Classification (Part 2)

Mohammed Brahimi & Sami Belkacem

Outline

- ■ Characteristics of Decision Trees
- ❏ Model Overfitting
- ❏ Model Evaluation and selection
- ❏ Conclusion

Characteristics of Decision Tree - Applicability

- **Nonparametric Approach**
  - No prior assumptions on data's probability distribution.
- **Wide Applicability**
  - Suitable for categorical and continuous datasets.
- **No Data Transformation**
  - Attributes can be used without binarization, normalization, or standardization.
- **Multiclass Problem Handling**
  - Handel multiclass without reducing them to binary tasks.
- **Interpretability**
  - Trained trees are easy to understand (particularly shorter ones).
- **Competitive Accuracy**
  - The result are comparable with other techniques for many simple data sets.

Characteristics of Decision Tree -Expressiveness

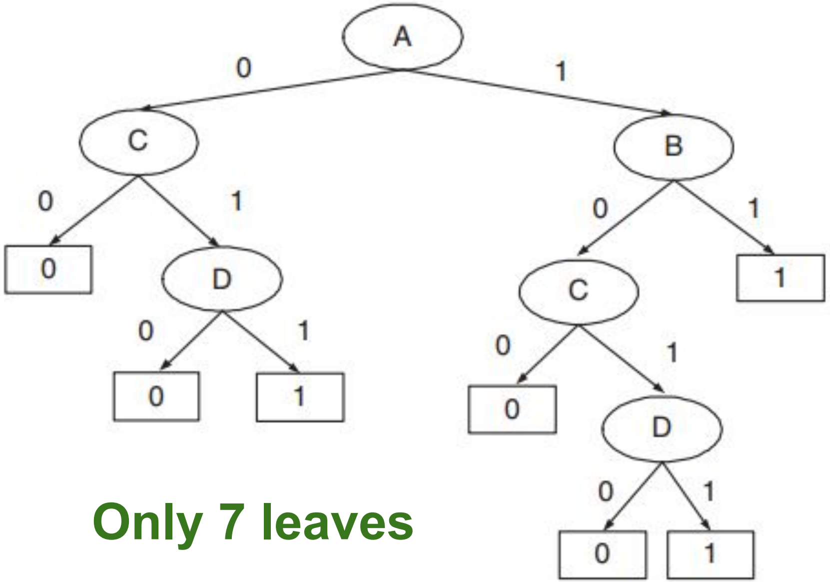
- **Universal Representation**
  - Tree can encode any function of discrete-valued attributes.
- **Efficient Encoding**
  - Discrete-valued function can be represented as an assignment table.
  - Decision tree can represent the assignment table efficiently.
  - Decision tree can group a combinations of attributes as leaf nodes.
- **Limitations**
  - Some functions, like the parity function, require a full decision tree for accurate modeling.

A	B	C	D	class
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Example of Compact Representation

Boolean function $(A \wedge B) \vee (C \wedge D)$ using a simpler tree with fewer leaf nodes, instead of a fully-grown tree.

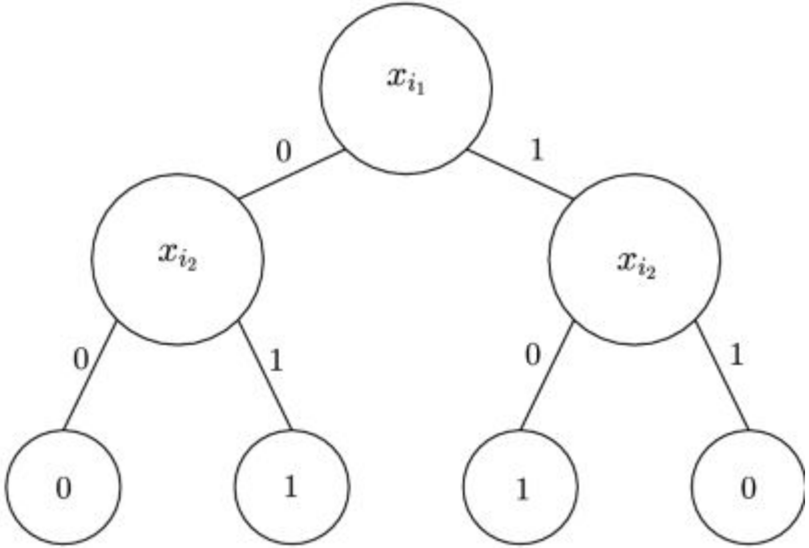
A	B	C	D	class
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1



Only 7 leaves

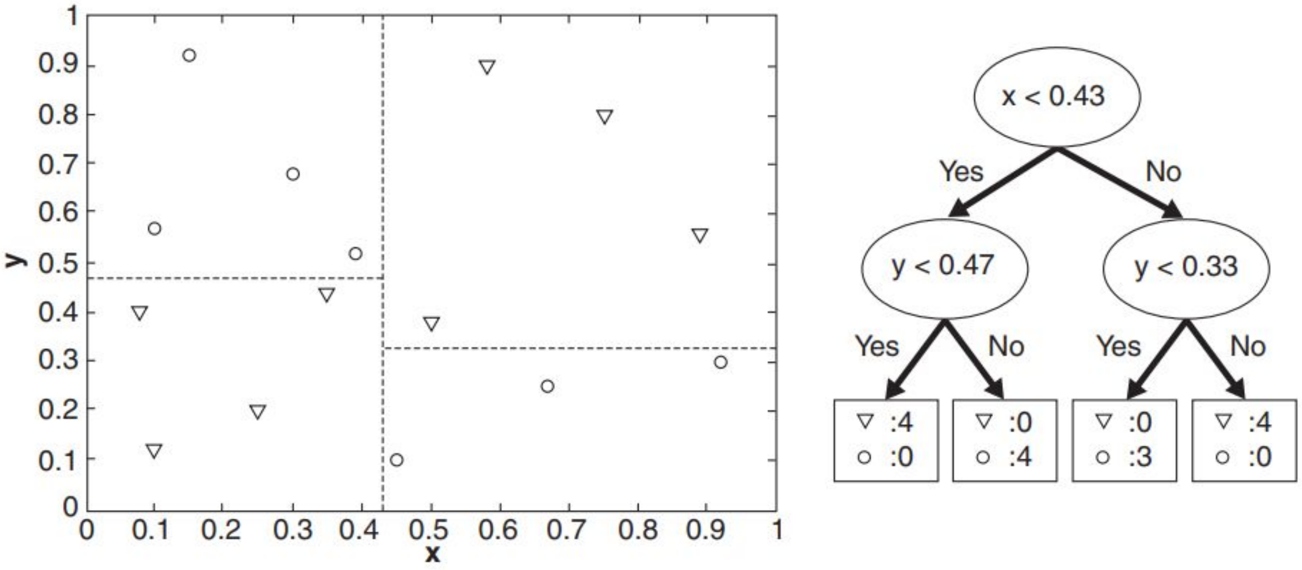
Example of Parity Representation

x	y	Parity
0	0	0
0	1	1
1	0	1
1	1	0



Corresponding Deterministic Decision Tree

Characteristics of Decision Tree - Rectilinear Splits



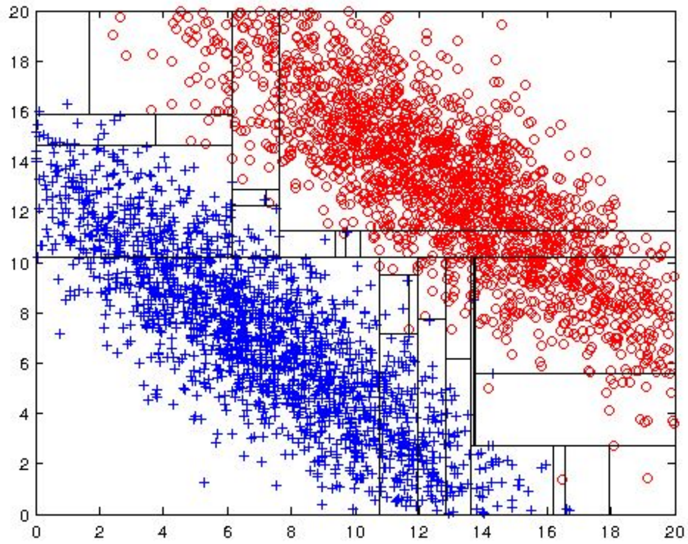
- Decision Trees use rectilinear splits to divide the data space.
- Simplifies complex multidimensional data into understandable segments.
- Effective in handling both categorical and continuous variables.

Characteristics of Decision Tree - Rectilinear Splits

What are the disadvantages of rectilinear splits ?

Characteristics of Decision Tree - Rectilinear Splits

Disadvantages of rectilinear splits



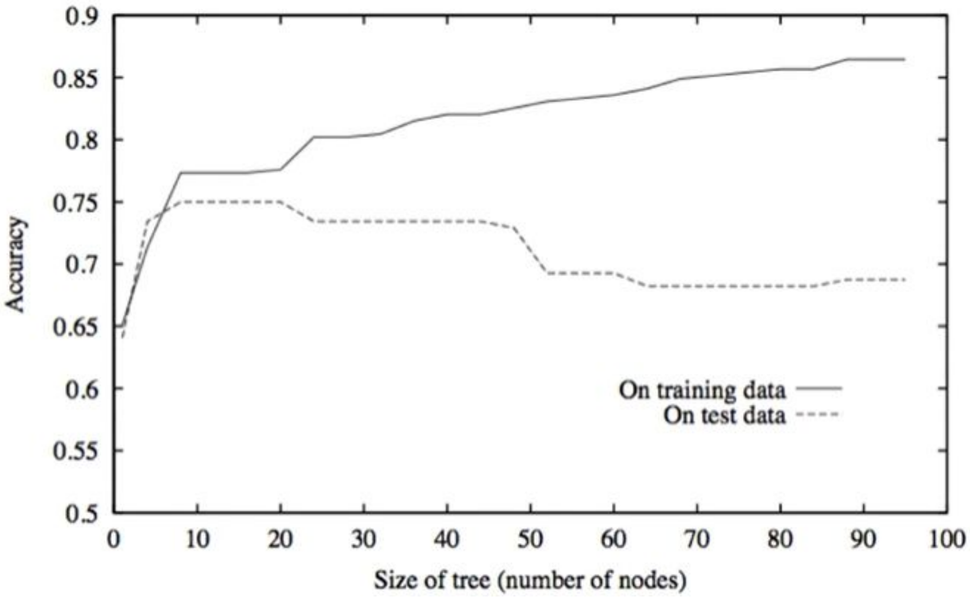
- **Struggle with Non-linear Boundaries:**
  - Ineffective in capturing complex, non-linear relationships in data.
- **Limited Flexibility:**
  - Restricts decision boundaries to orthogonal lines, limiting flexibility.
- **Oversimplification Risks:**
  - Can lead to oversimplified models that fail to capture the true nature of the data.

Outline

- Characteristics of Decision Trees
- Model Overfitting
- Model Evaluation and selection
- Conclusion

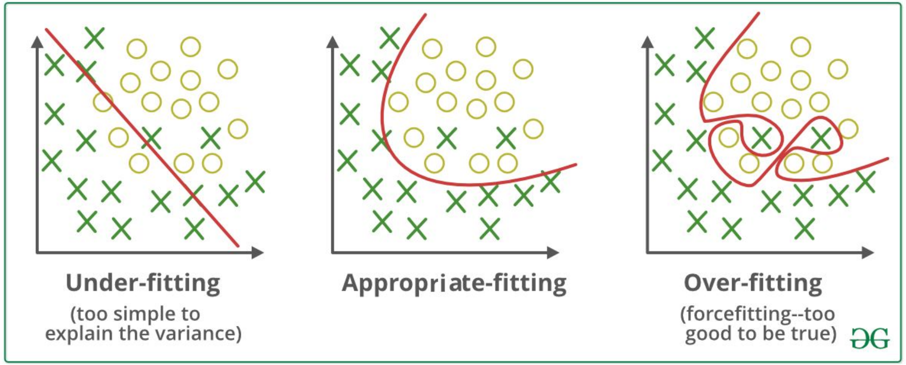
Model Overfitting

- Overfitting occurs when a model **fits training data too closely**, leading to poor generalization.
- A overfitted model may perform well on training data but poorly on test data.
- **Training vs Test Error:** As tree size increases, training error may decrease, but test error eventually increases.



Causes of Overfitting

- **Limited Training Size:**
  - A small training set may not represent true patterns, leading to overfitting.
- **High Model Complexity:**
  - Overly complex models can capture training-specific patterns, reducing generalizability.
- **Spurious Patterns Recognition:**
  - Models may learn irrelevant patterns present in training data (Ex. noise), which don't generalize to new data.

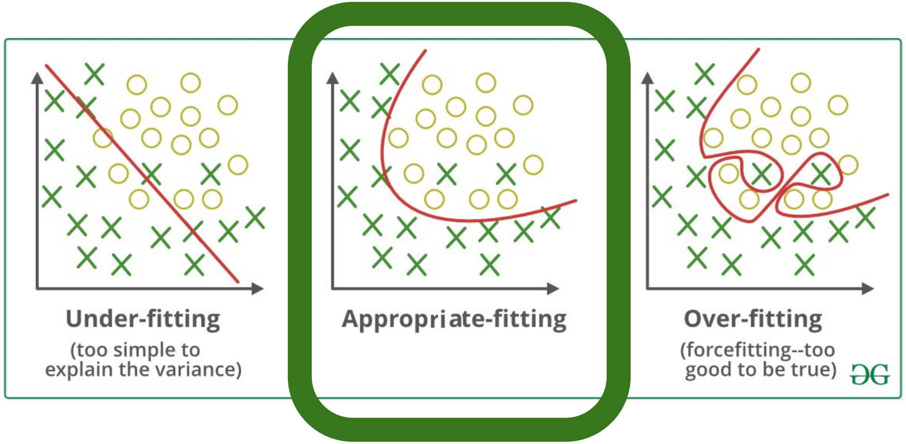


Overfitting vs Underfitting

Underfitting: Simple models may fail to capture essential patterns.

Data scientist challenge

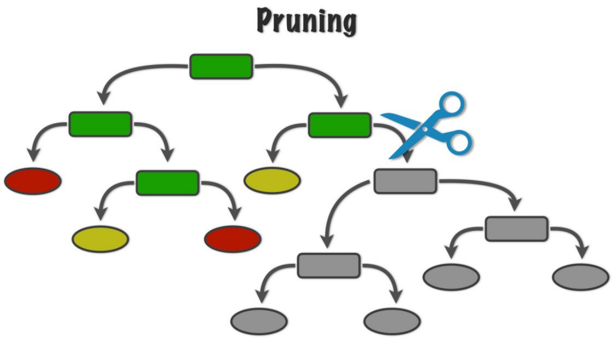
Find a model that does not overfit or underfit



Dealing with overfitting - Pruning

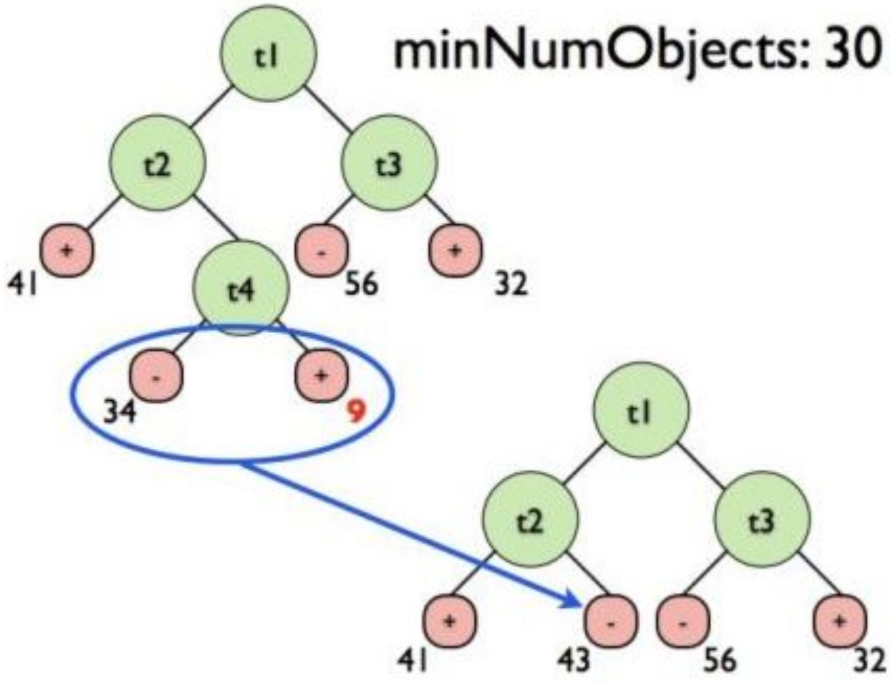
Pruning: cutting away branches that may be based on noisy or misleading data to prevent overfitting.

- **Pre-pruning:** Occurs during tree construction.
  - Limits tree growth by limiting the maximum depth or minimum leaf size.
  - Prevents overfitting by avoiding overly complex models.



- **Post pruning:** Applied after the tree is fully grown.
  - Removes branches that contribute little to classification accuracy.
  - Reduces model complexity, enhancing generalization to new data.

Example about pruning in decision tree



Example about pruning in decision tree

```
MultiAgent = 0:
| depth > 2:   class 0
| depth <= 2:
| | MultiIP = 1:   class 0
| | MultiIP = 0:
| | | breadth <= 6:   class 0
| | | breadth > 6:
| | | | RepeatedAccess <= 0.322:   class 0
| | | | RepeatedAccess > 0.322:   class 1
MultiAgent = 1:
| totalPages <= 81:   class 0
| totalPages > 81:   class 1
```

Max depth = 3

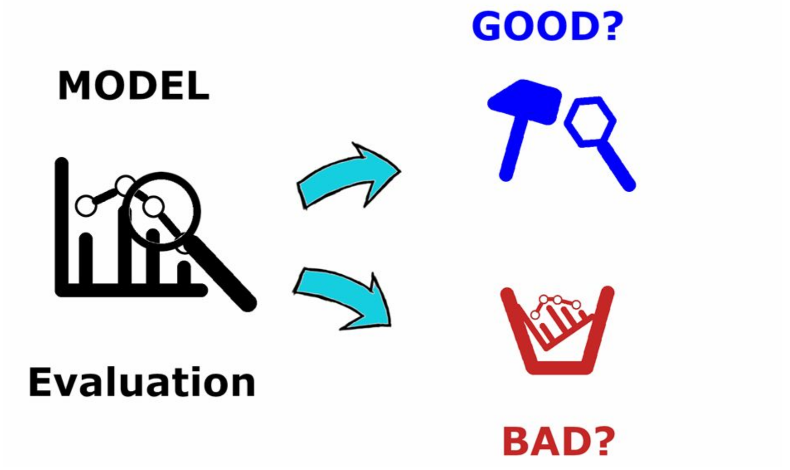
```
MultiAgent = 0:   class 0
MultiAgent = 1:
| totalPages <= 81:   class 0
| totalPages > 81:   class 1
```

Outline

- Characteristics of Decision Trees
- Model Overfitting
- Model Evaluation and selection
- Conclusion

Model Evaluation

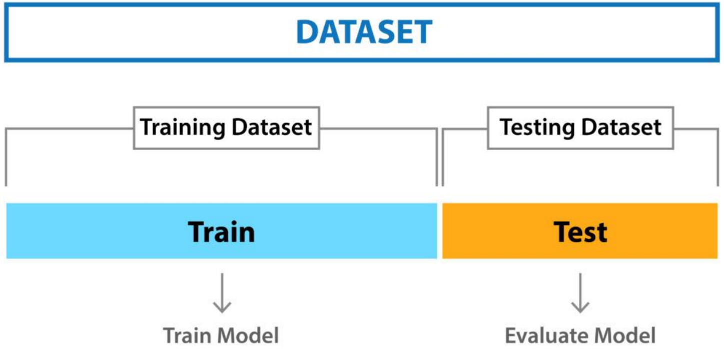
- **Objective:**
  - Estimate model performance on data not used during training.
  - Ensure robust model evaluation.
- **Labeled Test Set**
  - Utilize a separate test set, not involved in **model building**, for unbiased evaluation.
- **Holdout Method**
  - Randomly split data into training and test sets.
  - Use the test set to estimate generalization error.
- **Cross-Validation Method**
  - Divide data into multiple subsets; train and test the model on different subsets for a comprehensive performance estimate.



Holdout Method

Basic technique to partition data into training (D.train) and testing (D.test) sets.

- **Error Estimation**
  - Calculate error rate on **D.test (errtest)** as a measure of generalization error.
- **Data Proportion**
  - Analysts decide the split ratio.
  - Commonly two-thirds training and one-third testing.
- **Trade-offs**
  - Balancing D.train size for model learning and D.test size for reliable error estimation.
- **Repeated Holdout Method**
  - Enhances reliability by repeating the process and averaging error rates.



Model selection and validation set

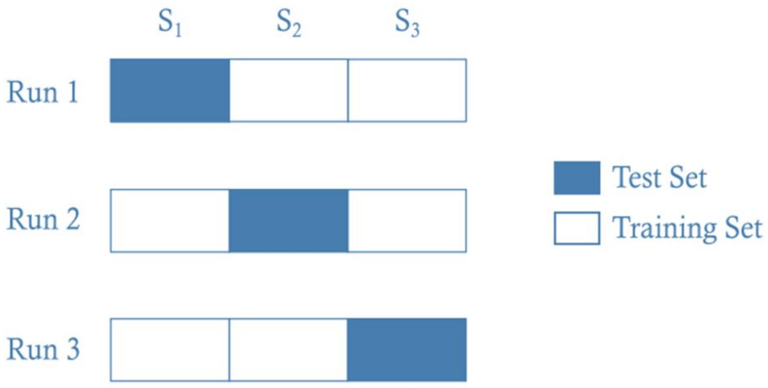
Achieve an optimal balance between <u>model complexity</u> and <u>performance</u>.

- **Complexity Measurement**
  - Complexity can be measured by the ratio of leaf nodes to training instances.
- **Limitation of Training Error:**
  - Training error rate is insufficient for effective model selection.
- **Validation Set:**
  - Essential for assessing generalization error.
- **Model Selection Strategy**:
  - Combine complexity with validation set performance to select the most effective model.

Cross validation

- Cross validation helps to avoid the split baise.
- Divide data into k equal folds.
- Each instance is used exactly once for error. calculation.
- The error is calculated based on:

$$err_{test} = \frac{\sum_{i=1}^{k} err_{sum}(i)}{N}$$



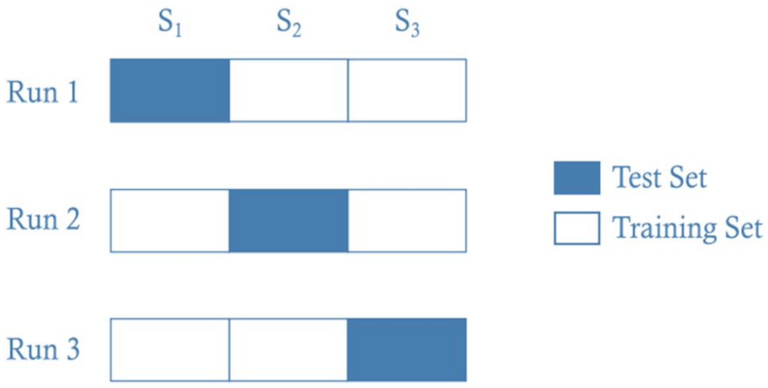
What if some classes don't appear in some folds?

Cross validation

- Cross validation helps to avoid the split baise.
- Divide data into k equal folds.
- Each instance is used exactly once for error calculation.
- The error is calculated based on:

$$err_{test} = \frac{\sum_{i=1}^{k} err_{sum}(i)}{N}$$

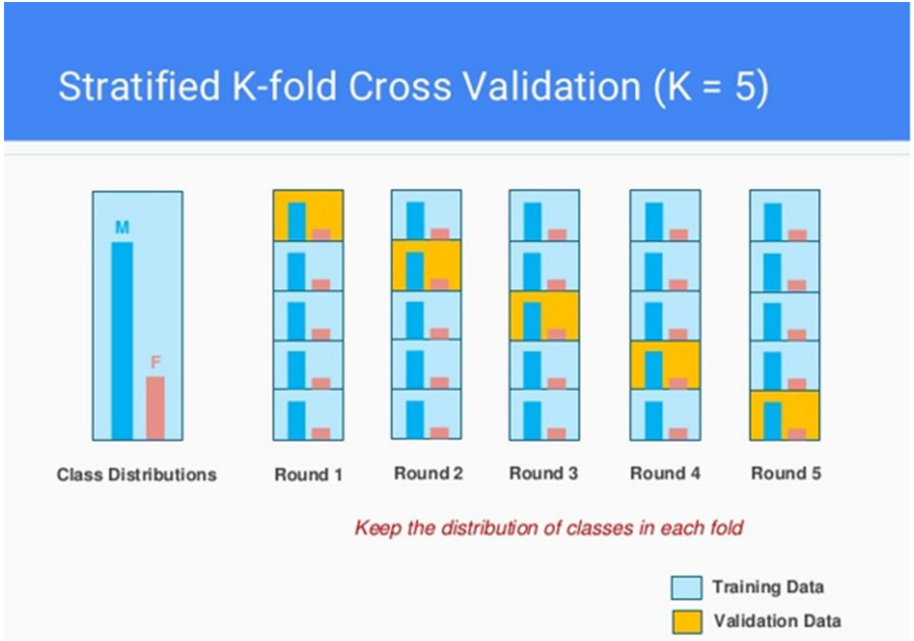
Number of errors in one fold.



What if some classes don't appear in some folds?

Cross validation

- **Stratified Sampling**
  - Ensures equal representation of classes in each partition.
- **Leave-One-Out Approach**
  - A special case where each instance is used once as a test set.
  - K = N
- **Estimating Error Variance**
  - Repeating cross-validation with different partitions provides robust error estimates.



Classification evaluation metrics

- **Accuracy**
  - Proportion of correctly predicted instances to total instances.
- **Precision**
  - Ratio of true positives to total predicted positives.
- **Recall (Sensitivity)**
  - Ratio of true positives to actual positives.
- **F1 Score**
  - Harmonic mean of precision and recall.
- **Confusion Matrix**
  - Visual tool categorizing true and false positives and negatives.

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

Classification evaluation metrics

		Predicted Class		
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Outline

- Characteristics of Decision Trees
- Model Overfitting
- Model Evaluation and selection
- Conclusion

Conclusion

Remember, the journey in data science is a continuous battle against overfitting and underfitting. Stay vigilant!