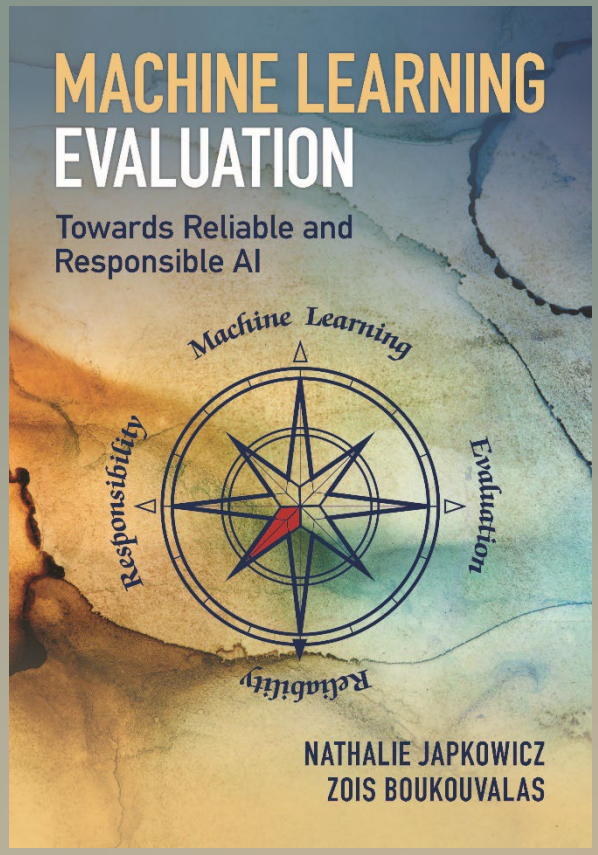
Machine Learning Evaluation

Implementing Responsible and Reliable AI

Nathalie Japkowicz
American University, Washington DC
Zois Boukouvalas
American University, Washington DC



IEEE Big Data 2024
Dec 15-18, 2024 @ Washington DC, USA

'By its nature, machine learning has always had evaluation at its heart. As the authors of this timely and important book note, the importance of doing evaluation properly is only increasing as we enter the age of machine learning deployment. The book showcases Japkowicz' and Boukouvalas' encyclopedic knowledge of the subject as well as their accessible and lucid writing style. Quite simply required reading for machine learning researchers and professionals.' **Peter Flach**, *University of Bristol*

Why does evaluation matter?

Why does evaluation matter?

One TikTok video in particular featured two people repeatedly pleading with the AI to stop as it kept adding more Chicken McNuggets to their order, eventually reaching 260. In a June 13, 2024, internal memo obtained by trade publication *Restaurant Business*, McDonald's announced it would end the partnership with IBM and shut down the tests.

In 2019, a study published in *Science* revealed that a healthcare prediction algorithm, used by hospitals and insurance companies throughout the US to identify patients in need of high-risk care management programs, was far less likely to flag Black patients.

Like many large companies, Amazon is hungry for tools that can help its HR function screen applications for the best candidates. In 2014, Amazon started working on AI-powered recruiting software to do just that. There was only one problem: The system vastly preferred male candidates. In 2018, **Reuters broke the news** that Amazon had **scrapped the project**.

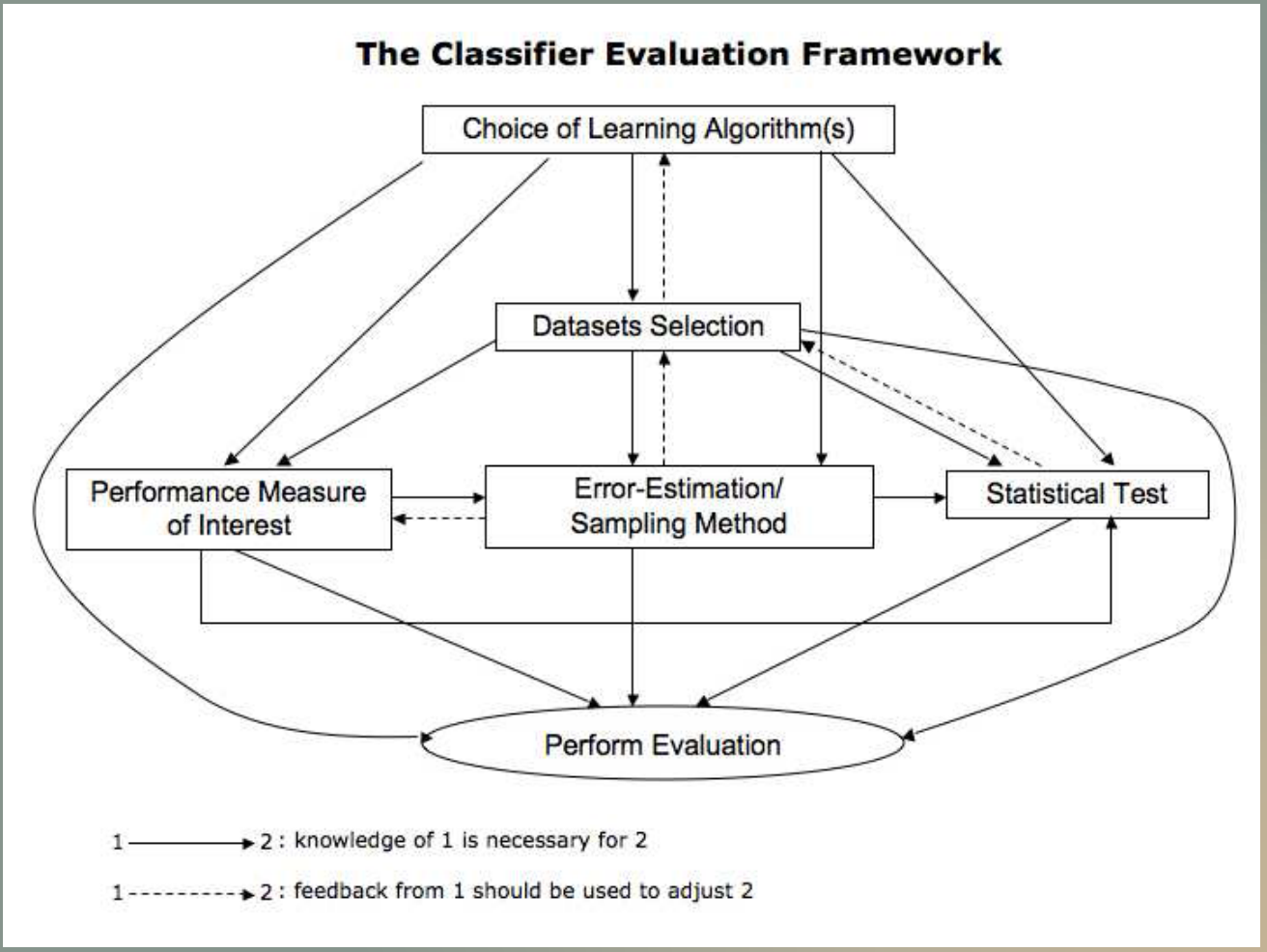
In March 2016, Microsoft learned that using Twitter interactions as **training data for ML** algorithms can have dismaying results.

Microsoft released Tay, an AI chatbot, on the social media platform, and the company described it as an experiment in "conversational understanding." The idea was the chatbot would assume the persona of a teenage girl and interact with individuals via Twitter using a combination of ML and natural language processing. Microsoft seeded it with anonymized public data and some material pre-written by comedians, then set it loose to learn and evolve from its interactions on the social network.

Within 16 hours, the chatbot posted more than 95,000 tweets, and those tweets rapidly turned overtly racist, misogynist, and anti-Semitic. Microsoft quickly suspended the service for adjustments and ultimately pulled the plug.

From: https://www.cio.com/article/190888/5-famous-analytics-and-ai-disasters.html

The main steps of classifier evaluation



What these steps depend on

These steps depend on the purpose of the evaluation:

- Comparison of a *new algorithm* to other (may be generic or application-specific) classifiers on a *specific domain* (e.g., when proposing a novel learning algorithm)
- Comparison of a *new generic algorithm* to other generic ones on a set of *benchmark domains* (e.g. to demonstrate general effectiveness of the new approach against other approaches)
- Characterization of *generic classifiers* on *benchmarks domains* (e.g. to study the algorithms' behavior on general domains for subsequent use)
- Comparison of *multiple classifiers* on a *specific domain* (e.g. to find the best algorithm for a given application task)

But these core aspects of evaluation are not all…

There also are:

- Safety and Data imperfections issues
- Safety and Algorithmic imperfections issues
- Safety and Platform and Implementation issues
- Data Bias
- Lack of Explainability
- Fairness issues
- Privacy and security issues
- Repeatability, reproducibility, and replicability issues

Going beyond Classification…

Though Classification is a core subdiscipline of machine learning, it is by no means the only one:

- Unsupervised Learning
- Regression
- Multi-Label Classification
- Image Segmentation
- Text Generation
- Time Series Analysis
- Data Stream Mining

Outline of the tutorial:

Topics:

- **Classification**
  - Choosing a performance measure
  - Sampling
  - Choosing a statistical test
- **Other ML Paradigms**
- **Machine Learning Deployment**
- **Reliability and Responsibility**

Note: In this tutorial, we present highlights from the book. The book, itself, contains many more topics and illustrations, all explored in greater depth.

Classification

Choosing a performance metric

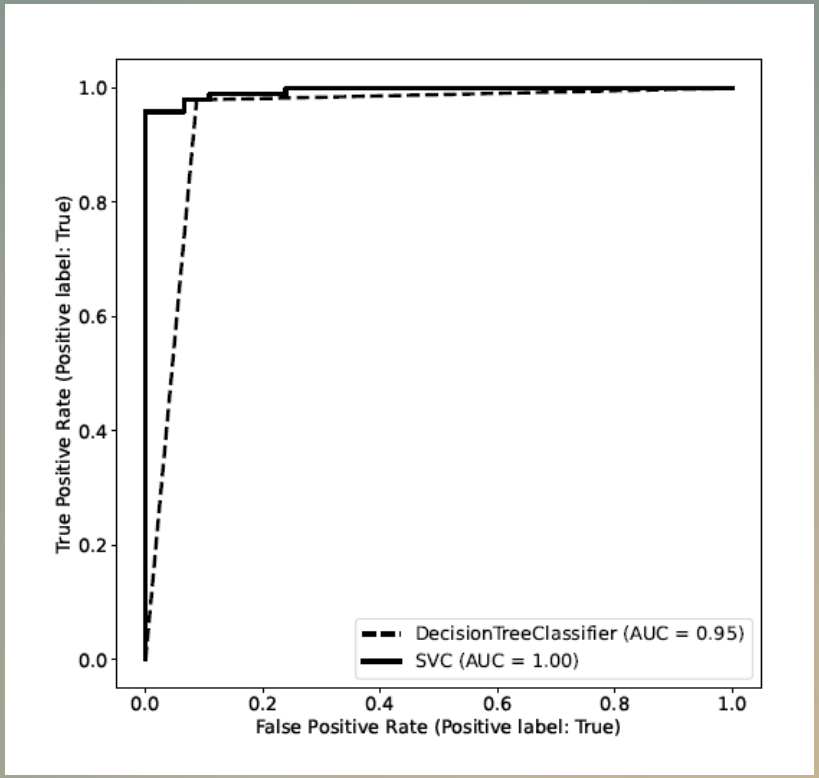
The confusion matrix

	Positive	**Negative**
Yes	True Positives (TP)	False Positives (FP)
No	False Negatives (FN)	True Negatives (TN)
Column Totals	P	N

Common evaluation metrics

- FP Rate = FP/N (False Alarm Rate)
- Precision = TP/(TP+FP)
- Accuracy = (TP+TN)/(P+N)
- TP Rate = TP/P = Recall = Hit Rate = Sensitivity
- F-Score = Precision * Recall (though a number of other formulae are also acceptable)

- ROC Analysis ➔ AUC



Do evaluation metrics agree?

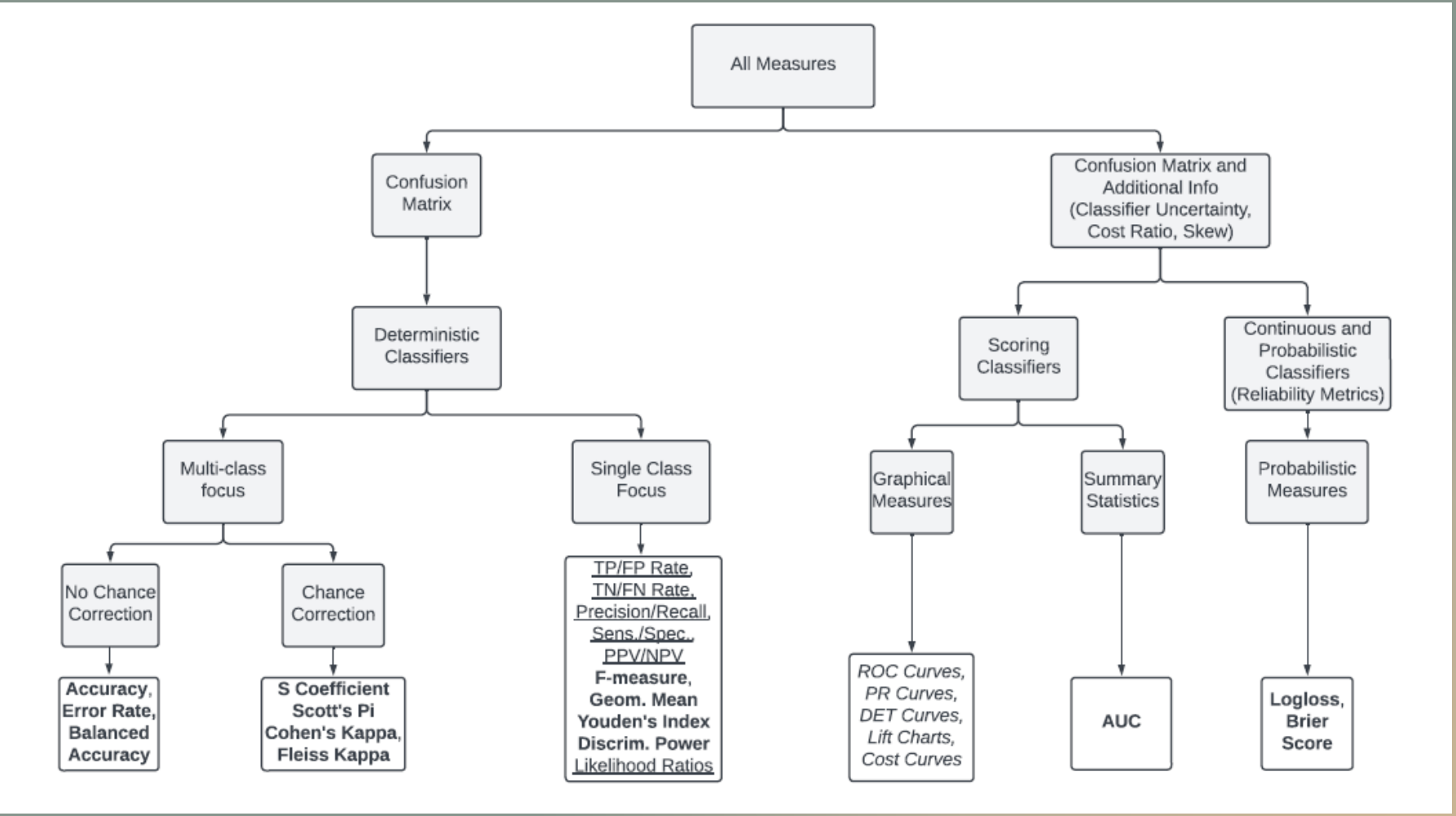
Algorithm	Acc	TPR	FPR	Prec	Rec	F	AUC
NB	.94	.87	.02	.98	.87	.92	.99
SVM	.91	.85	.04	.93	.85	.89	.97
DTree	.94	.91	.04	.93	.91	.92	.93
RandFor	.94	.91	.04	.93	.91	.92	.99
XGBoost	.98	.96	0	1	.96	.92	1
Bagging	.92	.80	0	1	.80	.89	.98

Table 5.1: A Study on the UCI Breast Cancer Domain

Algorithm	Acc	TPR	FPR	Prec	Rec	F	AUC
NB	.71	.49	.08	.84	.49	.62	.75
SVM	.64	.88	.58	.58	.88	.70	.71
DTree	.68	.76	.39	.64	.76	.69	.68
RandFor	.75	.79	.28	.72	.79	.75	.79
XGBoost	.78	.79	.22	.77	.79	.78	.81
Bagging	.71	.67	.25	.71	.67	.69	.70

Table 5.2: A Study on the UCI Liver Domain

Overview of classification metrics



The ambiguity of Accuracy

True class →	Pos	Neg
Yes	200	100
No	300	400
	P=500	N=500

True class →	Pos	Neg
Yes	400	300
No	100	200
	P=500	N=500

- Both classifiers obtain 60% Accuracy
- However, they exhibit very different behavior:
  - On the left: weak positive recognition rate/strong negative recognition rate
  - On the right: strong positive recognition rate/weak negative recognition rate

The ambiguity of Precision/Recall

True class →	Pos	Neg
Yes	200	100
No	300	400
	P=500	N=500

True class →	Pos	Neg
Yes	200	100
No	300	0
	P=500	N=100

- Both classifiers obtain the same precision and Recall values of of 66.7% and 40% (Note: the data sets are different)
- However, they exhibit very different behavior:
  - Same positive recognition rate of
  - Extremely different negative recognition rate: strong on the left / nil on the right

Correcting for Chance: Cohen's Kappa Measure -- Definition

- Agreement Statistics argue that accuracy does not take into account the fact that correct classification could be a result of coincidental concordance between the classifier's output and the label-generation process.
- Cohen's Kappa statistics corrects for this problem. Its formula is:

$$\kappa = (P_o - P_e^C) / (1 - P_e^C) \qquad \text{where}$$

  - P_o represents the probability of overall agreement over the label assignments between the classifier and the true process, and
  - P_e^C represents the chance agreement over the labels and is defined as the sum of the proportion of examples assigned to a class, times the proportion of true labels of that class in the data set.

Correcting for Chance: Cohen's Kappa Measure -- Illustration

Predicted -> Actual	A	B	C	Total
A	60	50	10	120
B	10	100	40	150
C	30	10	90	130
Total	100	160	140	

Accuracy = P_0 = (60 + 100 + 90) / 400 = 62.5%
P_e^C = 100/400 * 120/400 + 160/400 * 150/400 + 140/400 * 130/400 = 0.33875
κ = 43.29%

➔ Accuracy is overly optimistic in this example!

Sensitivity and Specificity

- Sensitivity and Specificity are two important metrics in the medical field but they are useful for other machine learning applications
- The metrics are typically used to assess the effectiveness of a clinical test in detecting a disease.
- Sensitivity tests how sensitive the test is to the presence of the disease, i.e., how many cases of the disease are successfully detected. (TP/P)
- Specificity reports the number of times the test result is negative for cases where the disease is not present (TN/N)
- Example: Sensitivity = 0.959 / Specificity = 0.957
  - There is a 4.1% chance that the patient will be denied treatment they need and a 4.3% chance that they will receive unnecessary treatment

Various Sensitivity/Specificity Combinations

- G-mean = $\sqrt{Sensitivity * Specificity}$
  - Helps detect issues of class imbalance, but does not penalize the result sharply unless there is an extremely low result for one of the metrics.
- Youden Index = Sensitivity – (1 – Specificity)
  - Helps avoid failure: Youden's Index compounds the misgivings of one metric with that of the other instead of allowing the success of one metric to compensate for the failure of the other. It is more pessimistic than the G-mean.
- Likelihood Ratios
  - LR+ = Sensitivity/(1-Specificity) LR- = (1- Sensitivity)/Specificity
  - LR+ indicates how many times more likely patients with a disease are to have a positive test than patients without the disease.
  - LR- indicates how many times less likely patients with a disease are to have a negative test than patients without the disease.

Other Metrics or Graphical Measures

- Deterministic Classifiers:
  - Balanced Accuracy
  - S-Coefficient
  - Scott's Pi
  - Fleiss Kappa
  - PPV/NPV
  - Discriminant Power
- Scoring Classifiers
  - Precision-Recall Curves
  - DET Curves
  - Lift Curves
  - Cost Curves
- Continuous and Probabilistic Classifiers
  - Log Loss
  - Brier Score

Recommendation

- Familiarize yourself with all the performance metrics available for your task.
- Understand the requirements of your domain.
- Match these requirements to the metrics and select a few candidates.
- Insure yourself that the candidate metrics are appropriate for the other steps of evaluation.

In short: choosing evaluation metrics for a task should not be done randomly. The researcher or practitioner should think logically about the implications of their choices!

Classification

Resampling

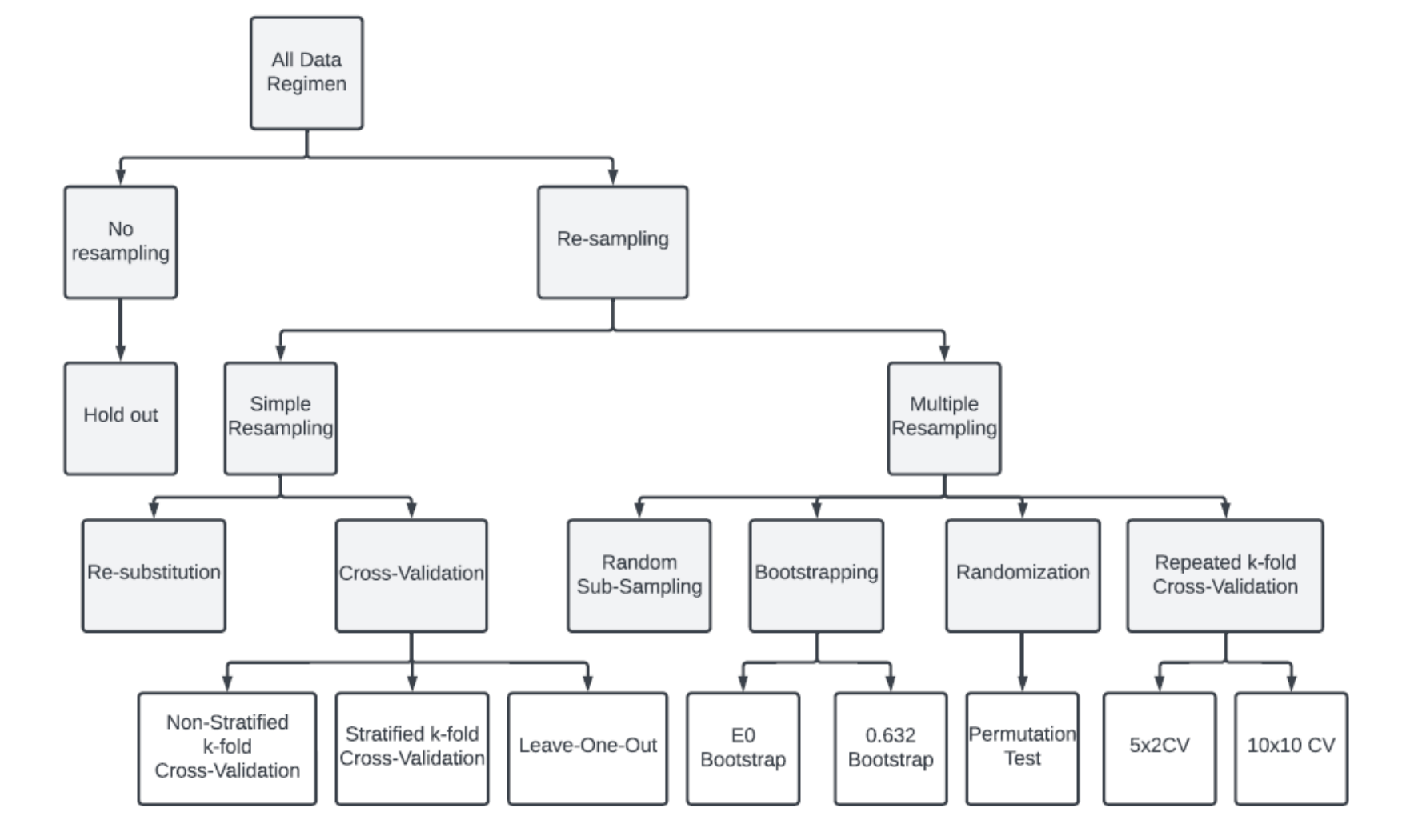
What is the purpose of resampling?

- Ideally, we would have access to the entire population or a lot of representative data from it.
- This is usually not the case, and the limited data available has to be re-used in clever ways in order to be able to estimate the error of our classifiers as reliably as possible, i.e., to be re-used in clever ways in order to obtain sufficiently large numbers of samples.
- Resampling is divided into two categories: *Simple re-sampling* (where each data point is used for testing only once) and *Multiple re-sampling* (which allows the use of the same data point more than once for testing)

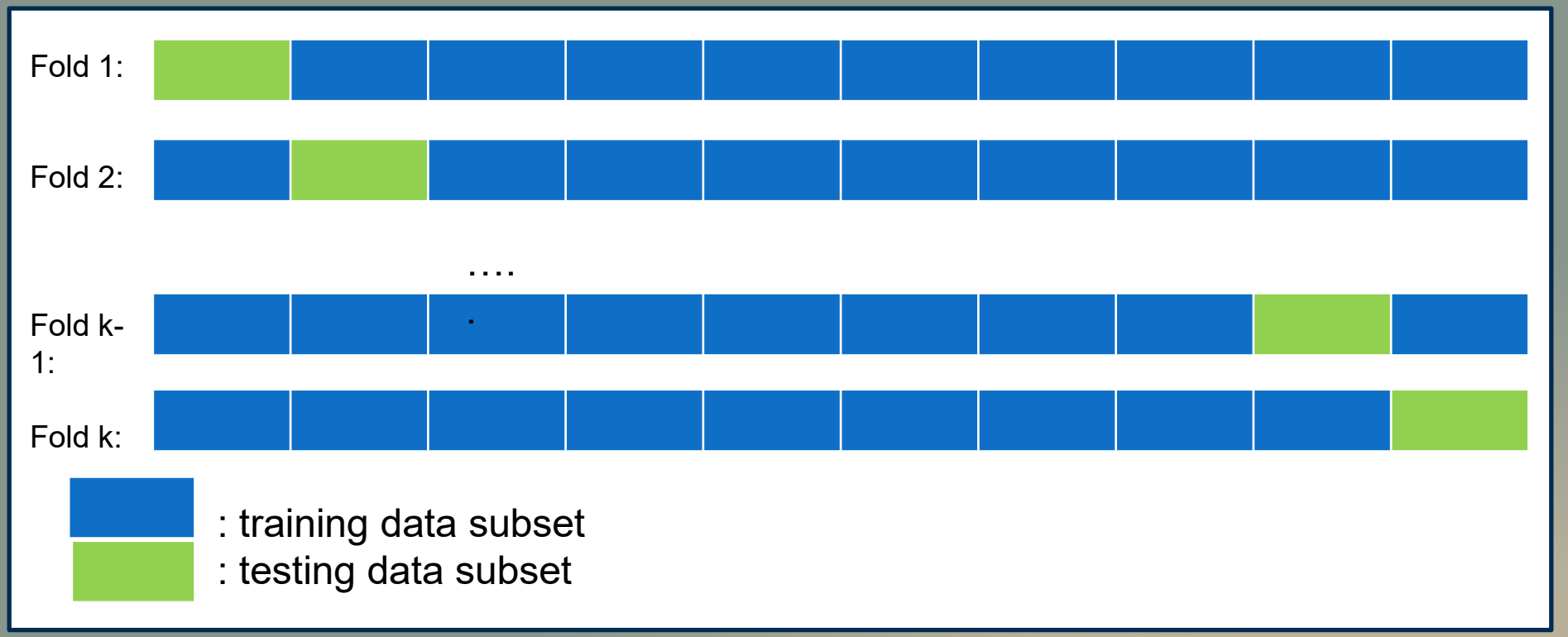
What are the dangers of resampling?

- Resampling is usually followed by statistical significance testing. Yet statistical significance testing relies on the fundamental assumption that the data used to obtain a sample statistics must be independent.
- However, if data is re-used, then this important independence assumption is broken, and the result of the statistical test risks being invalid.
- In addition to discussing a few re-sampling approaches, we will underline the issues that may arise when applying them.

Overview of resampling methods



K-fold crossvalidation



In Cross-Validation, the data set is divided into k folds and at each iteration, a different fold is reserved for testing and all the others, used for training the classifiers.

Some variations of crossvalidation

- Stratified k-fold Cross-Validation:
  - This variation is useful when the class-distribution of the data is skewed. It ensures that the distribution is respected in the training and testing sets created at every fold. This would not necessarily be the case if a pure random process were used.
- Leave one out:
  - In k-fold Cross-Validation, each fold contains m/k data points where m is the overall size of the data set. In leave one out, k =m and therefore, each fold contains a single data point.

Considerations about k-fold crossvalidation and its variants

- k-fold crossvalidation is the best known and most commonly used resampling technique.
- k-fold crossvalidation is less computer intensive than leave one out.
- The testing sets are independent of one another, as required by many statistical testing methods, but the training sets are highly overlapping. This can affect the bias of the error estimates.
- Leave one out produces error estimates with high variance given that the testing set at each fold contains only one example. The classifier is practically unbiased since each fold trains on almost all the data.

Bootstrapping

- Bootstrapping creates a large number of new training sets from the available dataset by drawing with replacement from that dataset.
- Bootstrapping is useful in practice when the dataset is too small for Cross Validation or Leave One Out approaches to yield a good estimate.
- There are two bootstrap estimates that are useful in the context of classification: the Є0 and the e632 Bootstrap.
- The Є0 bootstrap tends to be pessimistic because it is only trained on 63.2% of the data in each run. The e632 attempts to correct for this.

The Є0 and e632 Bootstraps

- Given a data set D of size m, we create k bootstrap samples B_i of size m by sampling from D with replacement (k is typically larger than 200).
- At each run, each of the k bootstraps represents the training set while the testing set is made up of a single copy of the examples from D that did not make it to B_i.
- At each run a classifier is trained and tested and $Є0_i$ represents the performance of the classifier at that run.
- Є0 represents the average of all the $Є0_i$'s.

$$e632 = 0.632 \times Є0 = 0.368 \times err(f)$$

where err(f) is the optimistically biased resubstitution error

Considerations about bootstrapping

- Bootstrapping yields better estimates than crossvalidation and its variants when the data set is very small.
- The results of bootstrapping in such cases was shown to have low variance.
- The Є0 bootstrap is a good estimator when the true error rate is very high.
- The e632 bootstrap is a good estimator on small data sets especially if the true error rate is small.
- Bootstrapping is a poor estimator for certain types of classifiers that do not benefit from the presence of duplicate instances. (E.g., k-NN with clones removed)

Repeated k-fold crossvalidation I

- In order to obtain more stable estimates of an algorithm's performance, it is useful to perform multiple runs of simple re-sampling schemes. This can also enhance replicability of the results.
- Two specific schemes have been suggested in the context of crossvalidation: 5x2 crossvalidation and 10x10 crossvalidation.
- k-fold crossvalidation does not estimate the mean of the difference between 2 learning algorithms properly. The mean at a single fold behaves better. This led Dietterich (1998) to propose the 5x2 crossvalidation regimen, in which 2-fold crossvalidation is repeated 5 times. Dietterich also proposed a statistical test for this case.

Repeated k-fold crossvalidation II

- Dietterich found that the paired t-test based on the 5x2 crossvalidation scheme had lower probability of issuing a type-I error but had less power than the k-fold crossvalidation paired t-test.
- Alpaydyn (1999) proposed to substitute the t-test at the end of the 5x2CV scheme proposed by Dietterich by an F-test. That test has an even lower chance of issuing a type-I error and has increased power.
- Bouckahert (2003) proposed several variations of a 10x10 crossvalidation scheme. Generally speaking, these schemes show a higher probability of Type-I error than 10-fold crossvalidation, but higher power.

A few remarks about averaging

- When considering resampling methods, it is usually necessary to average the results obtained on different trials.
- We distinguish between different kinds of averaging:
  - Micro Averaging: calculate metrics globally by counting the total true positives, false negatives, and false positives.
    - The performances of the large classes weigh more than those of the small classes.
  - Macro Averaging: calculate metrics for each label and find their unweighted mean.
    - Large and small classes contribute equally.

Recommendations

- While k-fold crossvalidation is often a good choice for resampling, it is useful to keep its variants and bootstrapping in mind in case where the dataset suffers from class imbalances (in those cases, consider stratified crossvalidation) or is particularly small (in those cases, consider leave one out or bootstrapping).
- Repeated k-fold crossvalidation should also be considered to increase result stability and, possibly, replicability. However, since the repetition takes away the simple resampling property of crossvalidation and its variants, the consequences on the subsequent statistical tests should also be given some attention.

Classification

Choosing a statistical test

The purpose of Statistical Significance Testing

- Performance metrics allow us to make observations about different classifiers, and resampling strategies attempt to decrease the chances of basing our conclusions on unrepresentative data.
- The question we ask now is related to the risk we are taking in trusting the results. In particular, we ask: can the observed results be attributed to real characteristics of the classifiers under scrutiny or are they observed by chance?
- The purpose of statistical significance testing is to help us gather evidence of the extent to which the results returned by an evaluation metric are illustrative of the general behavior of our classifiers.

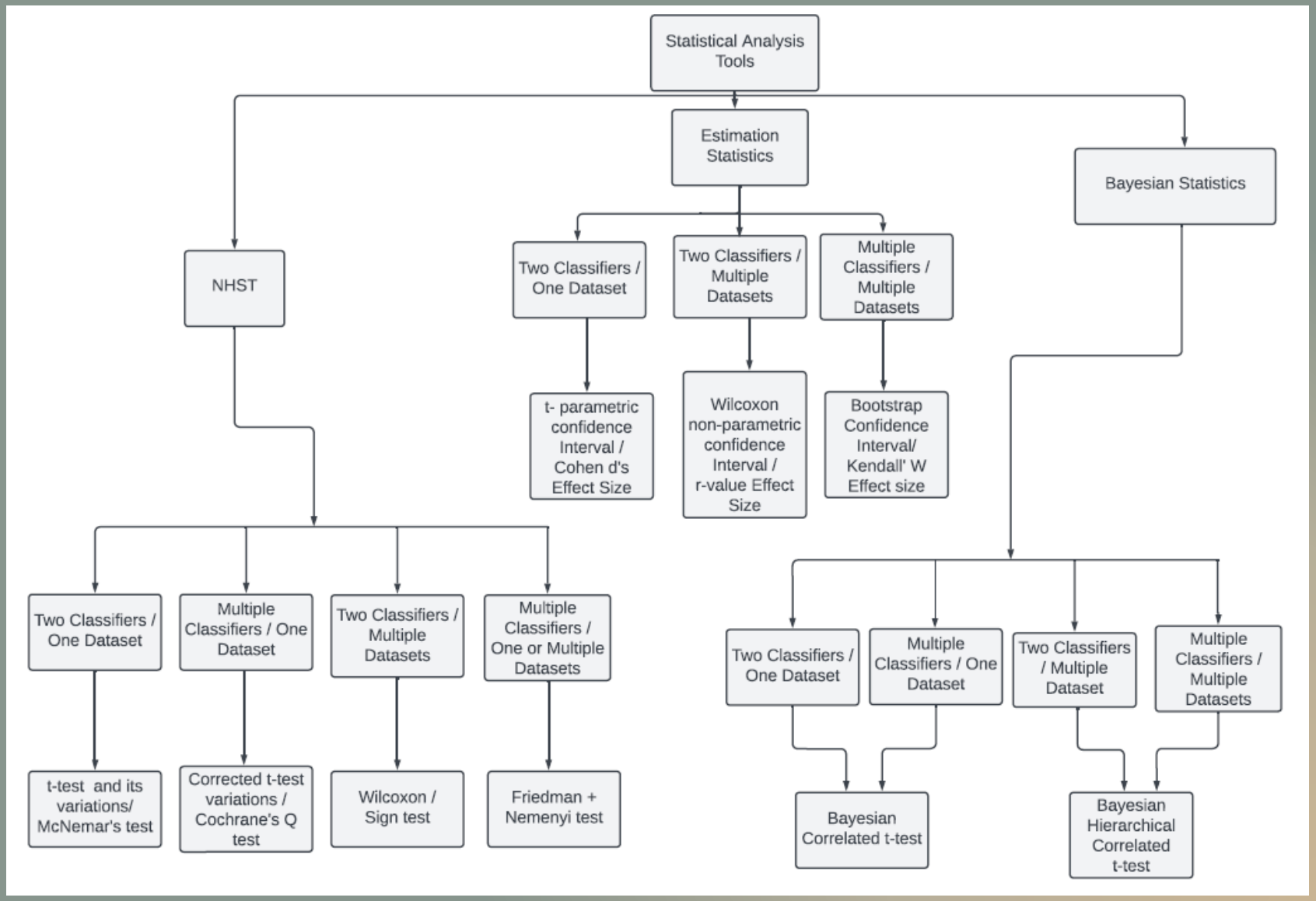
Two Statistical Paradigms

- Frequentist statistics:
  - Null statistical hypothesis testing (NSHT)
  - Estimation statistics (confidence intervals, effect size, power)
- Bayesian statistics
- Assessment:
  - NHST is a mixture of two incompatible approaches: Fisher/Neyman & Pearson
  - NHST augmented with estimation statistics improves upon NHST alone.
  - Bayesian statistics is coherent and answers the practical questions researchers seek answers for.
  - Bayesian statistics relies on priors which are hard to estimate

How to choose a statistical test?

- There are several aspects to consider when choosing a statistical test.
  - What kind of problem is being handled?
  - Whether we have enough information about the underlying distributions of the classifiers' results to apply a parametric test or whether we have to rely on a non-parametric test in the Frequentist paradigm; alternatively, whether we have enough information about the priors to apply a Bayesian test?
- Regarding the type of problem, we distinguish between:
  - The comparison of two algorithms on a single domain
  - The comparison of several algorithms on a single domain
  - The comparison of two algorithms on several domains
  - The comparison of multiple algorithms on multiple domains

Statistical tests overview



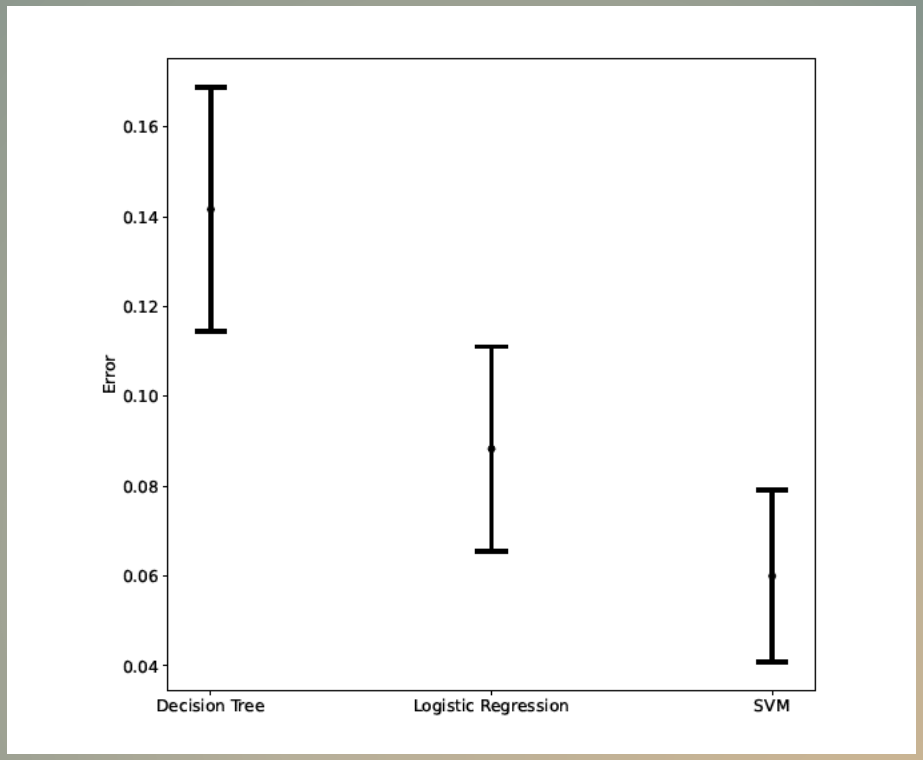
Two Classifiers/One Domain: Frequentist, non-parametric test: McNemar's test

- McNemar's test is the non-parametric counterpart of the t-test. It relies on four values observed on the test set:
  - The number of instances misclassified by both classifiers (c_{00})
  - The number of instances misclassified by f1 but correctly classified by f2 (c_{01})
  - The number of instances misclassified by f2 but correctly classified by f1 (c_{10})
  - The number of instances correctly classified by both classifiers (c_{11})
- The McNemar χ^2 statistics is given by $\chi^2_{McNemar} = \frac{(|c_{01}^{Mc} - c_{10}^{Mc}| - 1)^2}{c_{01}^{Mc} + c_{10}^{Mc}}$
- If $c_{01} + c_{10} \geq 20$ then χ^2_{MC} is compared to the χ^2 statistics:
  - If χ^2_{MC} exceeds the χ^2 statistics, then we can reject the null hypothesis that assumes that f1 and f2 perform equally well with 1-α confidence
- If $c_{01} + c_{10} < 20$, then the test cannot be used and the sign test should be used instead.

Two Classifiers/One Domain: Frequentist, Estimation Statistics: Confidence Intervals

$$CI^{R(\mathrm{DT})}_{Lower} = \overline{R}(\mathrm{DT}) - Z_P \times \frac{\sigma(x)}{\sqrt{|S_x|}}$$
$$= 0.149 - 1.96 \times \frac{0.136}{\sqrt{100}}$$

$$CI^{R(\mathrm{DT})}_{Upper} = \overline{R}(\mathrm{DT}) + Z_P \times \frac{\sigma(x)}{\sqrt{|S_x|}}$$
$$= 0.149 + 1.96 \times \frac{0.136}{\sqrt{100}}$$



Two Classifiers/One Domain: Frequentist, Estimation Statistics: Effect Size

- Statistical tests can determine whether a difference between classifiers is significant, but not whether it is of practical importance.
- Statistical Significance is known as the *effect*, and practical relevance is obtained by measuring the *size* of this effect.
- Cohen's d statistic is the most appropriate measurement of effect size for two classifiers and one domain. It is calculated as:
  - $d_{cohen} = \frac{(\overline{pm}(f_1) - \overline{pm}(f_2)}{\sigma_p}$

where σ_p is the pooled standard deviation and is defined as:

- $\sigma_p = \sqrt{\frac{\sigma_1^2 + \sigma_2^2}{2}}$

where σ_1^2 and σ_2^2 are the variances of pm(f1) and pm(f2), respectively.

Two Classifiers/One Domain: Bayesian Statistics: the correlated t-test I

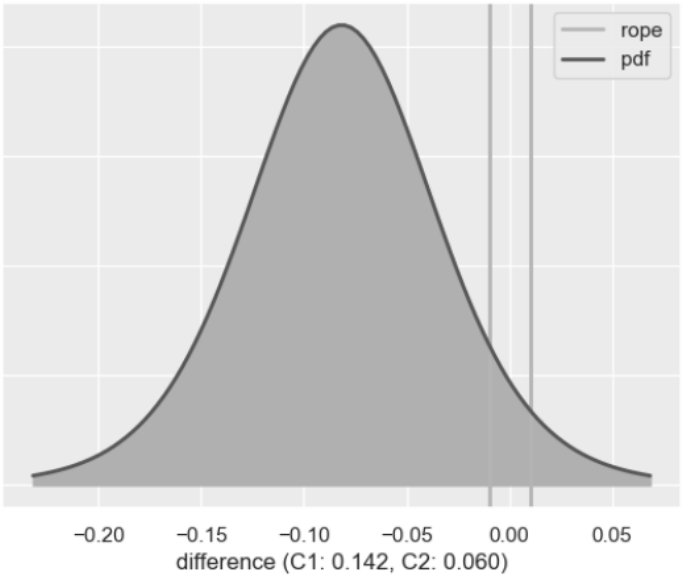


Figure 7.6: Positioning of the ROPE in the Labor Dataset

- Accuracy(DecisionTree) – Accuracy(SVM) on the UCI labor dataset
- The ROPE is small and the difference falls mostly on the negative side
- SVM's performance is most often superior to that of DecisionTree.

Two Classifiers/One Domain: Bayesian Statistics: the correlated t-test II

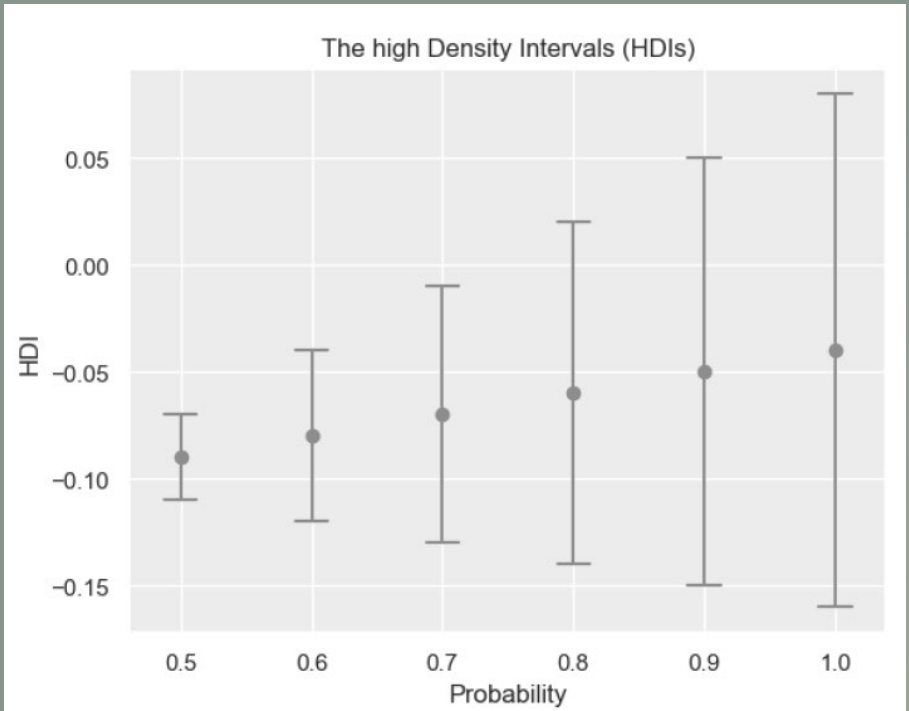


Figure 7.7: HDIs corresponding to Figure7.6

- High Density Intervals show the intervals where 50%, 60%, etc. of the data is concentrated.
- They can be interpreted in a way similar to the way Frequentist confidence intervals are.

Two Classifiers/Multiple Domains: Frequentist, non-parametric test: Wilcoxon's signed ranked test

Data	NB	SVM	NB-SVM	|NB-SVM|	Ranks	± Ranks
1	.9643	.9944	-0.0301	0.0301	3	-3
2	.7342	.8134	-0.0792	0.0792	6	-6
3	.7230	.9151	-0.1921	0.1921	8	-8
4	.7170	.6616	+0.0554	0.0554	5	+5
5	.7167	.7167	0	0	Remove	Remove
6	.7436	.7708	-0.0272	0.0272	2	-2
7	.7063	.6221	+0.0842	0.0842	7	+7
8	.8321	.8063	+0.0258	0.0258	1	+1
9	.9822	.9358	+0.0464	0.0464	4	+4
10	.6962	.9990	-0.3028	0.3028	9	-9

W_{S1} = 17 and W_{S2} = 28 ➔ T_{Wilcox} = min(17, 28) = 17
For n= 10-1 degrees of freedom and α = 0.005, V = 8 for the 1-sided test. V must be larger than T_{Wilcox} in order to reject the hypothesis. Since 17 > 8, we cannot reject the hypothesis that NB's performance is equal to that of SVM at the 0.005 level.

Two Classifiers/Multiple Domains: Bayesian statistics: Bayesian hierarchical correlated t-test

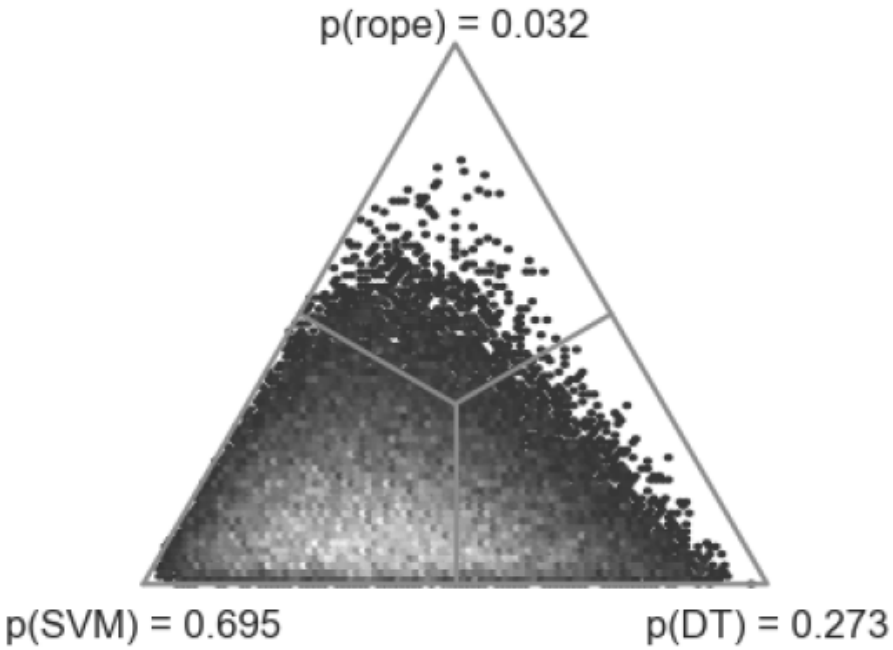


Figure 7.8: Visualizing the result of the Bayesian hierarchical correlated t-test comparing two algorithms (Decision Trees versus Support Vector Machines) over ten domains. The results show the probabilities with which SVM is superior to DT (69.5%), DT is superior to SVM (27.3%) and the two classifiers are equivalent (3.2%).

Multiple Classifiers/Multiple Domains: Frequentist, non-parametric statistics: Friedman's test

Domain	Classifier fA	Classifier fB	Classifier fC
1	85.83	75.86	84.19
2	85.91	73.18	85.90
3	86.12	69.08	83.83
4	85.82	74.05	85.11
5	86.28	74.71	86.38
6	86.42	65.90	81.20
7	85.91	76.25	86.38
8	86.10	75.10	86.75
9	85.95	70.50	88.03
19	86.12	73.95	87.18

Domain	Classifier fA	Classifier fB	Classifier fC
1	1	3	2
2	1.5	3	1.5
3	1	3	2
4	1	3	2
5	2	3	1
6	1	3	2
7	2	3	1
8	2	3	1
9	2	3	1
10	2	3	1
$R_{.j}$	15.5	30	14.5

$\chi_F^2 = [\frac{12}{10 \times 3 \times (3+1)} \times \sum_{j=1}^{3} (R_{.j})^2] - 3 \times 10 \times (3+1) = 15.05$ For a 2-tailed test at the 0.05 level of significance, the critical value is 7.8. $\chi_F^2 > 7.8$, i.e., Rejection of the NH.

Multiple Classifiers/Multiple Domains: Frequentist, non-parametric statistics: Nemenyi's test

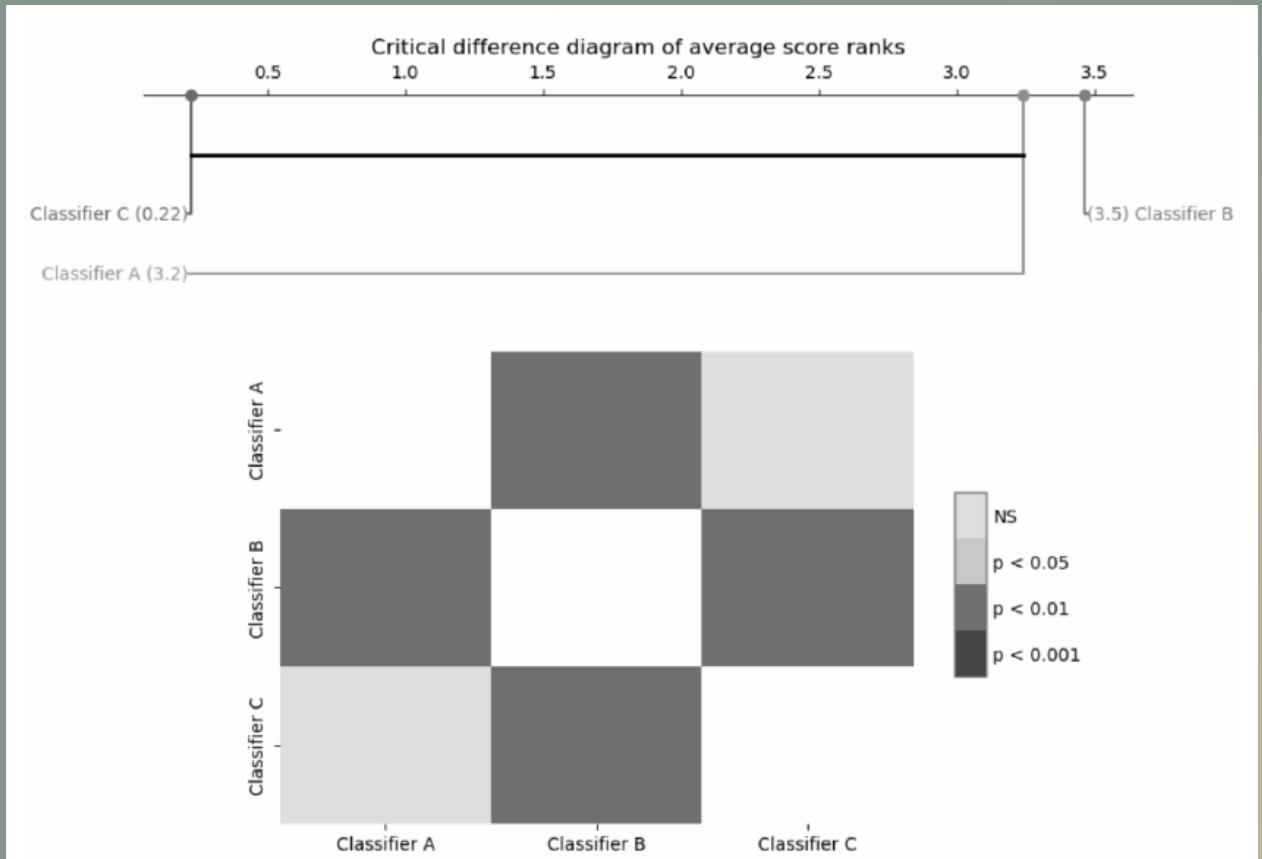


Figure 7.2: Critical Differences Diagrams for the above example

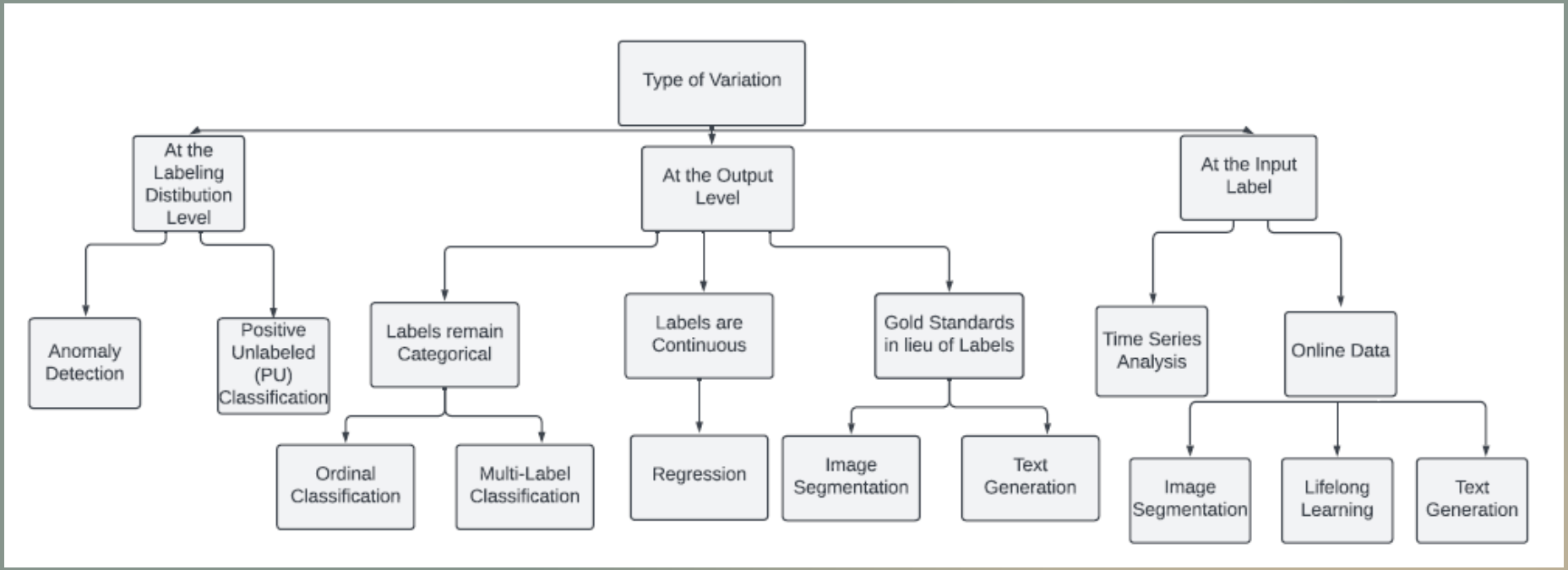
Recommendations

- Familiarize yourself with all the statistical methods available for the specific settings of your task.
- Match the settings with the most appropriate methods.
- Ensure that your choice does not violate any constraints and that if it does, you fully understand the consequences of that violation.
- Ensure that you understand the limitations of the statistical methods you selected so as not to make exaggerated claims.

Machine learning settings other than classification

Settings related to classification

Overview of supervised settings other than classification



Some examples: Regression I

The sum of squares error (SSE) is defined as follows:

$$SSE = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2.$$

The mean squared error (MSE):

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2.$$

The root mean squared error (RMSE), which is the square root of MSE:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2},$$

the mean absolute error (MAE), which sums the absolute errors made by the model:

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|,$$

Some examples: Regression II

instance number	1	2	3	4	5	6	7	8	9
True value	.1	.6	.3	.4	.6	.9	.8	.7	.2
Regress0 (perfect)	.1	.6	.3	.4	.6	.9	.8	.7	.2
Regress1 (close)	.15	.52	.31	.48	.58	.87	.8	.65	.23
Regress2 (flagrant outliers)	.1	0	.3	.4	.6	0	.8	.7	.2
Regress3 (milder outliers)	.1	.4	.3	.4	.6	.6	.8	.7	.2

Regressor	SSE	MSE	RMSE	MAE
Regress0	0	0	0	0
Regress1	.0201	.00223	.047	.029
Regress2	1.17	.13	.36	.17
Regress3	.13	.014	.12	.056

Some examples: multi-label classification I

inst#	Movie	Musical	Drama	Family	Fantasy
1	Annie	1	1	1	0
2	Harry Potter	0	0	1	1
3	West Side Story	1	1	0	0
4	The Godfather	0	1	0	0

- **Example-based metrics**: consider each test example and all its labels separately, assign a performance value to them and calculate the mean of the performance values obtained on the entire testing set.
- **Label-based metrics**: treat the testing set as *L* testing sets, one for each label where *L* is the total number of labels, use regular classification metrics on each “testing set", and aggregate the results using micro-, macro -or weighted- average.

Some examples: multi-label classification II

- **Exact Match Ratio**: $EMR = \frac{1}{n}\sum_{i=1}^{n} I(y_i = \hat{y}_i)$
- **0/1 Loss** = 1- EMR
- **Hamming Loss**: $HL = \frac{1}{n \times L}\sum_{i=1}^{n}\sum_{j=1}^{L} I(y_i^j \neq \hat{y}_i^j)$
- **Hamming Score**: $HS = \frac{1}{n}\sum_{i=1}^{n} \frac{|y_i \cap \hat{y}_i)|}{|y_i \cup \hat{y}_i)|}$

Example

inst#/label	l1	l2	l3	l4
1	1	1	1	0
2	0	0	1	1
3	1	1	0	0
4	0	1	0	0

Ground Truth

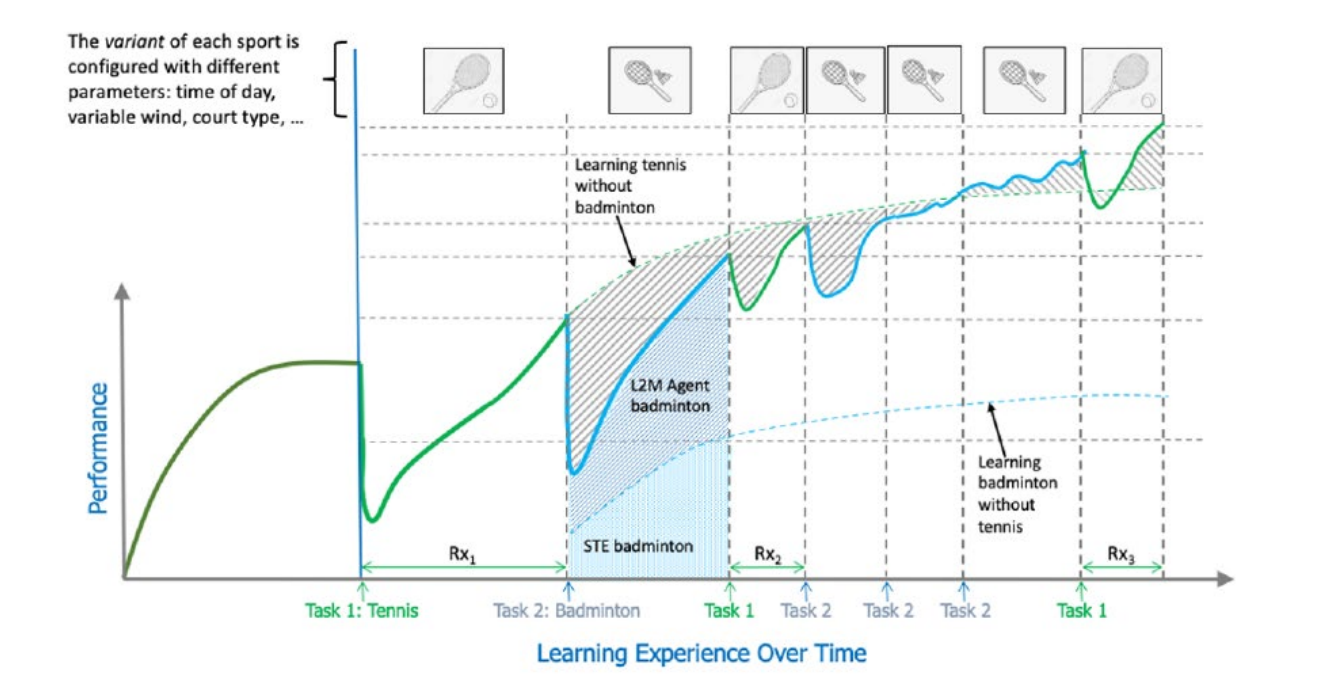
inst#/label	l1	l2	l3	l4
1	1	1	1	1
2	0	1	1	0
3	1	1	0	0
4	1	1	0	0

Predictions

Results:

- **EMR** = ¼ = .25 (only one instance matches perfectly;
- **0/1 Loss** = 3/4 = .75
- **HL** = 4/16 = .25
- **HS** = ¼ (3/4+1/3+2/2+1/2)= .646

Some examples: Lifelong Learning I



New, Alexander et al. "Lifelong Learning Metrics." *ArXiv* abs/2201.08278 (2022)

Challenges of Lifelong Learning

- Adaptation to new conditions [Adaptation]
- Catastrophic Forgetting Avoidance [Forgetting]
- Capacity Saturation Avoidance [Capacity]

Some examples: Lifelong Learning II
A special case: Lifelong Anomaly Detection

- **Lifelong ROC-AUC**: Average anomaly detection performance of the model after learning each task, for all previously learned tasks as well as the current task.
- **Backward Transfer for ROC-AUC (BWT)** measures the influence of learning new tasks on all previously learned tasks.
  - Positive values indicate that learning new tasks improves performance on previously learned ones.
  - Negative values indicate forgetting. Large negative values indicate catastrophic forgetting.
- **Forward Transfer for ROC-AUC (FWT)** measures the overall influence that learning each task has on the performance on tasks that will be learned in the future.

Single-task results are organized in matrix $R_{N\times N}$. Ri,j contains results in terms of the ROC-AUC metric obtained on task j after learning task i.

$$Lifelong\ ROC-AUC = \frac{\sum_{i\geq j}^{N} R_{i,j}}{\frac{N(N-1)}{2}}$$

$$BWT = \frac{\sum_{i=2}^{N}\sum_{j=1}^{i-1} R_{i,j} - R_{j,j}}{\frac{N(N-1)}{2}}$$

$$FWT = \frac{\sum_{i<j}^{N} R_{i,j}}{\frac{N(N-1)}{2}}$$

Machine learning settings other than classification

Unsupervised Learning

Metrics for Clustering: intrinsic versus extrinsic metrics

There are two principal types of measures to assess the clustering performance.

Intrinsic measures that do not require ground truth labels. Some of the clustering performance measures are Silhouette Coefficient, Calinski-Harabasz Index, Davies-Bouldin Index etc.

Extrinsic measures which require ground truth labels. Examples include Adjusted Rand index, Fowlkes-Mallows scores, Mutual information based scores, Homogeneity, Completeness and V-measure.

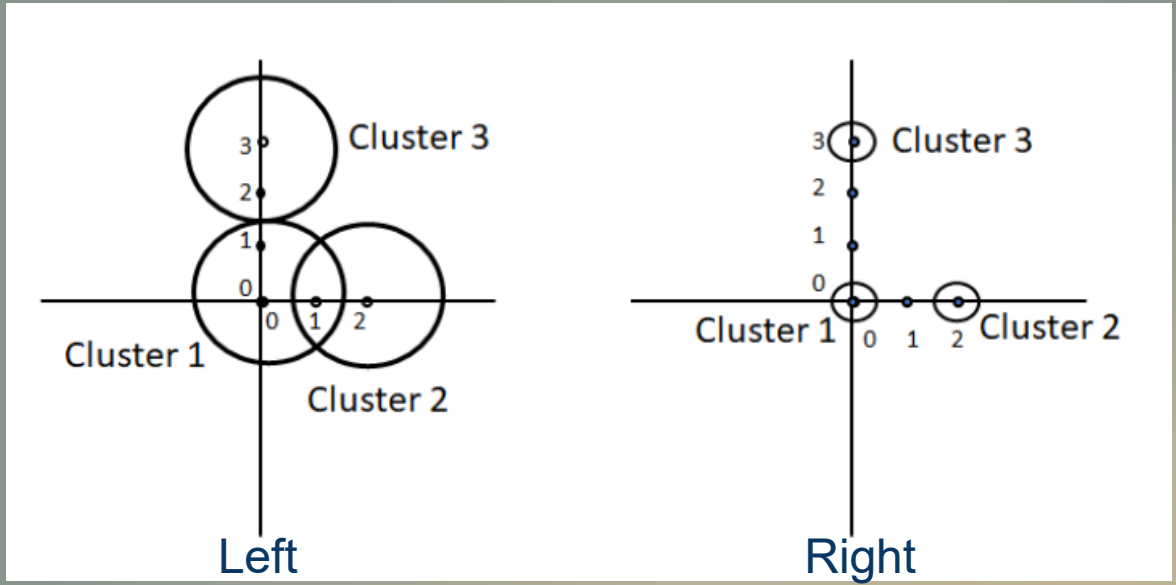
Exanple of an intrinsic metric: DBI

$$DBI = \frac{1}{l}\sum_{i=1}^{l} max_{j \neq i} \frac{s_i + s_j}{d(c_i, c_j)}$$

s_n = average distance of all points in cluster n

Assumptions:

Left: $s_1 = s_2 = s_3 = 1.5$
Right: $s_1 = s_2 = s_3 = 0.5$



Left :

$$DBI = \frac{1}{3}[\max(\frac{3}{2}, \frac{3}{3}) + \max(\frac{3}{2}, \frac{3}{3.6}) + \max(\frac{3}{3}, \frac{3}{3.6})] = \frac{1}{3}(\frac{3}{2} + \frac{3}{2} + \frac{3}{3}) = \frac{4}{3} = 1.33.$$

Right:

$$DBI = \frac{1}{3}[\max(\frac{1}{2}, \frac{1}{3}) + \max(\frac{1}{2}, \frac{1}{3.6}) + \max(\frac{1}{3}, \frac{1}{3.6})] = \frac{1}{3}(\frac{1}{2} + \frac{1}{2} + \frac{1}{3}) = \frac{4}{9} = 0.44.$$

Assessing GANs' performance: Inception Score (IS) and Frechet Inception Distance (FID)

$$IS = \exp\left(\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}}\left[D_{\text{KL}}(p(y|\boldsymbol{x})||p(y))\right]\right)$$

where:

- $\boldsymbol{x}$ represents an image from the real data distribution $p_{\text{data}}(\boldsymbol{x})$.
- $p(y|\boldsymbol{x})$ is the conditional class distribution given the image $\boldsymbol{x}$, typically estimated using an Inception model or a pre-trained classifier.
- $p(y)$ is the marginal class distribution, which can be approximated as the empirical distribution of labels in the training set.
- D_{KL} denotes the Kullback-Leibler divergence.

$$\text{FID} = \|\mu_{\text{real}} - \mu_{\text{fake}}\|^2 + \text{Tr}(C_{\text{real}} + C_{\text{fake}} - 2(C_{\text{real}}C_{\text{fake}})^{1/2})$$

where:

- μ_{real} and μ_{fake} represent the mean activation vectors (output of an intermediate layer) of the real and generated images, respectively.
- C_{real} and C_{fake} are the covariance matrices of the real and generated images, respectively.
- $\|\cdot\|$ denotes the Euclidean distance, and $\text{Tr}(\cdot)$ represents the trace of a matrix.

Machine Learning Deployment

Toward Deployment: Testing Standards in Software Engineering

Software development matured from experimental practice to an engineering field in the 1980s-1990s. Aim: to create reliable software systems for real-world applications

- Sub-Disciplines of Software Engineering
  - **Software Requirements**: Understand client needs and define software functionality
  - **Software Design**: Build conceptual architecture; define components and their interactions
  - **Software Development**: Implement system with basic testing and optimization
  - **Software Testing**: Ensure system functions as expected; identify and resolve bugs
  - **Software Maintenance**: Plan for future changes and adaptability

Toward Deployment: Testing Standards in Software Engineering

- **Types of Testing**
  - **White-Box Testing:** Examines internal code structure, targeting vulnerabilities
  - **Black-Box Testing:** Tests functionality without code knowledge; applied at unit, integration, or system levels
  - **Gray-Box Testing:** Combines both approaches for targeted testing

- **Challenges in Machine Learning Testing**
  - ML systems are non-deterministic, adding complexity to testing
  - Requires adapting traditional software testing standards to address unique ML challenges (e.g., unpredictability, performance variability)

Adapting Testing Standards: Non-Determinism in Machine Learning Systems

- **Data Dependence**: ML models are highly sensitive to the datasets they are trained on; variations in data can lead to different model outcomes. (e.g., Maximum-margin linear classifiers may yield different results based on training data.)
- **Algorithmic Bias**:
  - Removing specific biases increases variability (e.g., removing the maximum margin bias leads to infinite potential classifiers even within linear models).
- **Parameter Complexity**:
  - Linear Models: Fewer parameters, but still influenced by bias and data variability.
  - Nonlinear Models: High-variance behavior (e.g., KNN's data sensitivity, Neural Net's initialization and parameter randomness).
  - Result: Higher parameterization and complexity in ML systems generally increase non-deterministic outcomes.

Adapting Testing Standards: Testing ML Software – By Zhang et al. (2022)

Conditions for Testing:

- **Functional Requirements**:
  - *Correctness*: System should output expected results.
  - *Model Relevance*: Avoid underfitting/overfitting.
- **Non-Functional Requirements**:
  - Efficiency, robustness, fairness, security, privacy, interpretability.

Components to Test:

- *Data*: Check for noise, imbalance.
- *Learning Algorithm*: Ensure proper tuning, identify bugs.
- *Framework/Platform*: Detect issues in platforms like Scikit, Keras.

Testing Workflow:

- **Offline Testing**: Formal assessments and model validation.
- **Online Testing**: Real-time monitoring, user feedback, A/B testing.

Handling imperfections

Data Challenges in ML Systems:

- **Data Access Constraints**: Privacy laws (e.g., HIPAA) and NDAs limit access to training data.
- **Central Role of Data in ML**:
  - Sufficient sample size?
  - Correct data representation?
  - Noise-free data?
  - Overlap in modeled populations?
- **Types of Bias** *(Mehrabi et al., 2021)*:
  - **Historical Bias**: Algorithms reflect societal inequalities (e.g., COMPASS, Amazon recruitment).
  - **Representation Bias**: Under- or over-representation of certain groups (e.g., Buolamwini and Gebru's study on facial recognition misclassifying dark-skinned individuals).
  - Basically, the **Class Imbalance** problem!

Handling imperfections: Data Management & Mitigation Techniques

Technical Approaches to Mitigate Bias & Data Issues *(Roh et al., 2021)*:

- **Data Acquisition**: Data sharing (e.g., Kaggle, Data Lakes) and data search (e.g., GOODS).
- **Data Augmentation**: Adding external knowledge (e.g., Word2Vec, GLoVE, BERT).
- **Data Generation**: Crowdsourcing (Mechanical Turk) and synthetic data (e.g., SMOTE, GANs).
- **Data Labeling**: Crowdsourcing, active learning, weak supervision.
- **Data Cleaning**: Removing noise from features/labels.
- **Algorithmic Robustness**: Utilizing adversarial examples, transfer learning to improve model resilience.

Online testing of Machine Learning Systems

Limitations of Offline Testing:

- Offline testing only simulates real-world behavior, relying on historical data.
- May not account for unexpected future scenarios or real-time issues post-deployment.

Online Testing Approaches *(Zhang et al., 2022)*:

1. **Runtime Monitoring**: Continuous tracking of system behavior to detect runtime issues like races or deadlocks.
2. **User Response Monitoring**:

   A/B Testing: Compares new and old model versions with user cohorts to assess performance improvements.
3. **KPI Tracking**:

   Monitors key business metrics to ensure the model aligns with strategic goals and enhances performance.

Current industry practices in ML Deployment

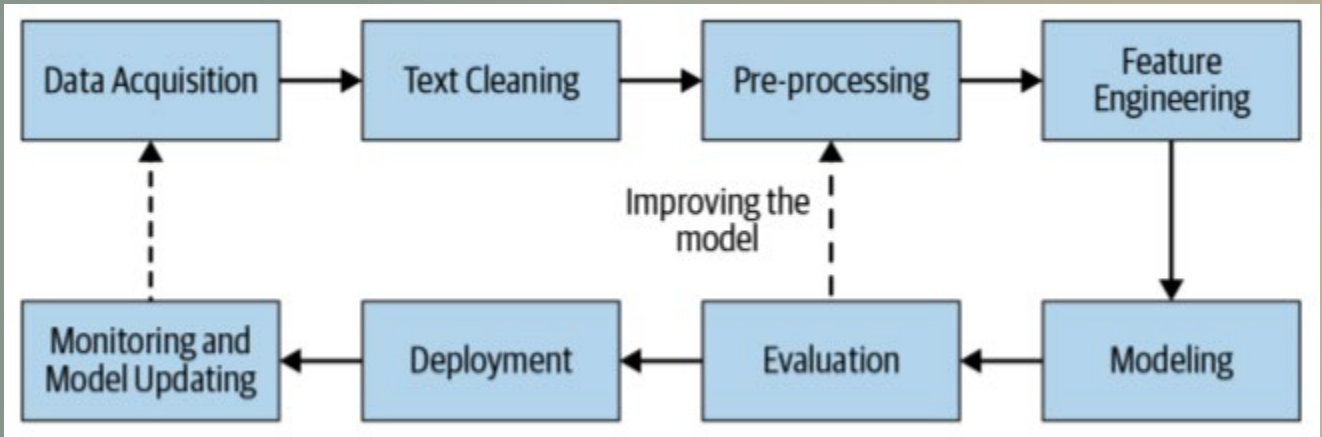
Deployment Issues

- **Slow Deployment Rate**: Transition from lab models to production is often slow (e.g., Alexa, Siri, assisted driving).
- **Skill Gaps**: Data scientists lack training for end-to-end deployment; models often remain unutilized due to deployment gaps (Patruno, Kaptein).
- **Comprehensive Process Needed**:
  - **AI-Driven Workflow**:
    - **Stages**: Data Prep → AI Modeling → Simulation & Testing → Deployment.
    - **Simulation Limits**: Must ensure model robustness across scenarios but may not capture all edge cases.

Current industry practices in ML Deployment

Integration Testing

- **Pipeline Reliability**:
  - Entire ML pipeline should undergo testing before deployment to catch issues that affect multiple components.
  - **Importance**: Small changes in one component can disrupt the entire pipeline, stressing the need for comprehensive integration tests.



A traditional Natural Language Processing Pipeline

MLOps: The Next Level in ML Deployment

MLOps Framework

- **Adaptation of DevOps Principles**:
  - **Version Control Expansion**: Applies not only to code but also to data, parameters, and training processes.
  - **Documentation**: Ensures consistent standards across development and production stages, aiding reproducibility.
- **Automated Safeguards**:
  - Embeds data collection, cleaning, and testing procedures directly into code to standardize processes.
  - Prevents ad-hoc, non-replicable steps in ML pipelines.

Continuous Pipeline Maintenance

- **Dynamic Data Handling**: ML models require periodic retraining to remain accurate.
- **Pipeline Management**:
  - Ensures smooth transitions and updates across model iterations, maintaining system continuity.

Improve safe development of ML systems

Challenges of Rapid Deployment

- **Potential Risks**: Fast deployment may compromise safety and thorough testing, risking poorly vetted ML-based products.
- **Case-Specific Standards**: Safety measures must match the product's impact (e.g., self-driving cars vs. voice recognition). Proper offline and online testing remains essential to avoid harm and PR issues.

Proposed Enhancements for Safe Deployment

- **Calibration Needs**: Algorithms should accurately estimate probabilities, especially in critical fields like healthcare, to prevent unnecessary risks and costs.
- **Quality Assurance (QA) Inspirations**:
  - **QA and QC Methods**: Adopt rigorous QA/QC standards from software engineering to meet fixed goals.
  - **Clinical Trial Approach**: Implement independent, clinical-trial-like evaluations for ML algorithms to ensure robustness and fair competition.

Improving Ethical Considerations

Principles of Responsible Machine Learning

- **Data Bias Detection and Mitigation**
  Ensuring that machine learning models are trained on unbiased and representative data, and that potential biases in the data are detected and addressed.
- **Explainability and Interpretability**
  Ensuring that machine learning models are transparent and explainable, and that their decisions and predictions can be understood and evaluated by users.
- **Model Fairness and Non-Discrimination**
  Ensuring that machine learning models do not discriminate against individuals or groups based on factors such as race, gender, or religion.
- **Data Privacy and Security**
  Respecting the privacy and data protection rights of individuals, and ensuring that personal data is collected, processed, and used in a responsible and ethical manner.
- **Human-Centered Machine Learning**
  Ensuring that machine learning models are developed with the needs and values of users and society in mind.
- **Reproducible Machine Learning**
  Ensuring that machine learning models can be replicated and verified by independent parties.

Data bias detection and mitigation

Data bias is a systematic error in datasets that leads to unfair or skewed model outcomes.

https://spotintelligence.com/2024/05/14/bias-mitigation-in-machine-learning/

Techniques:

- *Statistical analysis*
  - Descriptive Statistics: Explore data distribution and characteristics.
  - Regression Analysis: Examine relationships between variables.
  - T-test, ANOVA: Test for independence between categorical variables.
  - Chi-square test: Compare means across groups.
- *Blind testing*
- *Data augmentation*

Statistical Analysis a Case Problem

Predicting Customer Churn

- **Problem:** Suppose we have a dataset for a binary classification task, where we aim to predict whether a customer will churn $(y = 1)$ or not $(y = 0)$. The dataset contains input features x_j for $j = 1, \dots, p$.
- **Data Bias**: We suspect that the dataset may be biased, meaning that it does not represent the true population distribution fairly.
- **Identify Data Bias**:
  - Class Distribution: Calculate the proportion of positive and negative instances in the dataset.
  - Feature Distribution: Analyze the distribution of input features across different classes.
  - Correlation Analysis: Compute correlations between features and the target variable.
  - Group Comparisons: Group the data by relevant demographic attributes such as gender, race, income etc.

Addressing Data Bias

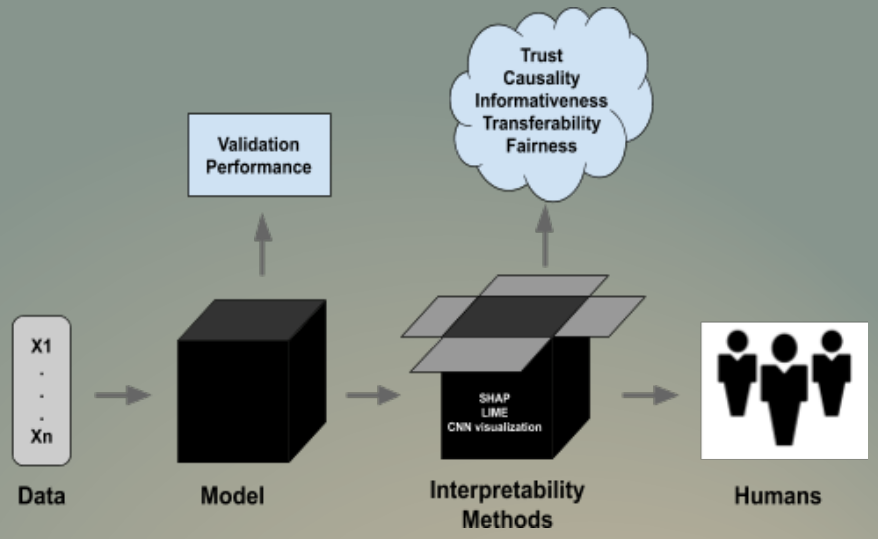
Predicting Customer Churn

If data bias is identified, it is essential to address it to ensure fair and accurate predictions. Possible strategies include:

- **Collecting more data**: If possible, collect more data to improve representation and reduce bias in the dataset.
- **Data augmentation**: Use techniques such as oversampling or synthetic data generation to balance the dataset and increase representation for underrepresented groups.
- **Feature engineering**: Modify or create new features that capture important characteristics for underrepresented groups, thereby reducing bias in feature representation.
- **Algorithmic modifications**: Explore fairness-aware algorithms that explicitly incorporate fairness constraints during model training to mitigate bias.

Explainability and Interpretability

XML focuses on creating ML models that provide interpretable, understandable explanations for predictions, enhancing trust and transparency.



https://blog.ml.cmu.edu/2020/08/31/6-interpretability/

Techniques:

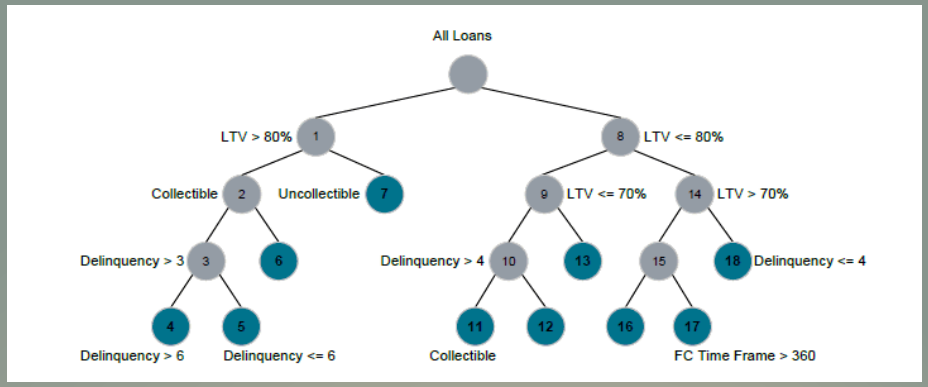
- *Model Visualization*: Shows model structure and data relationships.
- *Feature Importance Ranking*: Highlights key features influencing predictions.
- *Local Explanations*: Explains individual predictions, improving transparency.

Taxonomy of XML Techniques

- **Model-agnostic vs. Model-specific:** Model-agnostic techniques are adaptable to any model, while model-specific techniques are tailored for particular model types.
- **Rule-based vs. Model-based:** Rule-based techniques rely on explicit rules for explanations, while model-based techniques use the model's internal mechanisms, such as feature importance, to generate explanations.
- **Post-hoc vs. Ante-hoc:** Post-hoc XML techniques create explanations after the model's prediction, while ante-hoc XML techniques provide explanations during training
- **Local vs. Global:** Local XML techniques explain individual predictions, while global XML techniques provide insight into the model's overall behavior and patterns.
- **Human-centric vs. Machine-centric:** Human-centric techniques aim to create explanations that are clear and useful for people, whereas machine-centric XML techniques prioritize efficiency for machine processing.
- **Qualitative vs. Quantitative:** Qualitative techniques provide explanations in human-readable forms while quantitative techniques offer explanations as statistical or mathematical measures.
- **Interactive vs. Static:** Interactive techniques let users explore scenarios and conduct what-if analyses, while static techniques provide fixed explanations.

Decision Trees

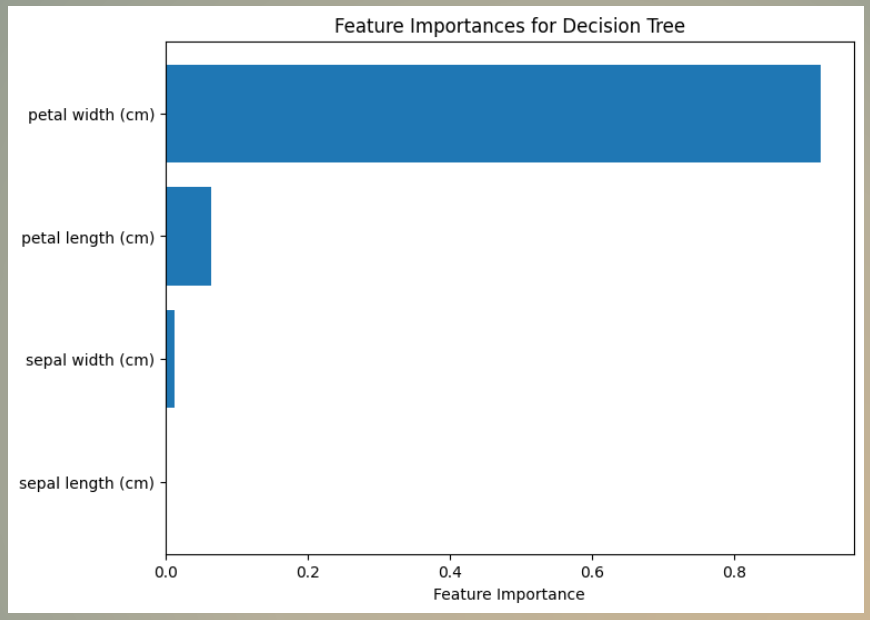
Decision trees are interpretable machine learning models that visually represent decision-making processes, providing insights into data patterns.



An example using Iris dataset:

- **Features:**
  - Sepal Length (cm)
  - Sepal Width (cm)
  - Petal Length (cm)
  - Petal Width (cm)

- **Target:** Classify iris flowers into one of three species. “Setoca”, “Versicolor”, “Virginica”.



Local Interpretable Model-Agnostic Explanations (LIME)

Model is making the right prediction in a **right** way

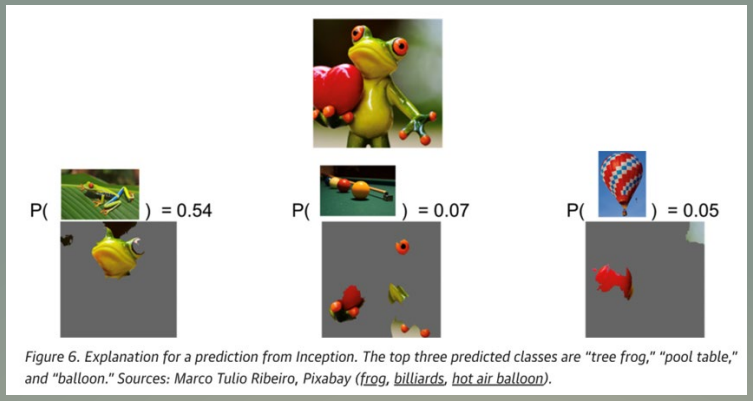


Figure 6. Explanation for a prediction from Inception. The top three predicted classes are "tree frog," "pool table," and "balloon." Sources: Marco Tulio Ribeiro, Pixabay (frog, billiards, hot air balloon).

Ribeiro, M.T., Singh, S. and Guestrin, C., 2016, August. " Why should i trust you?" Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1135-1144).

Process:

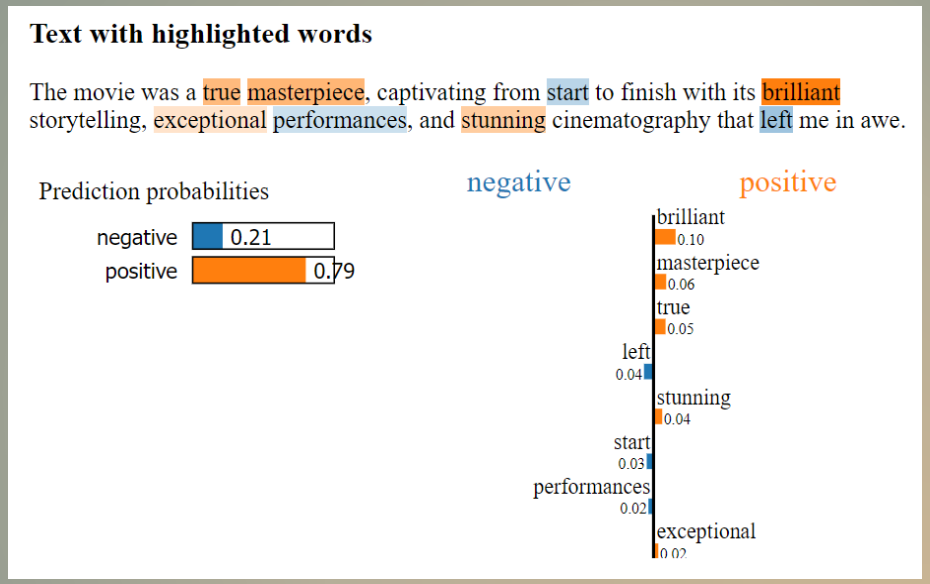
- **Background**: Let X denote the input space and $\boldsymbol{x} \in X$ represent a specific instance. The black-box model is denoted as $f: X \rightarrow Y$, where Y is the set of possible outcomes.
- **Local Linearity Assumption**: LIME assumes that the black-box model is locally linear around the instance of interest. To approximate the model's behavior in a local region, LIME creates a neighborhood around the instance x.
- **Proximity Measure**: LIME employs a similarity measure to define the proximity of instances to the instance of interest, which can be based on distance metrics such as Euclidean distance or cosine similarity.
- **Local Model Training**: LIME generates a local training set by sampling instances from the neighborhood of x according to the proximity measure. The black-box model's predictions are used to obtain the labels for the local training set.

LIME for a natural language processing task

- **Application:** Use of LIME to interpret predictions from a logistic regression classifier applied to sentiment analysis on IMDb movie reviews. Text data is converted into numerical features using TF-IDF vectorization, and the model's performance on training data is evaluated.

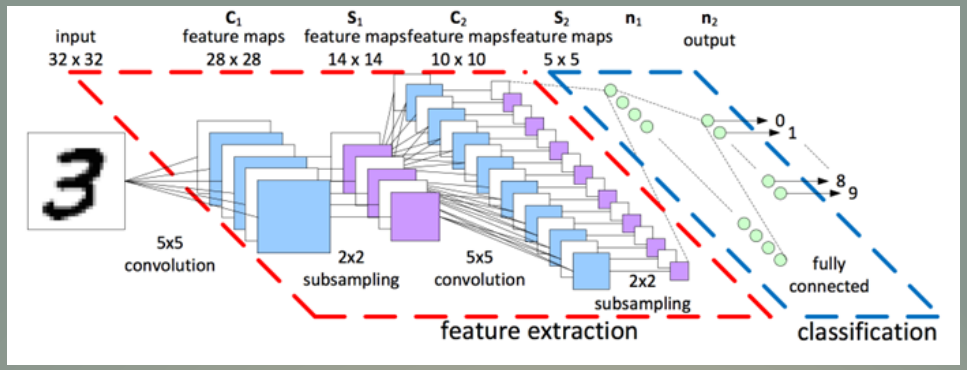
- LIME enhances model transparency by explaining specific predictions, showing how particular words or phrases influence the sentiment classification, which is crucial for understanding and trusting NLP models in real-world applications.

- **Results**: Visually highlights the key words or features driving the logistic regression classifier's positive sentiment prediction for a movie review, focusing on the top eight influential words to clarify how specific terms contribute to the model's decision.



Heat Maps

- **Convolutional neural networks (CNNs)** are deep learning models designed to process and analyze visual data by automatically detecting patterns through layers of convolutional filters.



Visualizing heatmaps of class activation:

- This technique is a type of class activation map (CAM) visualization, producing heatmaps to highlight image regions most relevant to a class prediction.
- Grad-CAM (Gradient-weighted Class Activation Mapping) generates class activation heatmaps by weighting the channels in a CNN's feature map based on the gradient of the predicted class score.
- In simple terms, Grad-CAM highlights the most important parts of the feature map for predicting a specific class by calculating how much each channel contributes to the class score.
- Grad-CAM creates a heatmap that highlights image regions most influential in predicting the target class, offering visual insight into the CNN's decision-making process.

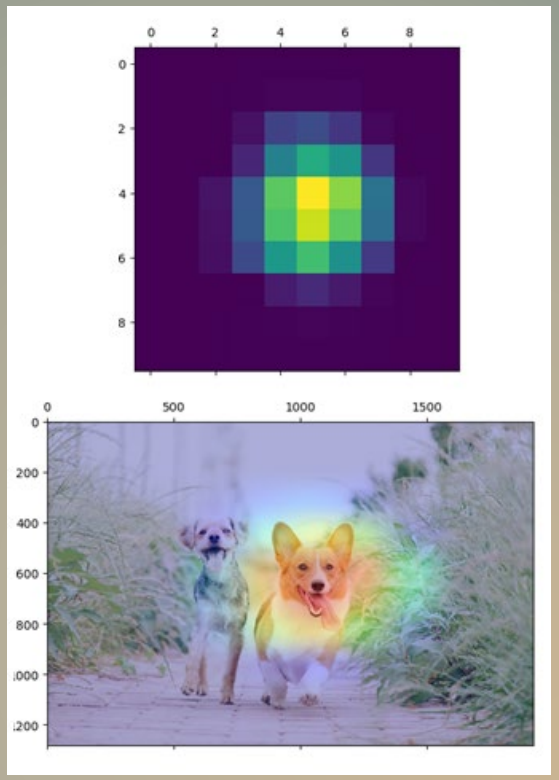
Heat Maps in Computer Vision

Application: Using the pre-trained Xception model, we aim to understand two key questions:

- Why did the model classify this image as a 'Pembroke' dog?
- Which regions in the image are most influential for this classification?

The workflow:

- To pinpoint regions relevant to the 'Pembroke' classification, we set up the Grad-CAM workflow by creating two models:
  - One maps the input image to the last convolutional layer activations.
  - The other maps these activations to the final class predictions.
- We compute the gradient of the predicted class with respect to the last gradient convolutional layer activations.
- The class activation heatmap is created by pooling and weighting the gradient tensor, then normalized to 0–1 for visualization.
- The heatmap is overlaid on the original image, highlighting key areas influencing the 'Pembroke' classification.



Model Fairness and Non-Discrimination

Fairness in machine learning refers to ensuring that models make decisions without bias or discrimination, treating all groups or individuals equitably.

https://www.turing.ac.uk/sites/default/files/2023-12/aieg-ati-fairness_1.pdf

- Fairness metrics
  - Demographic parity
  - Equalized odds
  - Equal opportunity
- Fairness Constraints

$$\min_{\theta} L(Y, \hat{Y}) + \lambda \, Fairness(\hat{Y}, A)$$

where θ represents the model parameters, L is the standard loss function, λ controls the trade-off between accuracy and fairness, $\hat{Y}$ is the predicted outcome and A is the protected attribute (e.g. gender).

Data Privacy and Security

Data privacy and security are crucial considerations in machine learning projects and vital in responsible machine learning.

https://www.traverselegal.com/blog/ai-data-privacy-and-security/

- Privacy-preserving techniques
  - Differential privacy
  - Secure multi-party computation
  - Federated learning
  - Homomorphic encryption
  - Secure aggregation
  - Data perturbation
  - Privacy-preserving data publishing

Data Privacy and Security

- Data anonymization and de-identification
  - Data anonymization
  - K-Anonymity
  - L-Diversity
  - T-closeness
- Secure data storage and transfer
  - Encryption
  - Secure protocols
  - Access control
  - Data integrity
  - Secure storage infrastructure
- User consent and data usage policies
- Secure infrastructure and system hardening

Human-Centered Machine Learning

- Human-Centered Design in Machine Learning
  - Engage users and stakeholders throughout the design process to understand their needs, challenges, and goals.
  - Use methods such as user research, surveys, and focus groups to gather feedback.

https://hcai.site/

  - Develop user personas to represent diverse user types and needs.
  - Design an intuitive, accessible, and easy-to-use user experience.
  - Incorporate user feedback to refine and improve the model.
  - Test with real users to evaluate effectiveness, usability, and impact.

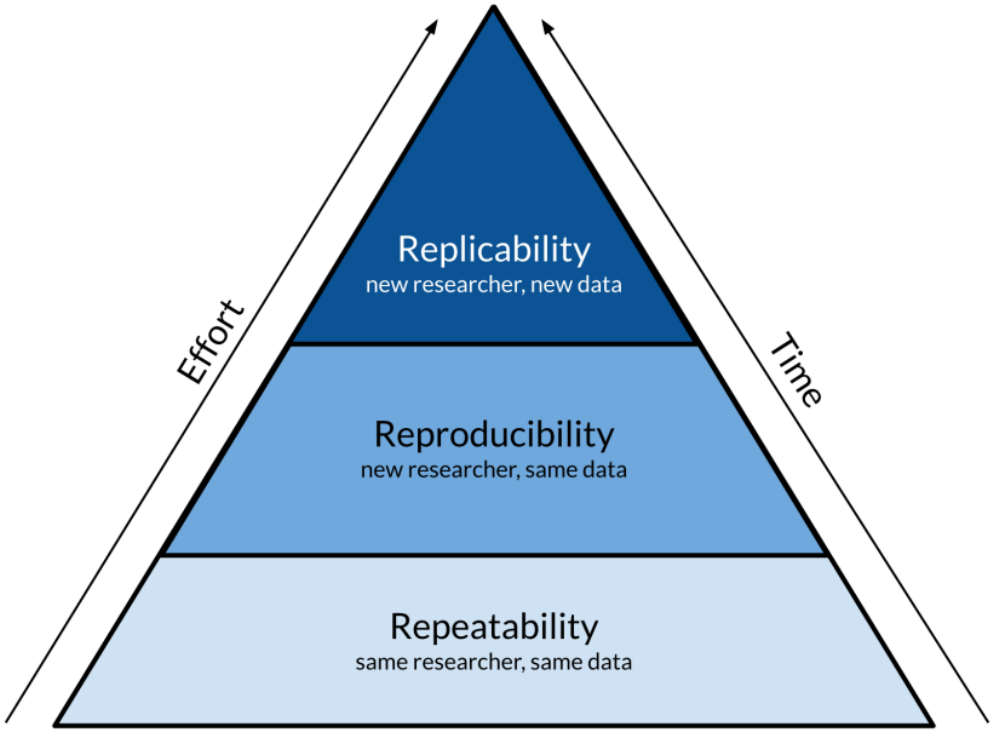
Example of Human-Centered ML

- **Moroney et al. (2021):** Developed a social media dataset and linguistic rules to evaluate the interpretability of a classification model for unreliable tweets, promoting usability and user relevance
- 17 linguistic characteristics identified in a list of 560 tweets

Linguistic attribute	Example from dataset
Hyperbolic, intensified, superlative, or emphatic language [2,16]	e.g., 'blame', 'accuse', 'refuse', 'catastrophe', 'chaos', 'evil'
Greater use of punctuation and/or special characters [2,15]	e.g., 'YA THINK!!?!!?!', 'Can we PLEASE stop spreading the lie that Coronavirus is super super super contagious? It's not. It has a contagious rating of TWO'
Strongly emotional or subjective language [2,16]	e.g., 'fight', 'danger', 'hysteria', 'panic', 'paranoia', 'laugh', 'stupidity' or other words indicating fear, surprise, alarm, anger, and so forth
Greater use of verbs of perception and/or opinion [15]	e.g., 'hear', 'see', 'feel', 'suppose', 'perceive', 'look', 'appear', 'suggest', 'believe', 'pretend'
Language related to death and/or war [8]	e.g., 'martial law', 'kill', 'die', 'weapon', 'weaponizing'
Greater use of proper nouns [11]	e.g., 'USSR lied about Chernobyl. Japan lied about Fukushima. China has lied about Coronavirus. Countries lie. Ego, global'
Shorter and/or simpler, language [11]	e.g., '#Iran just killed 57 of our citizens. The #coronavirus is spreading for Canadians Our economy needs support.'
Hate speech [8] and/or use of racist or stereotypical language	e.g., 'foreigners', 'Wuhan virus', reference to Chinese people eating cats and dogs
First and second person pronouns [15,16]	e.g., 'I', 'me', 'my', 'mine', 'you', 'your', 'we', 'our'
Direct falsity claim and/or a truth claim [2]	e.g., 'propaganda', 'fake news', 'conspiracy', 'claim', 'misleading', 'hoax'
Direct health claim	e.g., 'cure', 'breakthrough', posting infection statistics
Repetitive words or phrases [11]	e.g., 'Communist China is lying about true extent of Coronavirus outbreak - If Communist China doesn't come clean'
Mild or strong expletives, curses, slurs, or other offensive terms	e.g., 'bitch', 'WTF', 'dogbreath', 'Zombie homeless junkies', 'hell', 'screwed'
Language related to religion	e.g., 'secular', 'Bible'
Politically biased terms	e.g., 'MAGA', 'MAGAt', 'Chinese regime', 'deep state', 'Communist China'
Language related to financial or economic impact	e.g., 'THE STOCK MARKET ISN'T REAL THE ECONOMY ISN'T REAL THE CORONAVIRUS ISN'T REAL FAKE NEWS REEEEEEEEEEEEEEEEE'
Language related to the Trump presidential election, campaign, impeachment, base, and rallies	e.g., 'What you are watching with the CoronaVirus has been planned and orchestrated. '

Moroney, C., Crothers, E., Mittal, S., Joshi, A., Adalı, T., Mallinson, C., Japkowicz, N. and Boukouvalas, Z., 2021. The case for latent variable vs deep learning methods in misinformation detection: An application to covid-19. In Discovery Science: 24th International Conference, DS 2021, Halifax, NS, Canada, October 11–13, 2021, Proceedings 24 (pp. 422-432). Springer International Publishing.

The Rs in machine learning

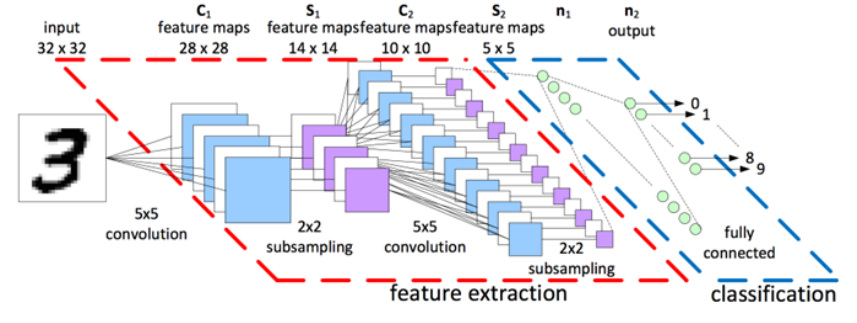


Based off of a figure from Essawy et al, 2020 https://doi.org/10.1016/j.envsoft.2020.104753

Reproducible Machine Learning

Repeatability: Refers to the ability to reproduce the same results when an experiment is conducted multiple times, using the same dataset, model, and experimental setup.

Example: Consider a machine learning experiment where researchers train a CNN for image classification using a publicly available dataset.



Seed Initialization: Set the random seed at the beginning of the experiment.

Code Versioning: Maintain a version-controlled repository for the codebase used in the experiment.

Documentation: Keep detailed records of the experimental setup, including hyper parameters, preprocessing steps, and model architecture.

Hardware and Software Specifications: Specify the hardware and software con-figurations used, including the type and version of the machine learning framework, libraries, and dependencies.

Reproducible Machine Learning

Reproducibility: Refers to the ability to recreate and validate the results of a study or experiment using the same data, code, and methods.

Example: Consider a machine learning study where researchers develop a deep learning model for sentiment analysis on a specific dataset. To ensure reproducibility, the following steps can be taken.

Data Availability: Make the dataset used in the study publicly available or provide detailed information about how to obtain the dataset. This ensures that other researchers can access the same data and reproduce the experiments.

Code Sharing: Share the code and scripts used to preprocess the data, train the model, and evaluate the results. Version control tools like Git can be used to maintain a repository that includes the code and its dependencies.

Documentation: Provide comprehensive documentation that describes the experimental setup, including details about hyperparameters, model architecture, and any pre-processing steps. This documentation helps in reproducing the experiments accurately.

Environment Description: Specify the software and hardware configurations used in the study, including the version of the machine learning framework, libraries, and dependencies. This information helps in replicating the experiments on different systems.

Reproducible Machine Learning

Replicability: Refers to the ability to reproduce the results of a study or experiment using independent data and methods.

Example: Consider a machine learning study where researchers propose a novel algorithm for image classification. To ensure replicability, the following steps can be taken.

Independent Data Collection: Collect a separate and independent dataset that is different from the one used in the original study. This ensures that the findings can be tested on new data.

Method Replication: Replicate the methodology described in the original study, including data preprocessing steps, feature extraction techniques, and model training procedures. Ensure that the implementation details are as close as possible to the original study.

Comparison and Evaluation: Apply the replicated method to the new dataset and compare the obtained results with the original study. Assess the similarity of the outcomes, such as classification accuracy or other relevant performance metrics.

Discussion and Interpretation: Discuss the similarities and differences observed between the replicated results and the original findings.

Reproducible Machine Learning Challenges

Reproducibility is a critical aspect of scientific research, including machine learning. Some of the challenges include:

Data Availability: Many studies use proprietary or restricted datasets, limiting replication efforts. Solutions include promoting open data practices, sharing platforms, and detailed instructions for data access.

Code and Software Dependencies: Machine learning experiments rely on complex code with many dependencies, where version differences in software packages can disrupt reproducibility. Sharing code and specifying software dependencies clearly can help address this issue.

Hyperparameter Tuning: Machine learning models often require tuning hyperparameters for optimal performance, but these choices can greatly impact results, complicating reproducibility. Clear documentation of the chosen hyperparameters and their rationale can help mitigate this issue.

Hardware and Computational Resources: Machine learning experiments often demand specific hardware, and variations in computational environments can affect reproducibility. Documenting the hardware and resources used can help others replicate the experiments.

Solutions to make ML projects reproducible

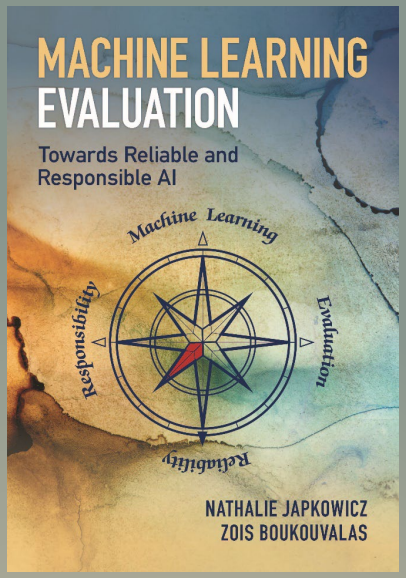
Tracking Experiment Metadata: Experiment tracking involves recording important meta-data associated with each experiment. This includes information such as the date and time of the experiment, the machine learning algorithm used, hyperparameters, dataset details, and any preprocessing steps applied. Tracking this metadata helps in understanding and reproducing the experiment's results.

Managing Code Versions: Experiment tracking and logging also involve managing code versions. It is essential to keep track of the code used for each experiment, including the specific commit or version. This ensures that the exact code used can be reproduced and traced back to the experiment's results.

Tools for Experiment Tracking and Logging: Several tools and frameworks are available for experiment tracking and logging in machine learning. Examples include MLflow, Neptune, Sacred, and TensorBoard. These tools provide features such as experiment versioning, result visualization, and collaboration functionalities.

Version Control: Version control is an essential aspect of an artifact store. It allows tracking and managing different versions of artifacts, such as models and datasets. This enables reproducibility and ensures that previous versions can be easily retrieved and compared.

Summary and Conclusion



Key Topics Covered:

- Classification techniques
- Selecting performance measures
- Sampling strategies
- Statistical tests
- Alternative ML paradigms
- Deployment practices for reliability and responsibility

Importance of Evaluation in Machine Learning

- Central Role of Evaluation:
  - Ensures accuracy and robustness in models
  - Helps prevent misinterpretation and supports rigorous validation
- Beyond Technical Performance:
  - Emphasizes the social and ethical impact of ML
  - Ensures trust and accountability in real-world applications