refers to some values that are used to measure the usefulness of a rule in providing accurate prediction.
     - Rule ranking measures are often used in the rule induction algorithm to prune off unnecessary rules and improve efficiency. They are also used in the class prediction algorithm to give a ranking to the rules which will be then be utilized to predict the class of new cases.

Classification Algorithms
Rule-Based

Building Classification Rules

- **Rule Induction Using Sequential Covering Algorithm (Direct Method)**
- Sequential Covering Algorithm can be used to extract IF-THEN rules form the training data.
- In this algorithm, each rule for a given class covers many of the tuples of that class.
- Some of the sequential Covering Algorithms are AQ, CN2, and RIPPER.
- As per the general strategy the rules are learned one at a time.
  - For each time rules are learned, a tuple covered by the rule is removed and the process continues for the rest of the tuples.
- **Rule Induction using Indirect Method:**
- Extract rules from other classification models (e.g. decision trees, neural networks, etc).
  - Examples: ***C4.5 rules***

Classification Algorithms
Rule-Based

Measures of Convergence and Accuracy

- **Coverage of a rule:**
  - Fraction of records that satisfy the antecedent of a rule
  - Count(instances with antecedent) / Count(training set)
  - **Example Rule:** ***(Blood Type = cold)*** $\rightarrow$ ***fish***
  - Coverage = 4/10 = 40%

Name	Blood Type	Give Birth	Can Fly	Live in Water	Class
human	warm	yes	no	no	**mammal**
python	cold	no	no	no	**reptile**
salmon	cold	no	no	yes	**fish**
whale	warm	yes	no	yes	**mammal**
bat	warm	yes	yes	no	**mammal**
pigeon	warm	no	yes	no	**bird**
turtle	cold	no	no	stime	**reptile**
penguin	warm	no	no	stime	**bird**
eel	cold	no	no	yes	**fish**
owl	warm	no	yes	no	**bird**

- **Accuracy of a rule:**
  - Fraction of records satisfying the antecedent which also satisfy the consequent of a rule
  - Count (instances with antecedent AND consequent) / Count(instances with antecedent)
  - **Example Rule:** ***(Blood Type = cold)*** $\rightarrow$ ***fish***
  - accuracy = 2/4 = 50%

Classification Algorithms
Rule-Based

- A record may trigger more than one rule
- **Solution?**
  - Ordered rule set
  - Unordered rule set – use voting schemes

- A record may not trigger any rules
- **Solution?**
  - Use a default class

Classification Algorithms
Rule-Based

- **Ordered Rule Set**
- Rules are rank ordered according to their priority
- An ordered rule set is known as a ***decision list***
- When a test record is presented to the classifier:
  - it is assigned to the class label of the highest ranked rule it has triggered (first encountered)
  - If none of the rules fired, it is assigned to the default class

Classification Algorithms
Rule-Based

- **Rule Ordering:**
- **Rule-based ordering:** Individual rules are ranked based on their quality
- **Class-based ordering:** Rules that belong to the same class appear together, ordered by class sizes

Direct Method: Sequential Covering

1. Start from an empty rule
2. Grow a rule using the **Learn-One-Rule** function
3. Remove training records covered by the rule
4. Repeat Step (2) and (3) until stopping criterion is met

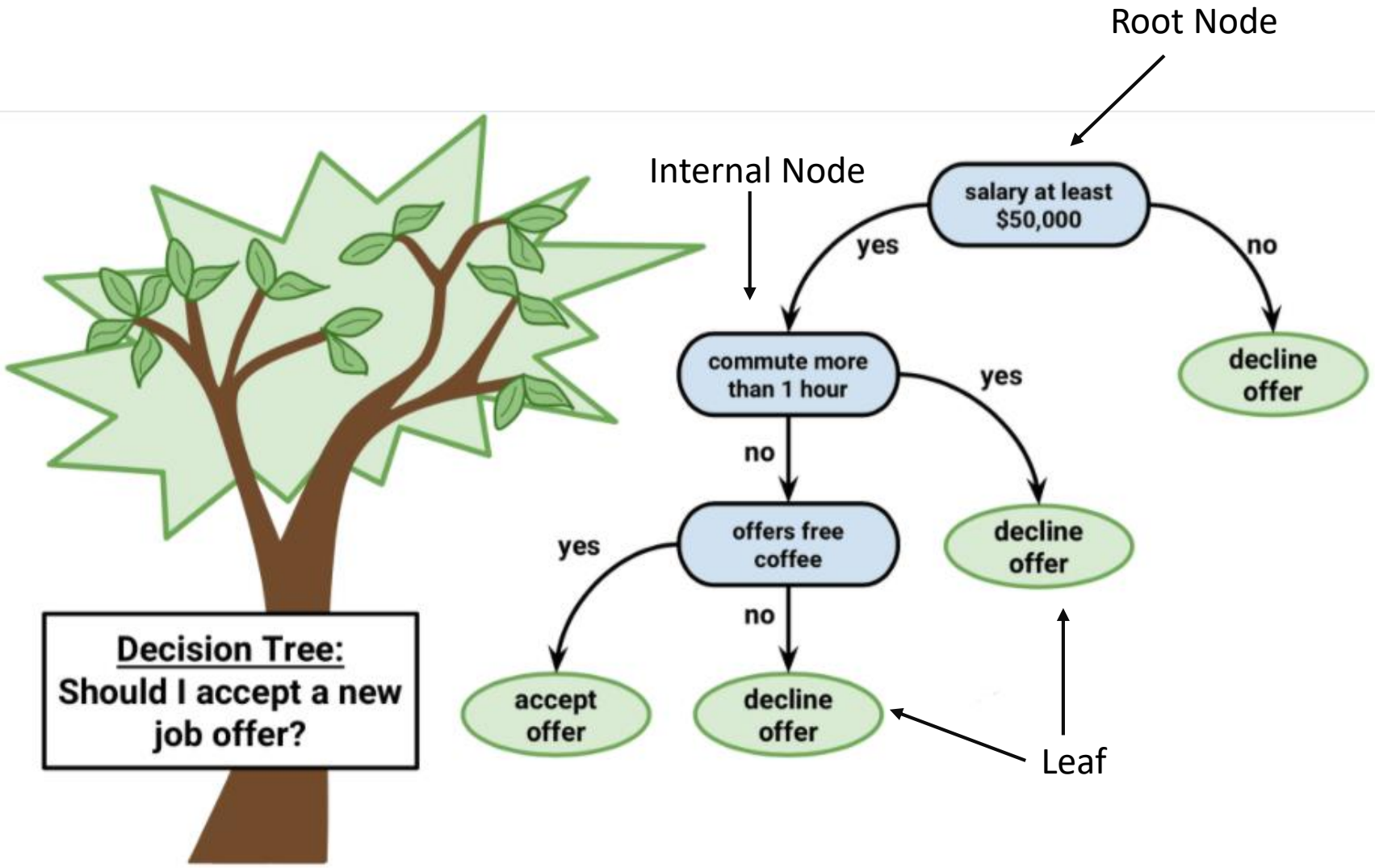
Classification Algorithms
Decision Trees

- The goal of using a Decision Tree is to create a training model that can be used to predict the class by learning simple decision rules inferred from prior data (training data).
- In Decision Trees, for predicting a class label for a record, we start from the **root of the tree**.
  - We compare the values of the root attribute with the record's attributes.
- On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

Classification Algorithms
Decision Trees

A decision tree is drawn upside down with its root at the top.

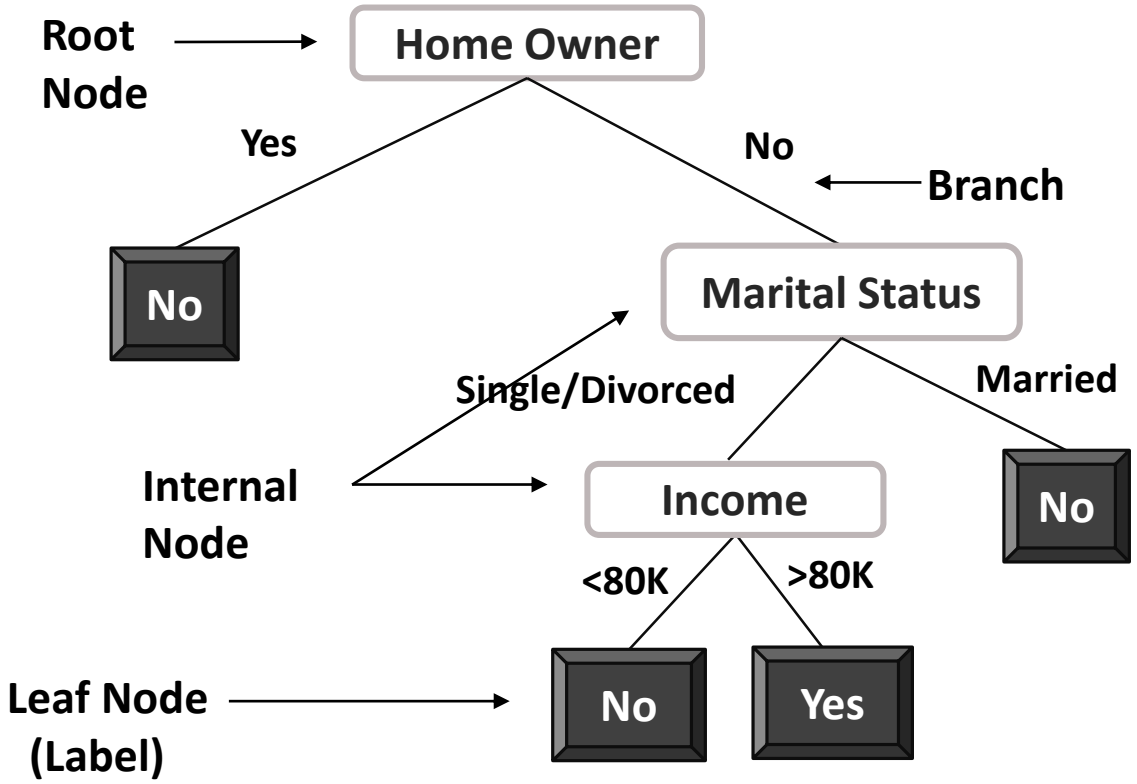
In the image on the right, the tree is split based on condition / internal nodes. The tree branches based on the result of the condition. The end of the branch that doesn't split anymore is the decision/leaf (class label)



Classification Algorithms
Decision Trees

- Let's identify important terminologies regarding a Decision Tree:
- **Root Node** represents the entire population or sample. It further gets divided into two or more homogeneous sets.
- **Splitting** is a process of dividing a node into two or more sub-nodes.
- When a sub-node splits into further sub-nodes, it is called a **Decision Node**.
- Nodes that do not split is called a **Terminal Node** or a **Leaf**.
- A sub-section of an entire tree is called **Branch**.
- A node, which is divided into sub-nodes is called a **parent node** of the sub-nodes; whereas the sub-nodes are called the **child** of the parent node.
- When you remove sub-nodes of a decision node, this process is called **Pruning**.

Classification Algorithms
Decision Tree

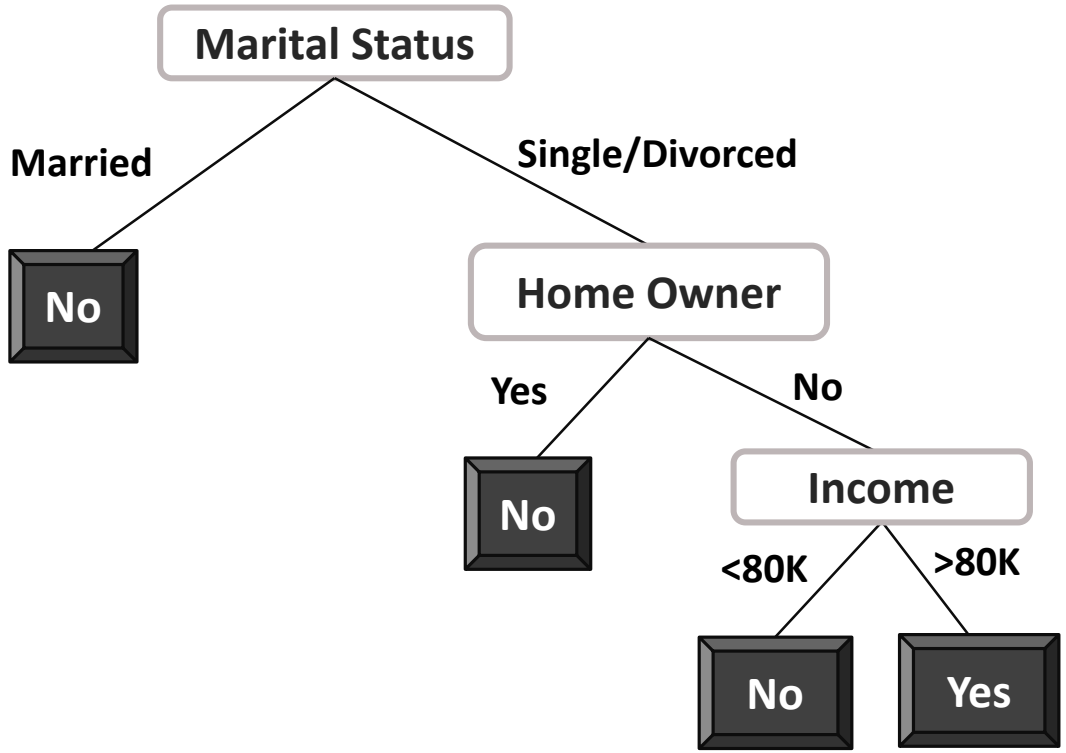


ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	Single	85K	**Yes**
9	No	Married	75K	**No**
10	No	Single	90K	**Yes**

Training / Induction

Classification Algorithms
Decision Tree

There can be more than one tree that fits the same training data

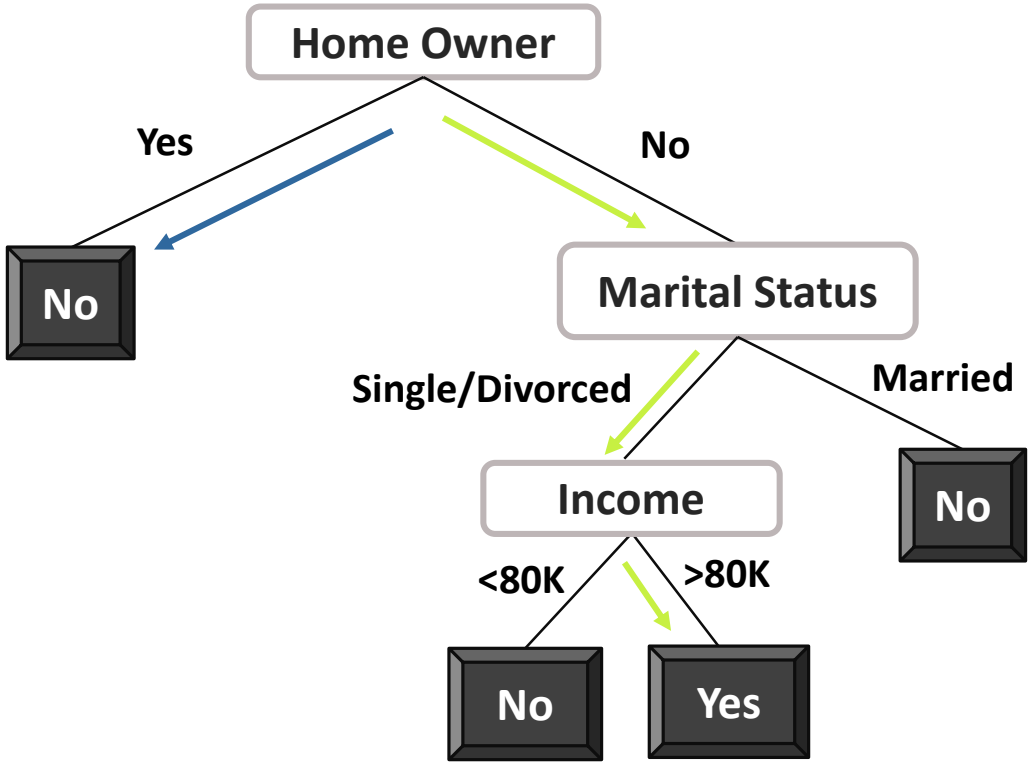


ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	95K	**Yes**
6	No	Married	60K	**No**
7	Yes	Divorced	220K	**No**
8	No	Single	85K	**Yes**
9	No	Married	75K	**No**
10	No	Single	90K	**Yes**

Training / Induction

Classification Algorithms
Decision Tree

Apply Model to Test Data



ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Married	80K	**No**
2	No	Single	102K	**Yes**

Testing / Deduction

Point to remember:
Once a decision tree has been constructed, it is a simple matter to convert it into an equivalent set of rules. Converting to rules improves readability as rules are often easier for people to understand

This is referred to as Indirect Induction

Classification Algorithms
Decision Tree

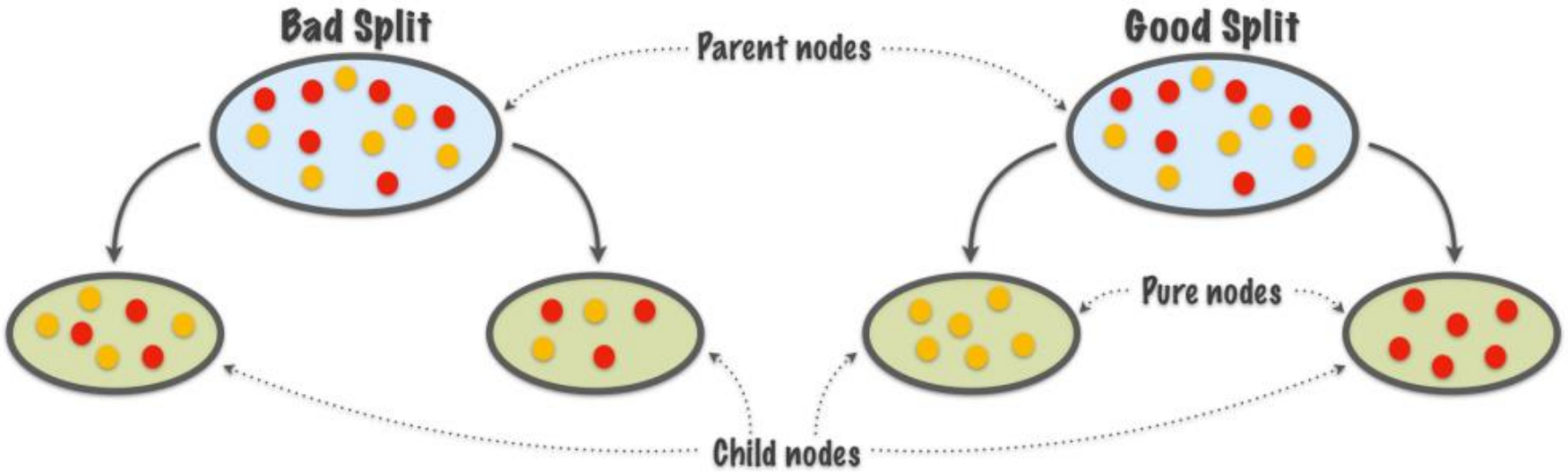
Design Issues of Decision Tree Induction

- Methods for Expressing Test Conditions
  - Attribute types to consider: binary, nominal, ordinal, continuous
  - Splitting options: binary, multi-way

Classification Algorithms
Decision Tree

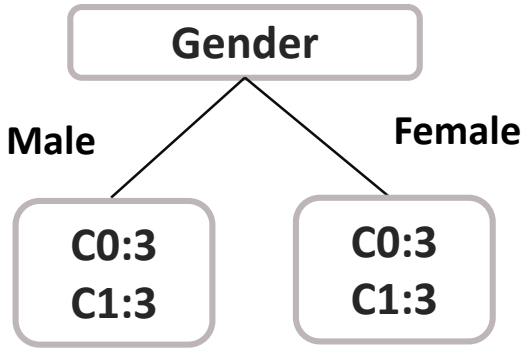
Design Issues of Decision Tree Induction

- How should training records be split?
  - Method for specifying test condition
    - depending on attribute types
  - **Measure for evaluating the goodness of a test condition**

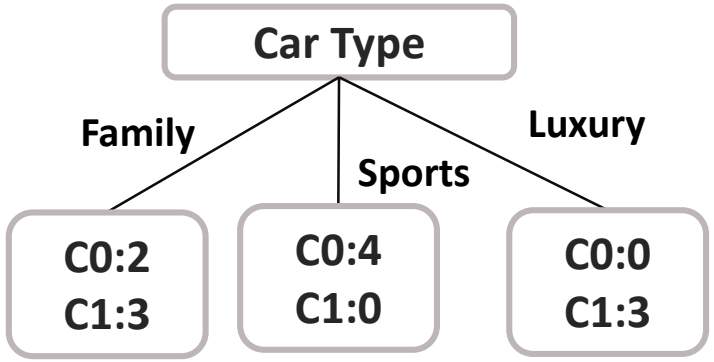


Classification Algorithms
Decision Tree

Before Splitting: 6 records of class 0 (C0)
6 records for class 1 (C1)



no solid or definitive determination



fairly predictable outcomes based on car type

ID	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	**C0**
2	M	Sports	Medium	**C0**
3	M	Sports	Medium	**C0**
4	F	Sports	Small	**C0**
5	F	Sports	Small	**C0**
6	F	Family	Medium	**C0**
7	M	Luxury	Large	**C1**
8	M	Family	Extra Large	**C1**
9	M	Family	Large	**C1**
10	F	Luxury	Medium	**C1**
11	F	Luxury	Large	**C1**
12	F	Family	Small	**C1**

Nodes with a *purer* class distribution are preferred
Measure of node impurity:
Entropy
Information Gain

Classification Algorithms
Decision Tree

- The process of building a decision tree involves asking a question at every instance and then continuing with the split.
  - When there are multiple features that decide the target value of a particular instance, which feature should be chosen as the root node to start the splitting process?
  - And in which order should we continue choosing the features at every further split at a node?
- The main idea of a decision tree is to identify the features which contain the most information regarding the target feature and then split the dataset along the values of these features such that the target class values at the resulting nodes are as pure as possible.
  - A feature that best separates the uncertainty from information about the target feature is said to be **the most informative feature**.
  - The search process for a most informative feature goes on until we end up with pure leaf nodes.

need to measure the informational value of the features and use the feature with the most information as the feature to split the data on.

Classification Algorithms
Decision Tree

- **Entropy**: It is the degree of uncertainty, impurity or disorder of a random variable, or a measure of purity.
  - It characterizes the impurity of an arbitrary class of examples.
  - used to measure the impurity or randomness of a dataset.

- Imagine choosing a yellow ball from a box of just yellow balls (say 100 yellow balls).
  - Then this box is said to have 0 entropy which implies 0 impurity.

- Now, let's say 30 of these balls are replaced by red and 20 by blue.
  - If we now draw another ball from the box, the probability of drawing a yellow ball will drop from 1.0 to 0.5.
  - Since the impurity has increased, entropy has also increased while purity has decreased.

The higher the entropy, the harder it is to draw any conclusions from that information

Classification Algorithms
Decision Tree

- We want to determine which attribute in a given set of training features is most useful for discriminating between the classes to be learned.
  - **Information gain** tells us how important a given attribute of the feature vectors is. We will use it to decide the ordering of attributes in the nodes of a decision tree.
- To find the best feature which serves as a root node in terms of information gain, we first use each descriptive feature and split the dataset along the values of these descriptive features and then calculate the entropy of the dataset.

Information Gain = original entropy – entropy after split

Information Gain = entropy(parent) – [average entropy(children)]

- This gives us the remaining entropy once we have split the dataset along the feature values.
  - We subtract this value from the originally calculated entropy of the dataset to see how much this feature splitting reduces the original entropy which gives the information gain of a feature.
- **Constructing a decision tree is all about finding an attribute that returns the highest information gain and the smallest entropy.**
  - The feature with the largest information gain should be used as the root node to start building the decision tree.

Classification Algorithms
Decision Tree

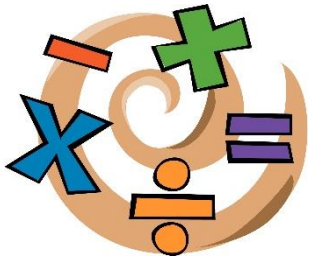
Advantages:

- **Easy to interpret**. Closely mirror human decision-making.
  - While other machine Learning models are close to black boxes, decision trees provide a graphical and intuitive way to understand what our algorithm does.
- Compared to other Machine Learning algorithms Decision Trees require **less data** to train.
- They can be used for **Classification** and **Regression**.
- They are **simple**.
- They are tolerant to **missing values**.

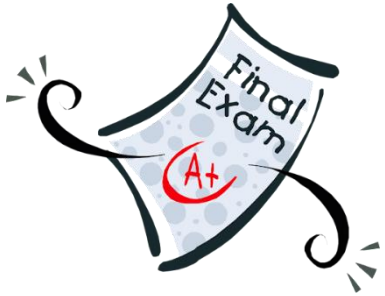
Disadvantages:

- They are quite prone to **over fitting** to the training data and can be susceptible to outliers.
  - The splitting process results in fully grown trees until the stopping criteria are reached. But, the fully grown tree is likely to overfit the data, leading to poor accuracy on unseen data.
- **They are weak learners**: a single decision tree normally does not make great predictions, so multiple trees are often combined to make '*forests*' to give birth to stronger ensemble models.

Classification Algorithms
Decision Tree



- Let's say three students have prepared for a mathematics examination.
- The **first student** has only studied Addition mathematic operations and skipped other mathematics operations such as Subtraction, Division, Multiplication etc.
- The **second student** has a particularly good memory. Thus, second student has memorized all the problems presented in the textbook.
- And the **third student** has studied all mathematical operations and is well prepared for the exam.
- In the exam the first student will only be able to solve the questions related to Addition and will fail in problems or questions asked related to other mathematics operations.
- The second student will only be able to answer questions if they happened to appear in the textbook (as he has memorized it) and will not be able to answer any other questions.
- The third student will be able to solve all the exam problems reasonably well.



Classification Algorithms
Decision Tree

- Machine Learning algorithms have similar behavior to our three students.
- Sometimes the model generated by the algorithm are similar to the first student.
  - They learn from only from a small part of the training dataset, in such cases the model is **Underfitting**.
- Sometimes the model will memorize the entire training dataset, like the second student.
  - They perform very well on known instances, but fault badly on unseen data or unknown instances. In such cases the model is said to be **Overfitting**.
- And when model does well in both the training dataset and on the unseen data or unknown instances like the third student, it is a good fit.

Classification Algorithms
Decision Tree

- In machine learning we describe the learning of the target function from training data as inductive learning.
  - *Induction* refers to learning general concepts from specific examples (training set).
- **Generalization** refers to how well the concepts learned by a machine learning model apply to specific examples *not seen by the model when it was learning.*
- The goal of a good machine learning model is to generalize well from the training data to any data from the problem domain.
  - This allows us to make predictions in the future on data the model has never seen.
- There is a terminology used in machine learning when we talk about how well a machine learning model learns and generalizes to new data, *namely overfitting and underfitting.*
  - Overfitting and underfitting are the two biggest causes for poor performance of machine learning algorithms.

Classification Algorithms
Decision Tree

- In statistics, a fit refers to how well you approximate a target function.
- This is good terminology to use in machine learning, because supervised machine learning algorithms seek to approximate the unknown underlying mapping function for the output variables given the input variables.
  - *Generalization* is the model's ability to give sensible outputs to sets of input that it has never seen before.
- **Overfitting** happens when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data.
  - This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model.
  - The problem is that these concepts do not apply to new data and negatively impact the models ability to generalize.
- **Underfitting** refers to a model that can neither model the training data nor generalize to new data.
  - An underfit machine learning model is not a suitable model and will be obvious as it will have poor performance on the training data.

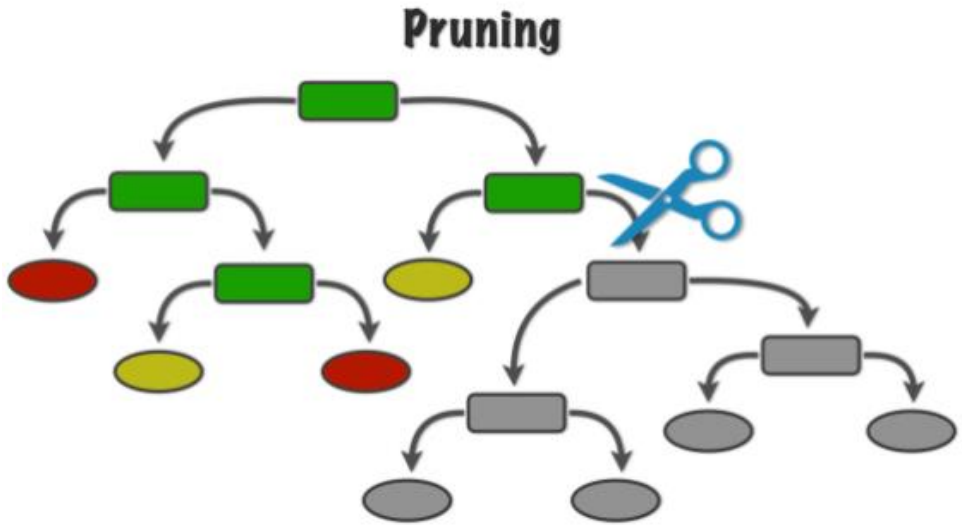
Classification Algorithms
Decision Tree

Design Issues of Decision Tree Induction

- How should the splitting procedure stop?
  - Stop splitting if all the records belong to the same class or have identical attribute values
  - Early termination

Classification Algorithms
Decision Tree

- The performance of a tree can be increased by pruning.
- **Pruning** reduces the size of decision trees by removing parts of the tree that do not provide power to classify instances.
  - It involves removing the branches that make use of features having low importance.
  - This way, we reduce the complexity of tree, and thus increasing its predictive power by reducing overfitting



- **Approaches:**
  - Pre-Pruning
  - Post-Pruning

Classification Algorithms
Decision Tree

- **Pre-Pruning (early stopping)**: stop the growth of a decision tree before it overfits to the training data.
- The idea here is to stop the trees growth before it makes overly niche splits that don't generalize well and, in practice, this works very well.
- In decision trees, the root node sequentially adds splits until the child nodes are pure, now we need an alternative rule that tells the tree to stop growing before the nodes are pure and **classifies the new terminal nodes by their majority class.**
- There are a huge number of potential stopping rules that can be conceived, so here is a non-exhaustive list of some popular choices:
  - Maximum tree depth: Simply pre define an arbitrary number for the maximum depth (or max number of splits) and once the tree reaches this value the growing process terminates.
  - Minimum number in node: Define a minimum number of observations to appear in any child node for a split to be valid.
  - Minimum decrease in impurity: Define a minimum acceptable decrease in impurity for a split to be accepted.
  - Maximum features: considering a subset of available features to split on may improve the final generalizability.

Classification Algorithms
Decision Tree

- **Post-Pruning** on the other hand takes a tree that has already been overfit and makes some adjustments to reduce/remove the observed overfitting.
- A good pruning rule will pinpoint the splits that don't generalize well, often by using an independent test set, and remove them from the tree. Again, there are many different approaches to implementing pruning but three popular choices are:
  - Critical value pruning: Retrospectively estimates the strength of each node from calculations done in the tree building stage. Nodes that don't achieve a certain critical value are pruned, unless a node further along the branch does reach it.
  - Error complexity pruning: Generates a series of trees each made by pruning the full tree by different amounts and selects one of these by assessing its performance with an independent data set.
  - Reduced error pruning: Runs the independent test data through the full tree and, for each non-leaf node, compares the number of errors if the sub tree from that node is kept vs removed. The pruned node will often make fewer errors using the new test data than the sub tree makes. The node that sees the biggest difference in performance is pruned and this process is continued until further pruning will increase the misclassification rate.

Classification
Cross-Validation

- The overall data set is divided into:
  1. the training data set
  2. validation data set
  3. test data set
- The training set is used to fit the different models.
- The process of deciding whether the numerical results quantifying hypothesized relationships between variables, are acceptable as descriptions of the data, is known as validation. Generally, an error estimation for the model is made after training, better known as evaluation of residuals. In this process, a numerical estimate of the difference in predicted and original responses is done, also called the training error.
- The performance on the validation set is used for the model selection.

Classification
Cross-Validation

- The advantage of keeping a test set that the model hasn't seen before during the training and model selection steps is that we avoid over-fitting the model and the model is able to better generalize to unseen data.
- In many applications, however, the supply of data for training and testing will be limited, and in order to build good models, we wish to use as much of the available data as possible for training.
- As there is never enough data to train your model, removing a part of it for validation poses a problem of underfitting. By reducing the training data, we risk losing important patterns/ trends in data set, which in turn increases error induced by bias.
- So, what we require is a method that provides ample data for training the model and also leaves ample data for validation. *K Fold cross validation does exactly that.*

K-Fold Cross Validation

- The *error estimation is averaged over all k trials to get total effectiveness of the model*.
- As can be seen, every data point gets to be in a validation set exactly once, and gets to be in a training set *k-1* times.
- ***This significantly reduces bias as we are using most of the data for fitting, and also significantly reduces variance as most of the data is also being used in validation set.***
- Interchanging the training and test sets also adds to the effectiveness of this method.
- **As a general rule and empirical evidence, K = 5 or 10 is generally preferred**, but nothing's fixed and it can take any value.

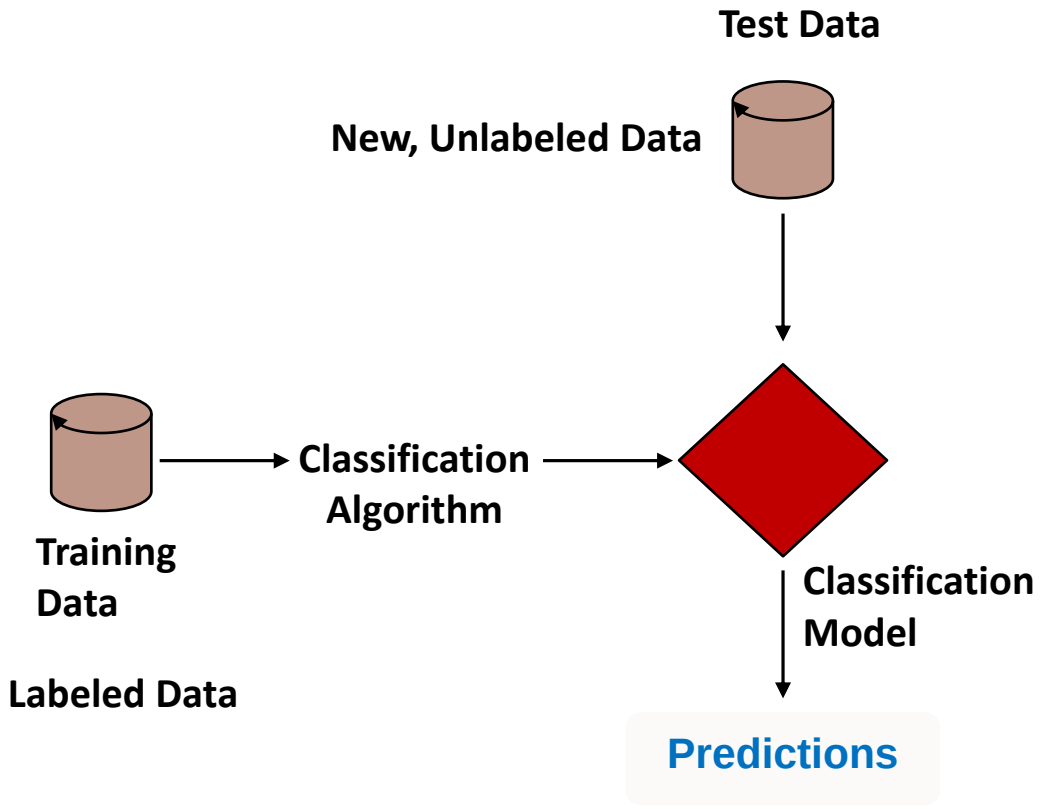
K-Fold Cross Validation

- The general procedure is as follows:

- Shuffle the dataset randomly.
- Split the dataset into k groups
- For each unique group:
  - Take the group as a test data set
  - Take the remaining groups as a training data set
  - Fit a model on the training set and evaluate it on the test set
  - Retain the evaluation score and discard the model
- Summarize the skill of the model using the sample of model evaluation scores.

- **Important:** each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure. This means that each sample is given the opportunity to be used in the hold out set 1 time and used to train the model k-1 times

Supervised Learning
Classification Algorithms - Memory based



Rule based classification

Stores the entire training data and classifies new data at run-time

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes

Training (Labeled Data)

today	cool	sunny	normal	false	

Test Data (New, Unlabeled)

Classification Algorithm

Predictions

Memory based classification

Supervised Learning
Classification Algorithms - Memory based

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes

Training Data

Looks for an exact match

today	cool	sunny	normal	false	?

Testing Data

Supervised Learning
Classification Algorithms - Memory based

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes

Training Data

Predicts label based on found match

today	cool	sunny	normal	false	yes

Testing Data

Supervised Learning
Classification Algorithms - Memory based

If an exact match is not found, looks for the next closest match

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes

Training Data

Testing Data

tomorrow	mild	sunny	normal	false	?

Supervised Learning
Classification Algorithms - Memory based

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes

Training Data

tomorrow	mild	sunny	normal	false	?

Testing Data

Supervised Learning
Classification Algorithms - Memory based

Day	Temperature	Outlook	Humidity	Windy	Play Golf?
07-05	hot	sunny	high	false	no
07-06	hot	sunny	high	true	no
07-07	hot	overcast	high	false	yes
07-09	cool	rain	normal	false	yes
07-10	cool	overcast	normal	true	yes
07-12	mild	sunny	high	false	no
07-14	cool	sunny	normal	false	yes
07-15	mild	rain	normal	false	yes
07-20	mild	sunny	normal	true	yes
07-21	mild	overcast	high	true	yes
07-22	hot	overcast	normal	false	yes
07-23	mild	rain	high	true	no
07-26	cool	rain	normal	true	no
07-30	mild	rain	high	false	yes

Training Data

tomorrow	mild	sunny	normal	false	yes

Testing Data

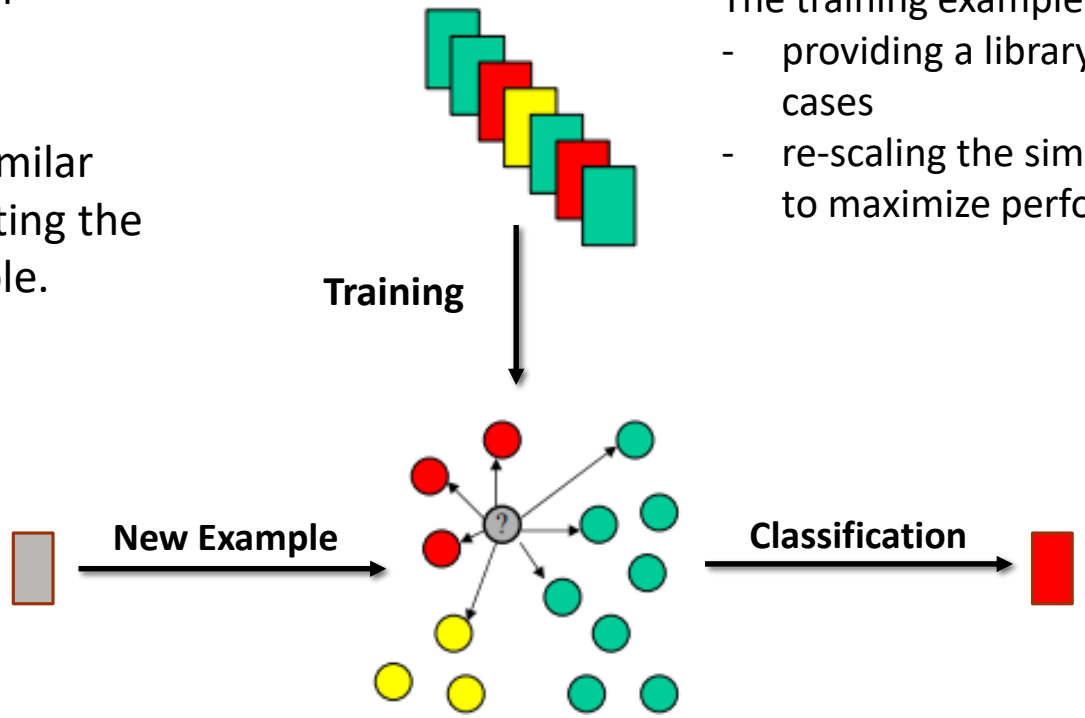
Supervised Learning
Classification Algorithms - Memory based

- **No model is learned**
  - The stored training instances themselves represent the knowledge
  - Training instances are searched for instance that most closely resembles new instance
- ***lazy learning***
  - Rote-learner: Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
- Other names for memory-based algorithms:
  - Lazy learners, Instance-based, Exemplar-based ,Case-based, Experience based
- This strategy is opposed to **eager** learning algorithms where:
  - Data is compiled into a compressed description or model
  - The training data is discarded after compilation of the model
  - Incoming patterns are classified using the induced model

Classification Algorithms
Memory based

K-Nearest Neighbor algorithms classify a new example by comparing it to all previously seen examples.

The classifications of the ***k*** most similar previous cases are used for predicting the classification of the current example.



The training examples are used for:

- providing a library of sample cases
- re-scaling the similarity function to maximize performance

Classification Algorithms
Memory based (kNN) - Nearest Neighbor Classifiers

- kNN is considered a lazy learning algorithm
  - Defers data processing until it receives a request to classify an unlabelled example
  - Replies to a request for information by combining its stored training data
  - Discards the constructed answer and any intermediate results

- Requires three things:
  - **Training set:** the set of labeled records
  - **A Distance Metric**: to compute distance between records
  - **Some value of *k***, the number of nearest neighbors to retrieve / consider

Classification Algorithms
Memory based (kNN) - Nearest Neighbor Classifiers

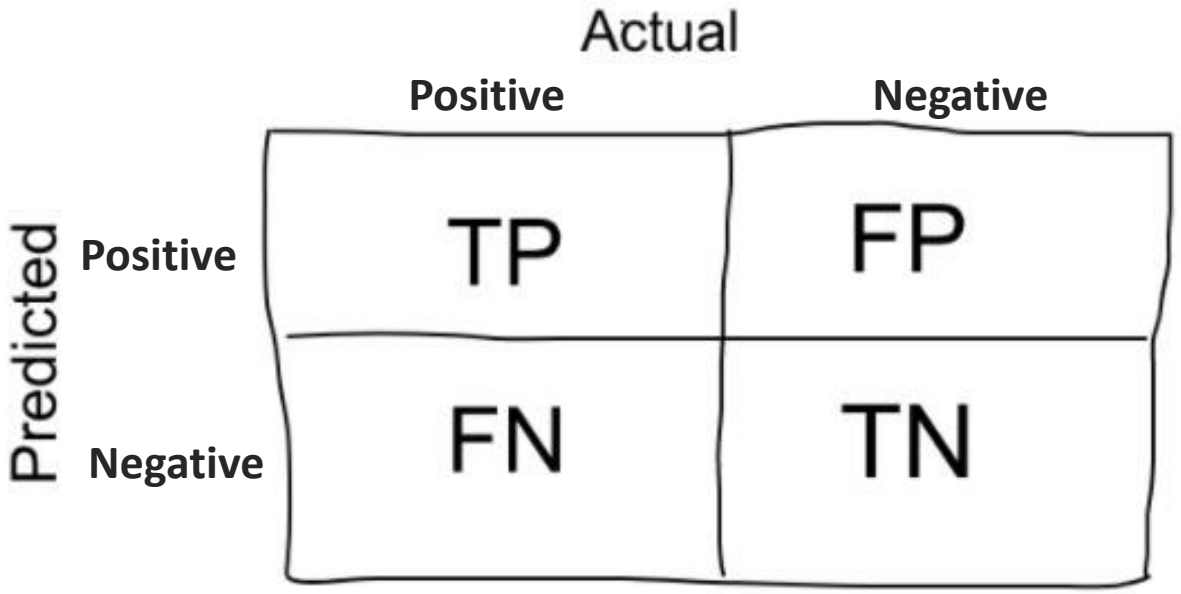
- Requires three things:
  - **Training set:** the set of labeled records
  - **A Distance Metric**: to compute distance between records
  - **Some value of *k***, the number of nearest neighbors to retrieve / consider
- To classify an unknown record:
  - Compute distance to other training records
  - Identify *k* nearest neighbors
  - Use class labels of nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)
- K-nearest neighbors of a record x are data points that have the k smallest distances to x

Issues with k-NN

- k-NN classifiers are **lazy learners** since they do not build models explicitly, a priori
- Classifying unknown records are relatively expensive
- Can result in arbitrarily shaped decision boundaries
- Easy to handle variable interactions since the decisions are based on local information
- Selection of right proximity measure is essential
- Superfluous or redundant attributes can create problems
- Missing attributes are hard to handle

Evaluating Classification Models
Performance Evaluation

- **Confusion Matrix:**

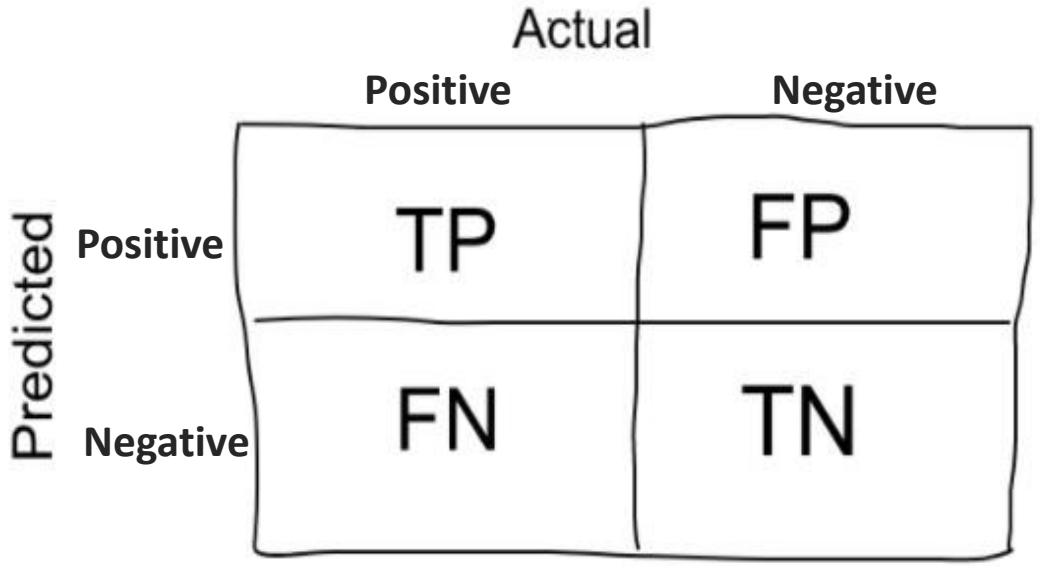


Evaluating Classification Models
Confusion Matrix:

- **True Positives (TP):** True positives are the cases where the actual class of the data point was positive and the predicted is also positive.
- **True Negatives (TN):** True negatives are the cases when the actual class of the data point was negative and the predicted is also negative.
- **False Positives (FP):** False positives are the cases when the actual class of the data point was negative and the predicted is positive.
- **False Negatives (FN):** False negatives are the cases when the actual class of the data point was positive and the predicted is negative.

Evaluating Classification Models
Performance Evaluation

- **Confusion Matrix:**



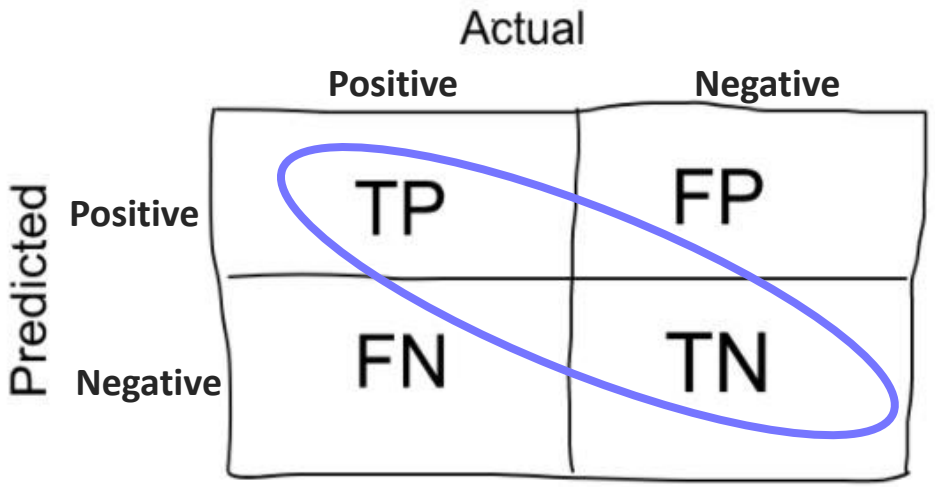
There are 100 people whose temperature was checked.
45 were correctly allowed entry
30 were correctly denied entry
15 were incorrectly allowed entry
10 were incorrectly denied entry

		Actual: Allow	Actual: Deny
Predicted	Allow	45	15
	Deny	10	30

Evaluating Classification Models
Performance Evaluation - Accuracy

- Accuracy in classification problems is the number of correct predictions made by the model over all kinds predictions made.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN}$$



There are 100 people whose temperature was checked.
45 were correctly allowed entry
30 were correctly denied entry
15 were incorrectly allowed entry
10 were incorrectly denied entry
Find the accuracy of the model.

		Actual: Allow	Actual: Deny
Predicted	Allow	45	15
	Deny	10	30

$$Accuracy = \frac{45 + 30}{45 + 15 + 10 + 30}$$

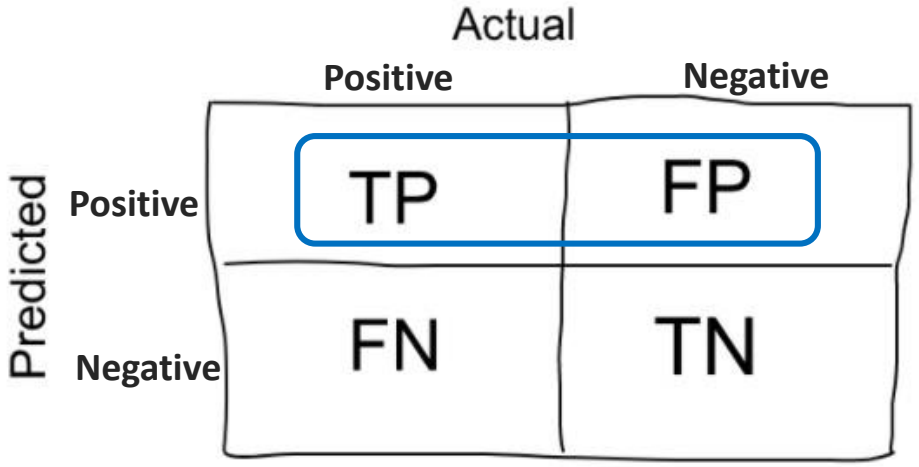
$$Accuracy = \frac{75}{100}$$

Evaluating Classification Models
Performance Evaluation - Precision

- Precision for a class is the number of true positives (i.e. the number of items correctly labelled as belonging to the positive class) divided by the total number of elements labelled as belonging to the positive class (i.e. the sum of true positives and false positives)
- Put another way, it is the number of correct positive predictions divided by the total number of positive class values predicted. It is also called the Positive Predictive Value (PPV).
  - Precision can be thought of as a measure of a classifiers exactness.
  - A low precision can also indicate a large number of False Positives
- In view of the example we have been following in class: Precision is a measure that tells us what proportion of people that have been allowed entry, should actually have been allowed entry.
- Precision score of 1.0 for a class means that every item labelled as belonging to that class does indeed belong to the class.

Evaluating Classification Models
Performance Evaluation - Precision

$$Precision = \frac{TP}{TP + FP}$$



There are 100 people whose temperature was checked.
45 were correctly allowed entry
30 were correctly denied entry
15 were incorrectly allowed entry
10 were incorrectly denied entry
Find the precision of the model.

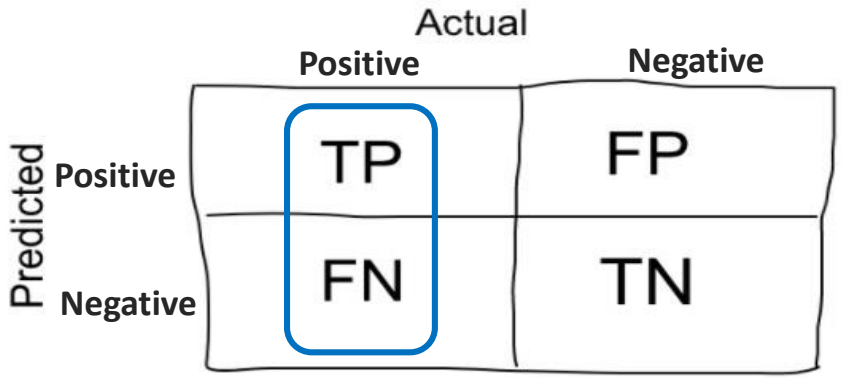
		Actual: Allow	Actual: Deny
Predicted	Allow	45	15
	Deny	10	30

$$Precision = \frac{45}{45 + 15}$$

$$Precision = \frac{45}{60} = 0.75$$

Evaluating Classification Models
Performance Evaluation - Recall

- Recall is the number of True Positives divided by the number of True Positives and False Negatives.
  - Put another way, it is the number of positive predictions divided by the number of positive class values in the test data.
  - It is also called Sensitivity or True Positive Rate.
  - Interpretation: for all the people who were allowed entry, recall tells us how many we correctly identified as people that should have been given entry.



$$Recall = \frac{TP}{TP + FN}$$

There are 100 people whose temperature was checked.
45 were correctly allowed entry
30 were correctly denied entry
15 were incorrectly allowed entry
10 were incorrectly denied entry
Find the recall of the model.

		Actual: Allow	Actual: Deny
Predicted	Allow	45	15
	Deny	10	30

$$Recall = \frac{45}{45 + 10}$$

$$Recall = \frac{45}{55} = 0.81$$

Evaluating Classification Models
Performance Evaluation – When to use Accuracy?

- Accuracy is a good measure when the target variable classes in the data are nearly balanced.
  - Stated simply: when the training data has enough information to learn the target output based on the provided data.
- Accuracy should NEVER be used as a measure when the target variable classes in the data are a majority of one class.

What is the accuracy of the model?

$$Accuracy = \frac{7}{8}$$ **87.5%**

Temperature	Entry
35°C	**Allow**
31°C	**Allow**
32°C	**Allow**
41°C	**Deny**
34°C	**Allow**
36°C	**Allow**
37°C	**Allow**
…	…
…	…

→ **entry='allow'** →

Temperature	Entry
33°C	**Allow**
32°C	**Allow**
35°C	**Allow**
36°C	**Allow**
40°C	**Allow**
35°C	**Allow**
34°C	**Allow**
32°C	**Allow**

Evaluating Classification Models
Performance Evaluation - F1 Score

- The F1 Score is measured using the formula:

$$F_1\ Score = 2 \times \frac{Precision\ \times Recall}{Precision + Recall}$$

- It is also called the F-Score or the F-Measure.
  - The F1 score conveys the balance between the precision and the recall.
  - F1 score is the harmonic mean of the precision and recall
  - Preferred metric when working with imbalanced sets.

Task:

- Using the provided confusion matrix of a sample model, calculate the following:

1. Accuracy
2. Precision
3. Recall
4. F-Score

		Actual	
		Positive	**Negative**
Predicted	**Positive**	61	12
	Negative	3	104

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

= (61+104) / 180

= 0.91

$$Precision = \frac{TP}{TP + FP}$$

= 61 / (61+12)

= 0.835

$$Recall = \frac{TP}{TP + FN}$$

= 61 / (61+3)

= 0.95

$$F_1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

*= 2 * (0.79325 / 1.785)*

= 0.88