Building a Reproducible Model Workflow Cont.

Train, Validation and Experiment Tracking

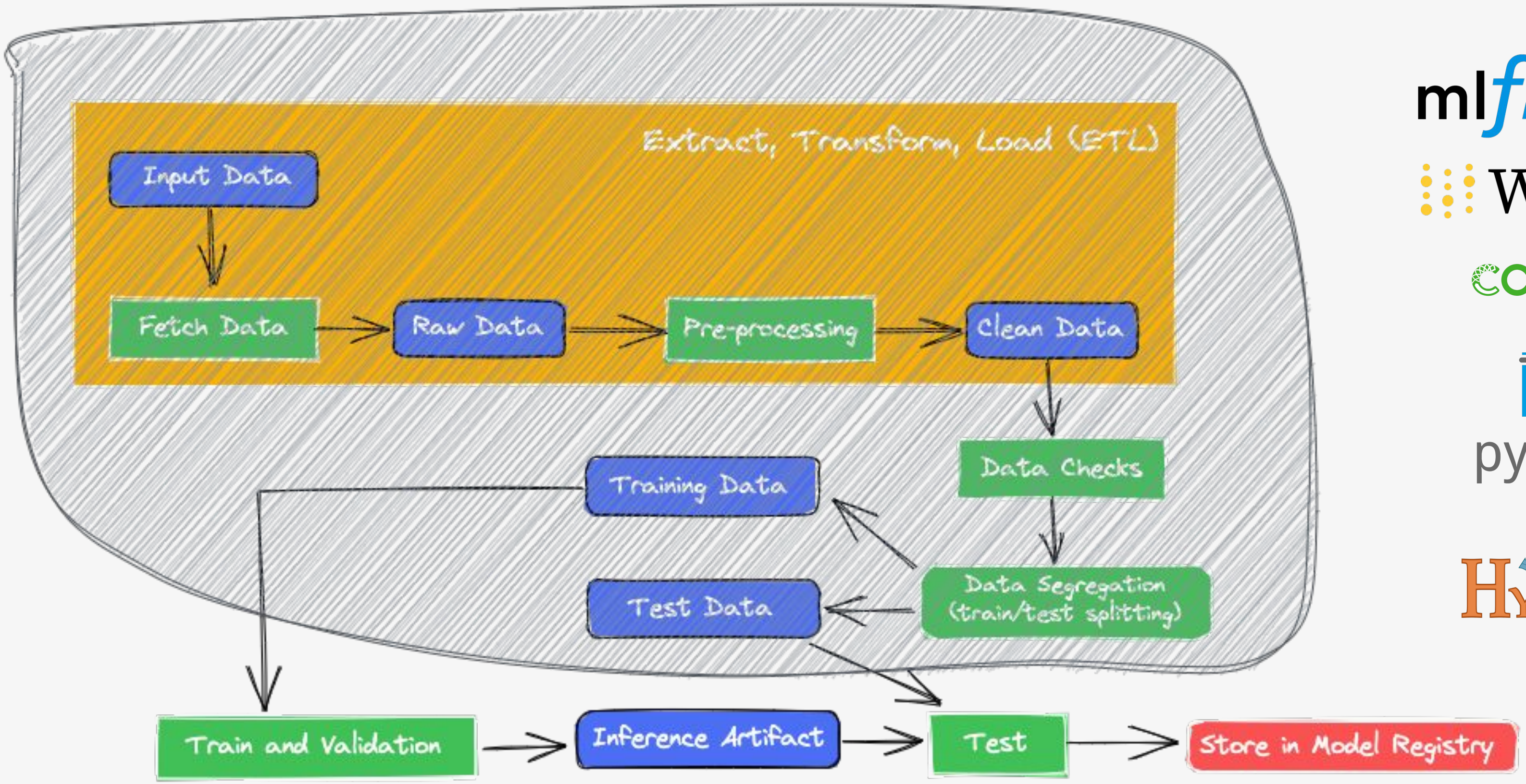
UFRN UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE

DCA

Table of Contents

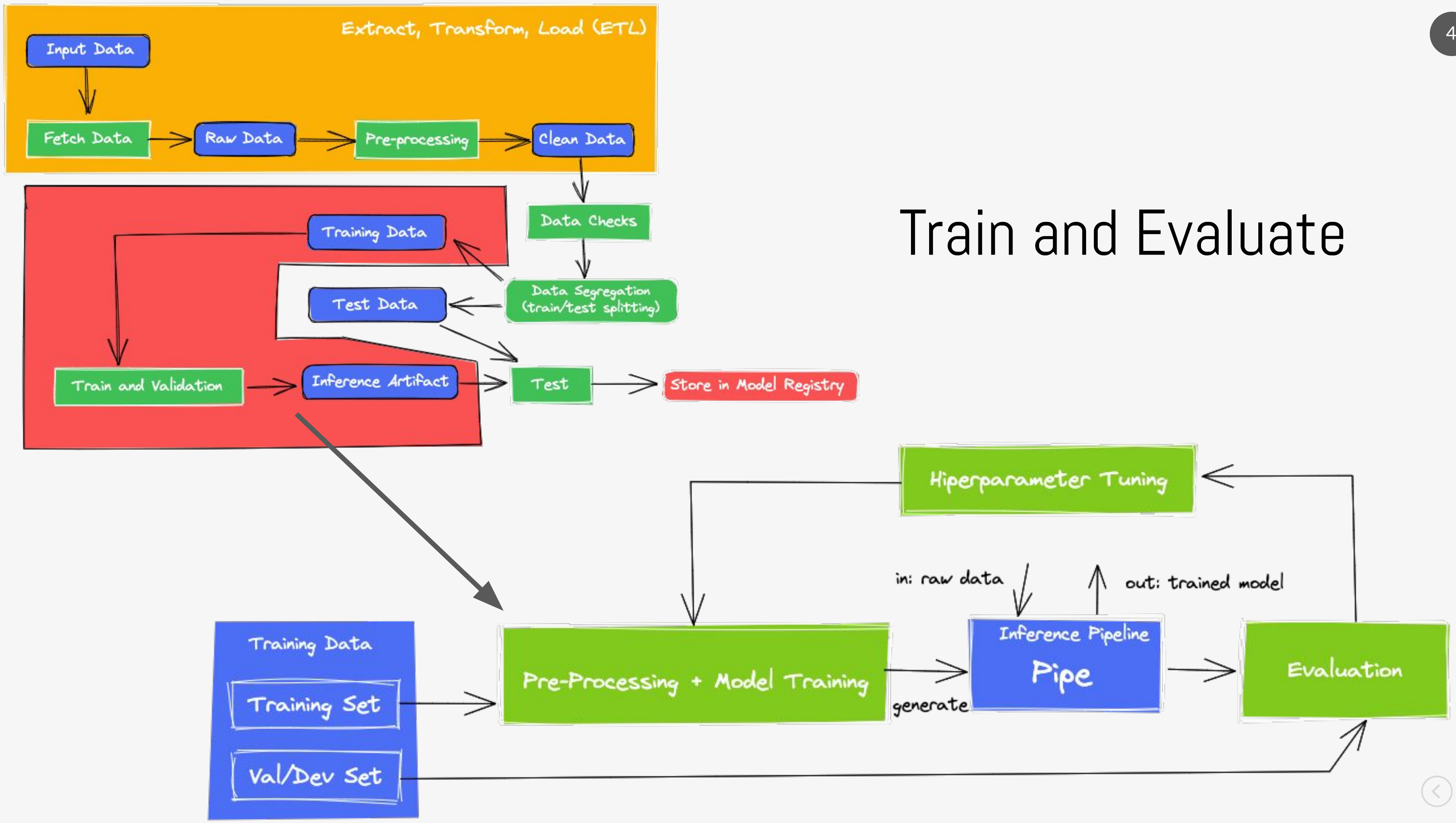
An inference pipeline is an ML pipeline that contains everything that needs to run in production at inference time: a pre-processing step that transforms the data input to the data expected by the model, and then the model

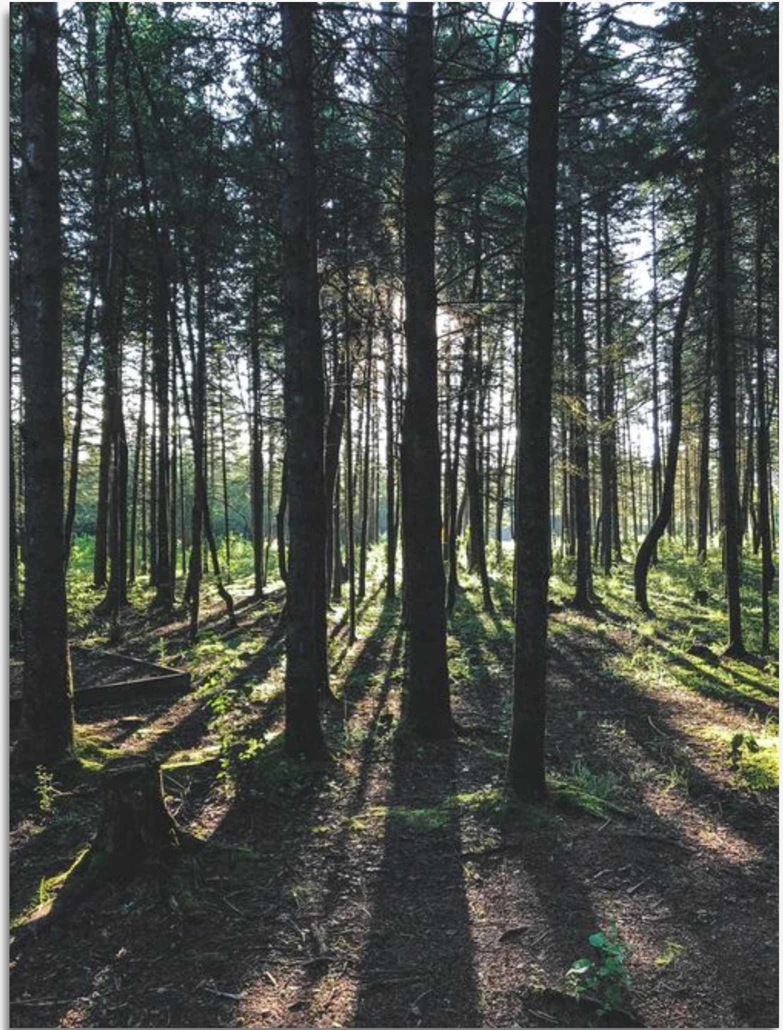
Previously on lessons



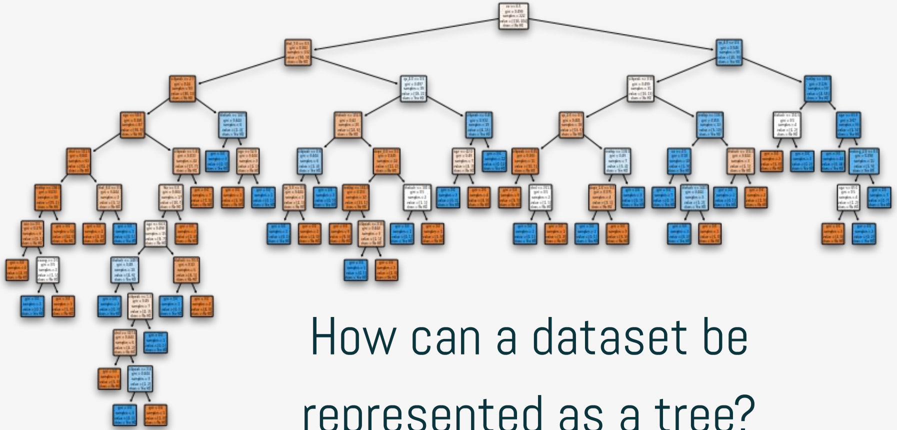
mlflow
W&B
CONDA
pytest
HYDRA

Train and Evaluate





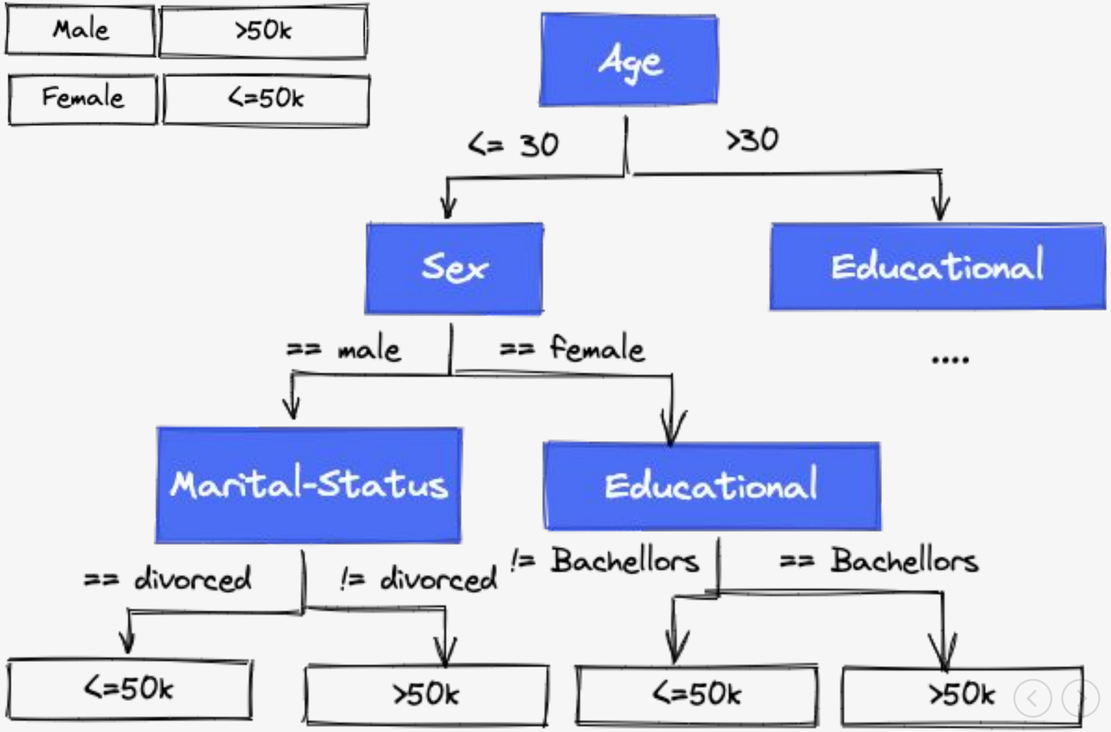
Decision Trees



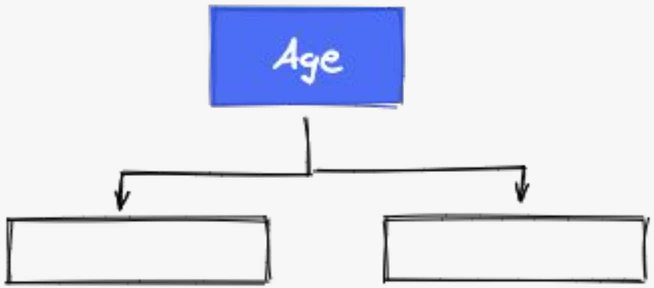
How can a dataset be represented as a tree?

Age	Marital-Status	Educational	Sex	High-Income
28	Never-Married	Bachelors	Female	<=50k
46	Never-Married	Assoc-acdm	Female	<=50k
35	Married-civ-spouse	Some-college	Male	<=50k
27	Married-civ-spouse	Bachelors	Male	>50k
59	Divorced	Some-college	Female	<=50k

Decision Tree (classification)



How can we split the tree?

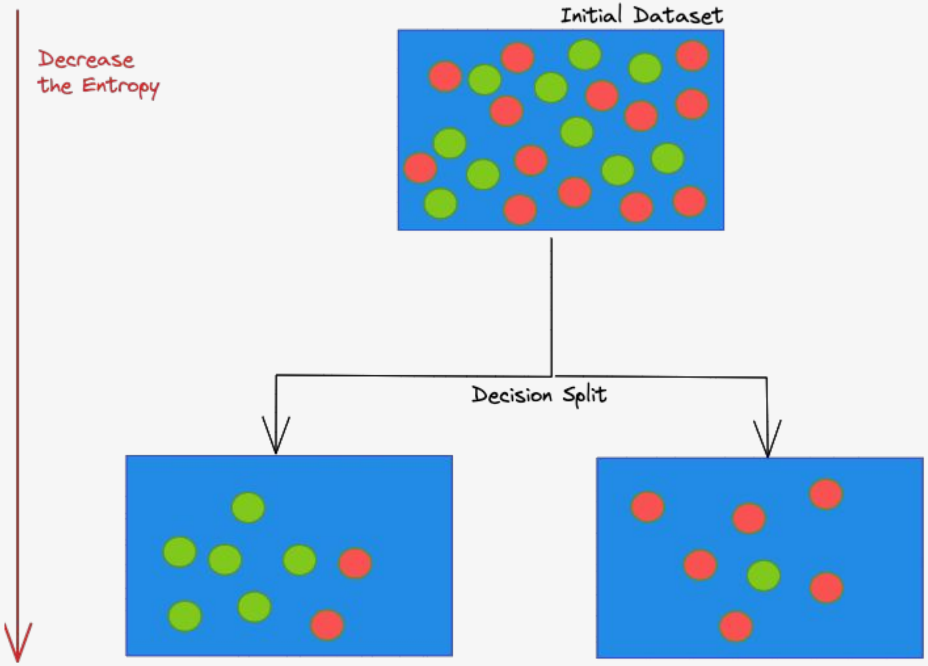


Algorithm used in Decision Trees

1. ID3 (Entropy)
2. Gini Index
3. Chi-Square
4. Reduction in Variance
   a. C4.5, pruning
5. ...

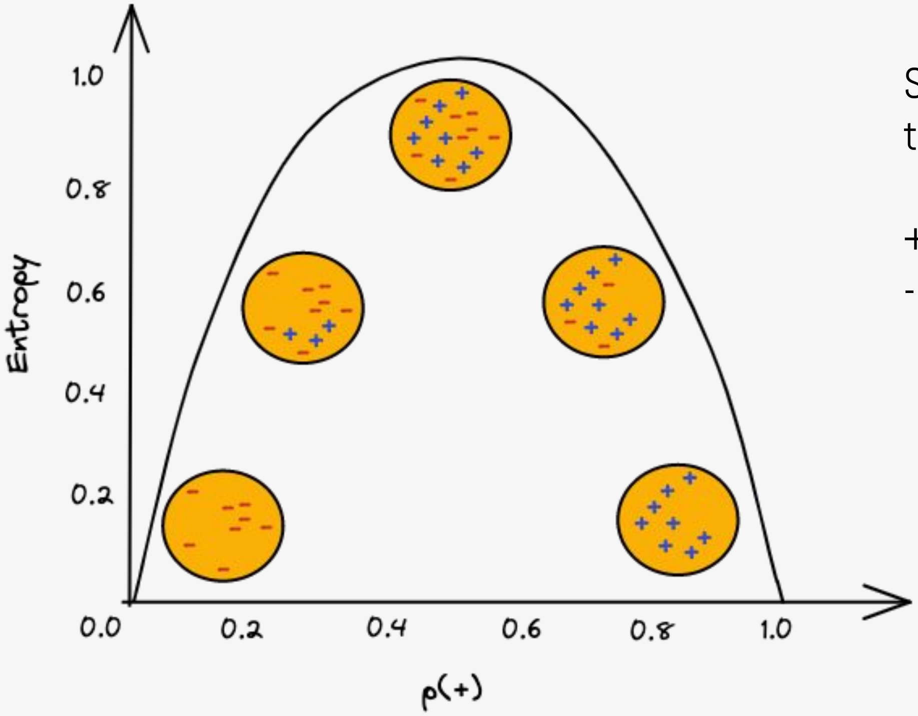
Entropy is an indicator of how messy your data is.

Why Entropy in Decision Trees?



- The goal is to tidy the data.
- You try to separate your data and group the samples together in the classes they belong to.
- You maximize the purity of the groups as much as possible each time you create a new node of the tree
- Of course, at the end of the tree, you want to have a clear answer.

Mathematical Definition of Entropy



Suppose a set of N items, these items fall into two categories:

+ gain > 50k (k)
- gain <= 50k (m)

$$p = \frac{k}{N}, q = \frac{m}{N}$$
$$Entropy = -p \log p - q \log q$$

Generalization

Feature X

$$E(x) = - \sum_{i=1}^{c} P(x_i) \log_b P(x)$$

$P(x_i)$ is the fraction of examples in a given class i

<= 50k.	17288
> 50k.	5487

```
from scipy.stats import entropy
entropy(df_train.high_income.value_counts(), base=2)
0.7965702796015677
```

Entropy using the frequency table of two attributes

		High Income		
		<= 50k	> 50k	
Age	<=37	7206	3883	11089 (48%)
	>37	10082	1604	11686 (52%)

$$E(T \mid X) = \sum_{c \in X} \frac{|X_c|}{|X|} E(T \mid X_c)$$

```
cross = pd.crosstab(
          df_train.age <= df_train.age.median(),
          df_train.high_income)
```

```
0.486894 * entropy(cross.iloc[0], base=2) \
+ 0.513106 * entropy(cross.iloc[1], base=2)
0.7509335429830957
```

Information Gain

$$IG(T,X) = E(T) - E(T|X)$$

Information Gain from X on T

The information gain is based on the **decrease in entropy after a dataset is split** on an attribute.

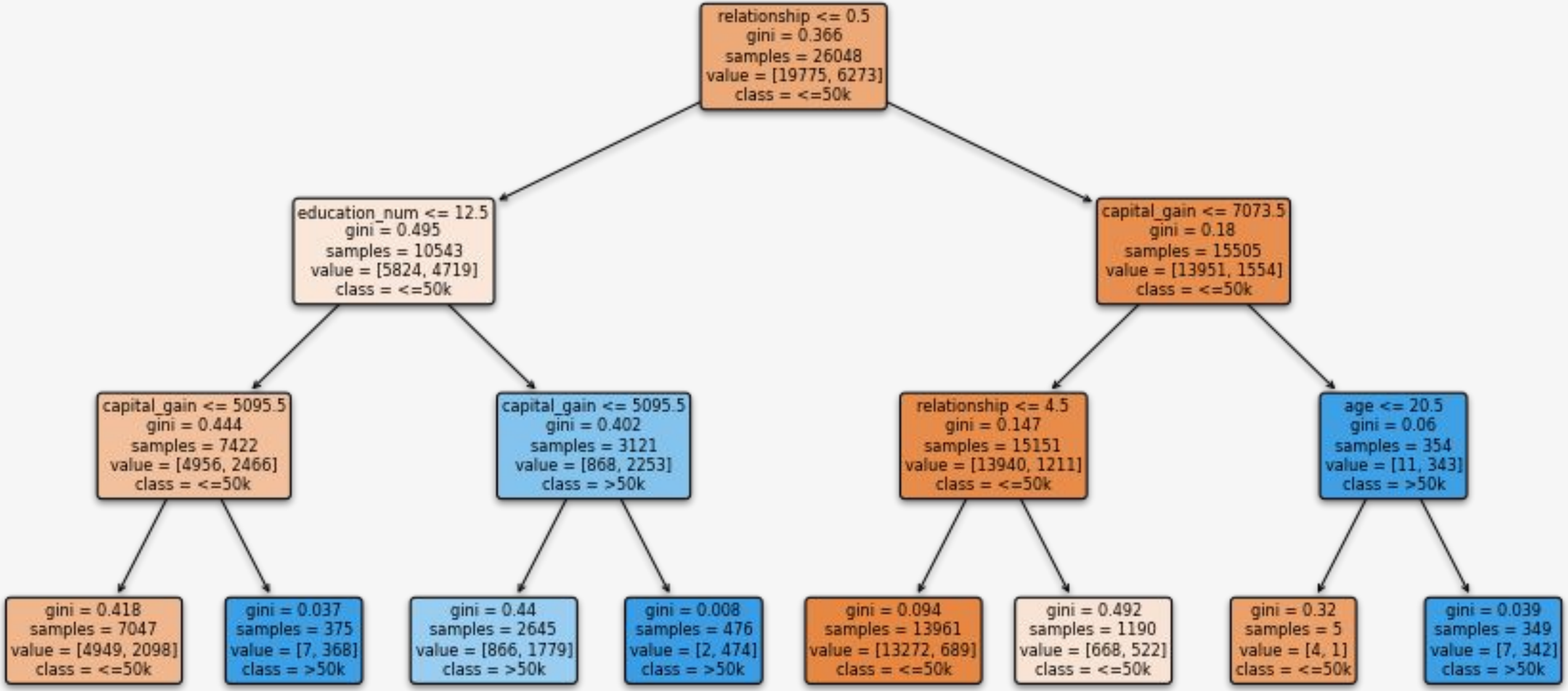
Constructing a decision tree is all about finding attribute that returns the **highest information gain** (i.e., the most homogeneous branches).

$$\text{Gini}(x) = 1 - \sum_{i=1}^{c} P(x_i)^2$$

$$\text{Entropy}(x) = - \sum_{i=1}^{c} P(x_i) \log_b P(x_i)$$

Gini index or Entropy is the criterion for calculating **Information Gain**. Both of them are measures of impurity of a node.

```
from sklearn.tree import plot_tree
```



Taxonomy of Classifier Evaluation Metrics

Threshold Metrics

Ratio when a predicted class does not match

Accuracy, Error, Sensitivity, Specificity, G-mean, precision, recall, fbeta-measure

Ranking Metrics

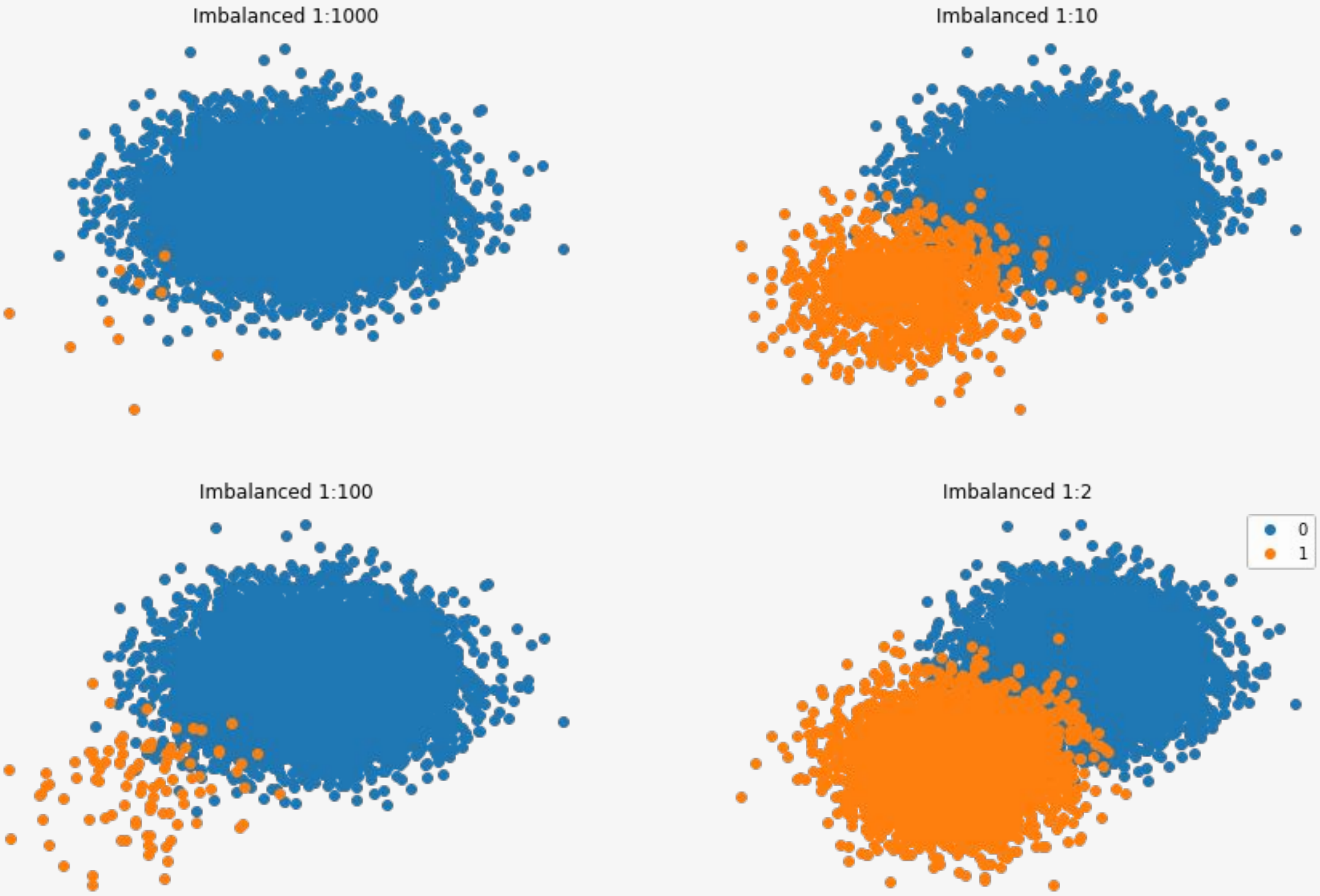
Based on score of class membership and variatons of thresholds to measure the effectiveness of classifiers.

ROC Curve, ROC AUC, Precision-Recall Curve

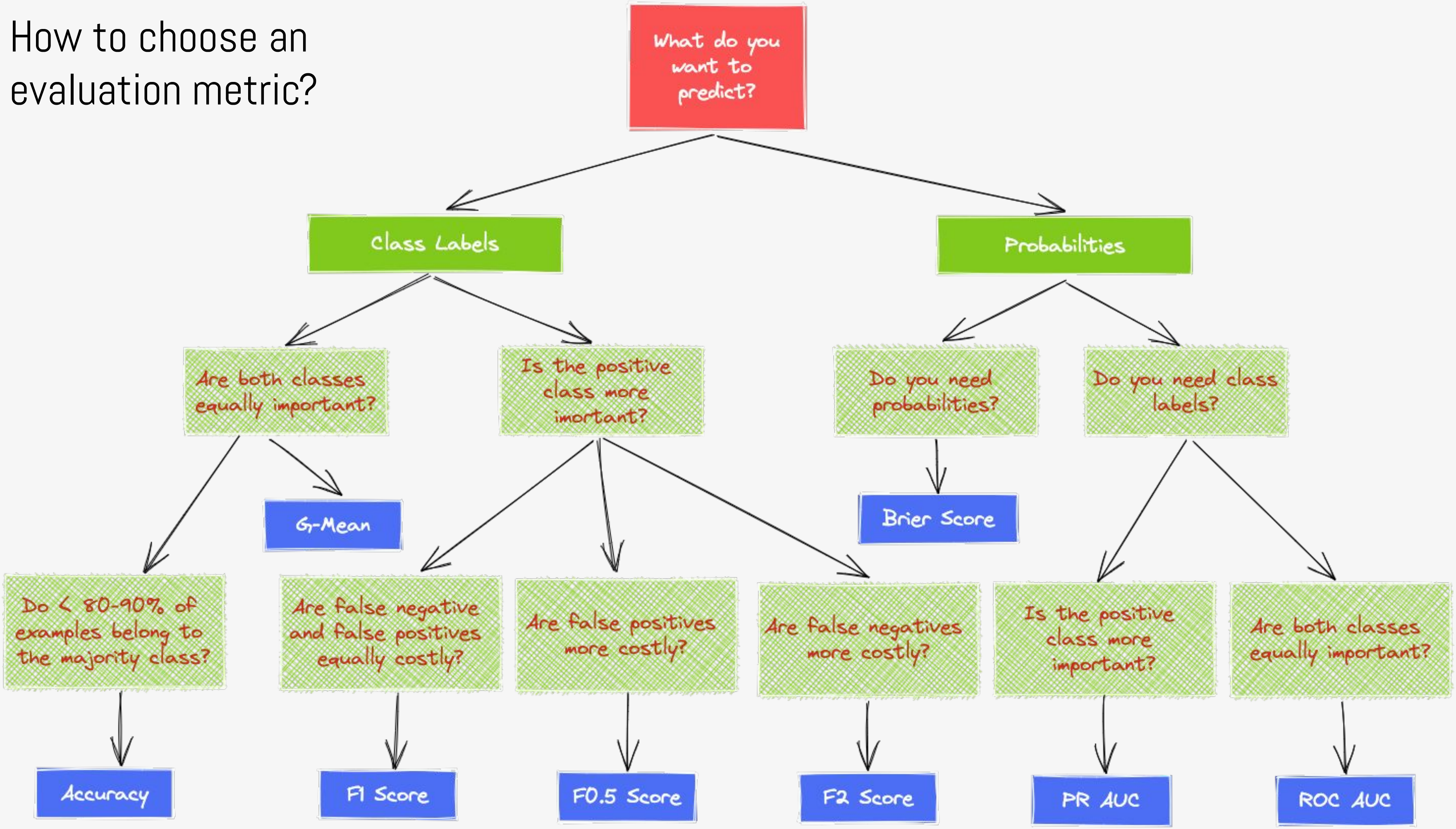
Probability Metrics

Quantify the uncertainty in a classifier's prediction

Log-Loss
Brier Score

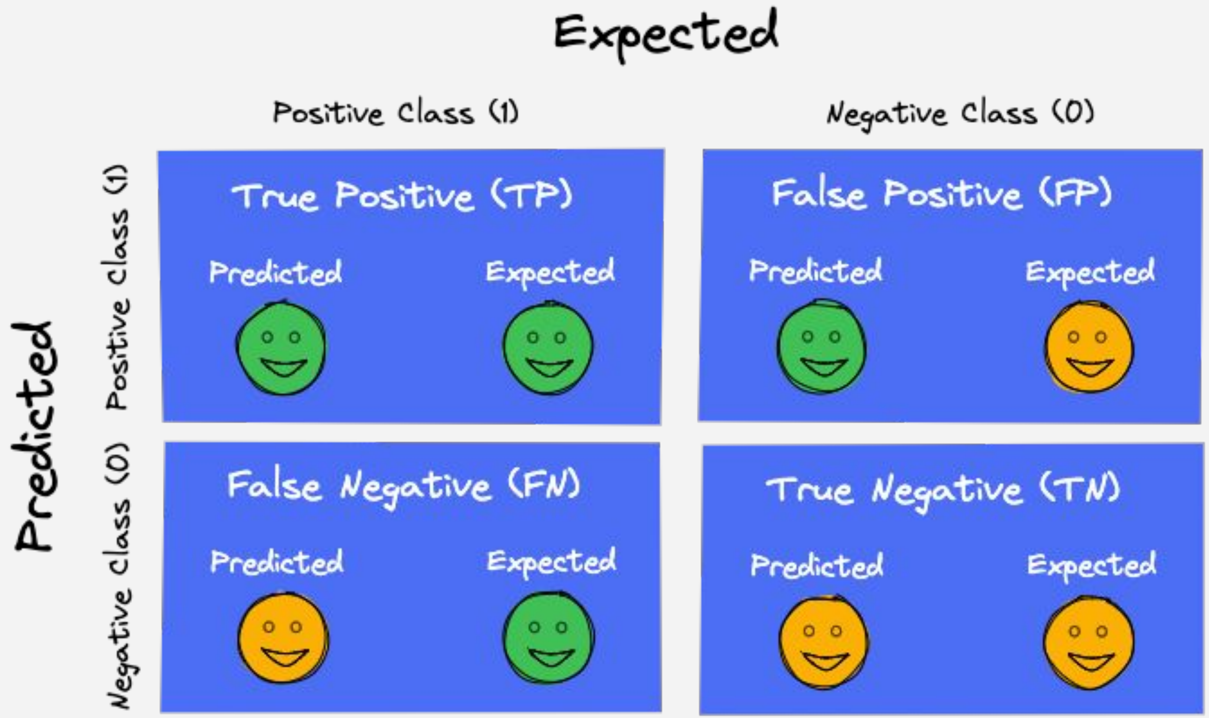

Imbalanced 1:1000
Imbalanced 1:10
Imbalanced 1:100
Imbalanced 1:2
0
1

How to choose an evaluation metric?



@Brownlee, Jason. Imbalanced classification with python.

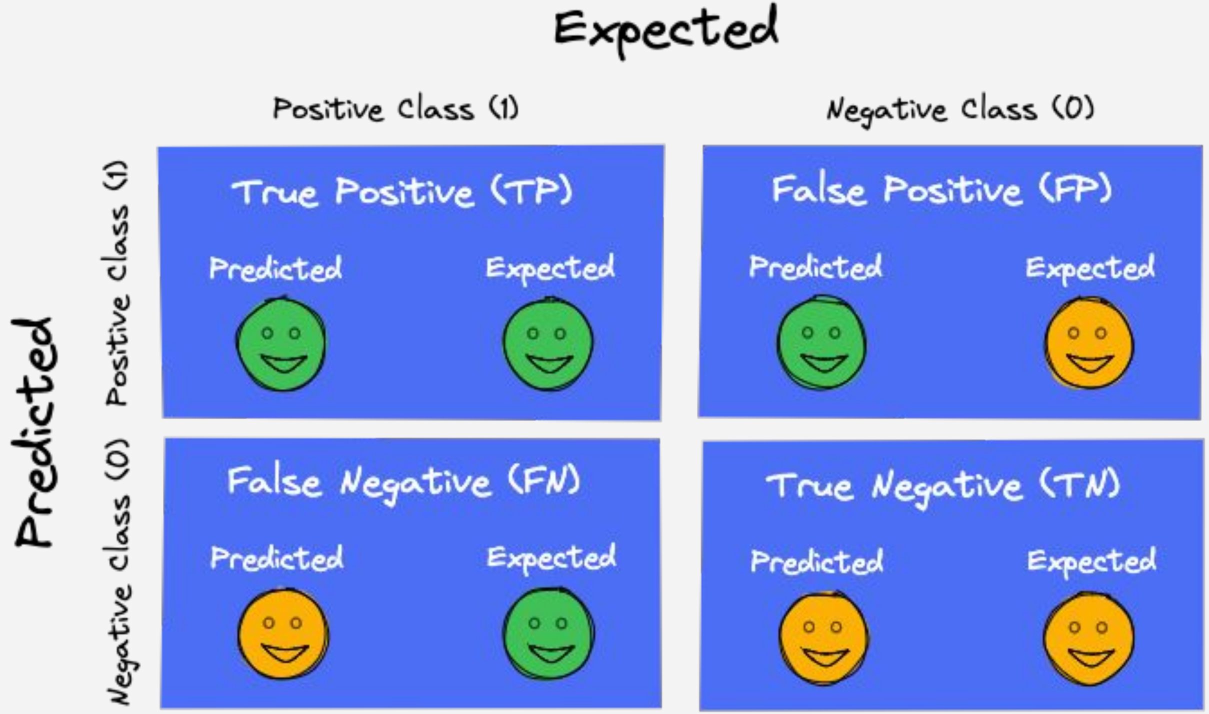
Confusion Matrix



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

$$\text{Error} = 1 - \text{Accuracy}$$

Confusion Matrix

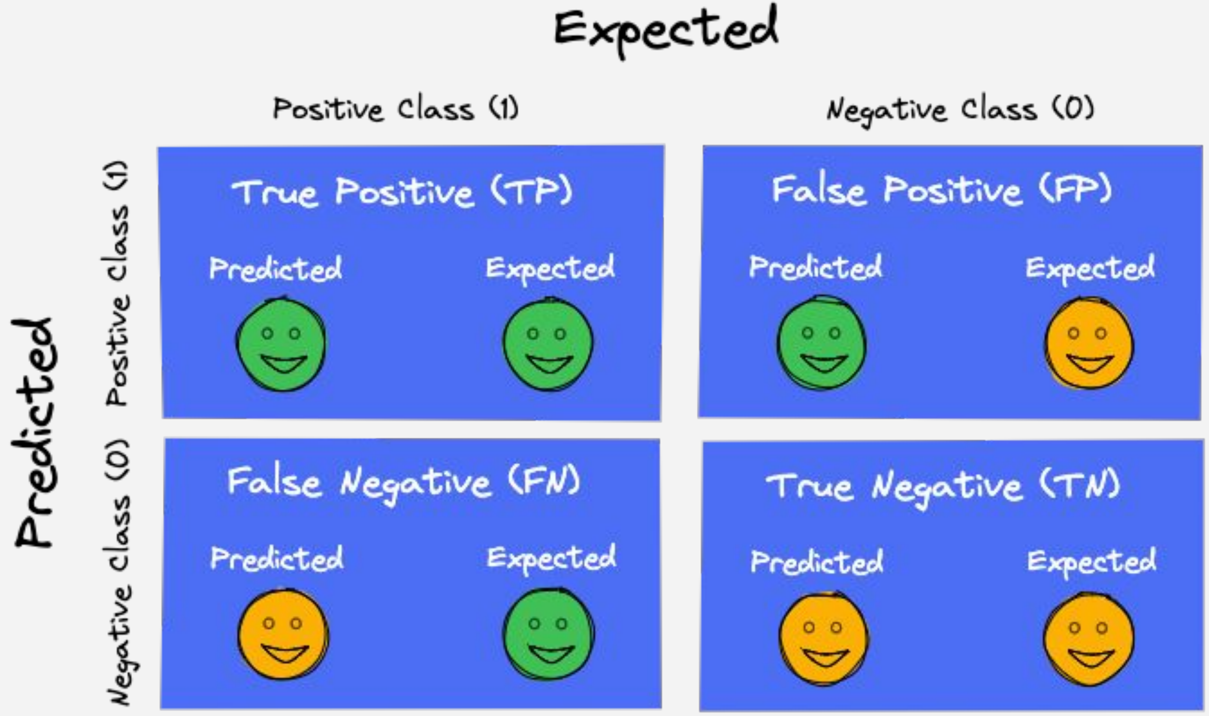


$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{FP + TN}$$

$$\text{G-mean} = \sqrt{\text{Sensitivity} \times \text{Specificity}}$$

Confusion Matrix

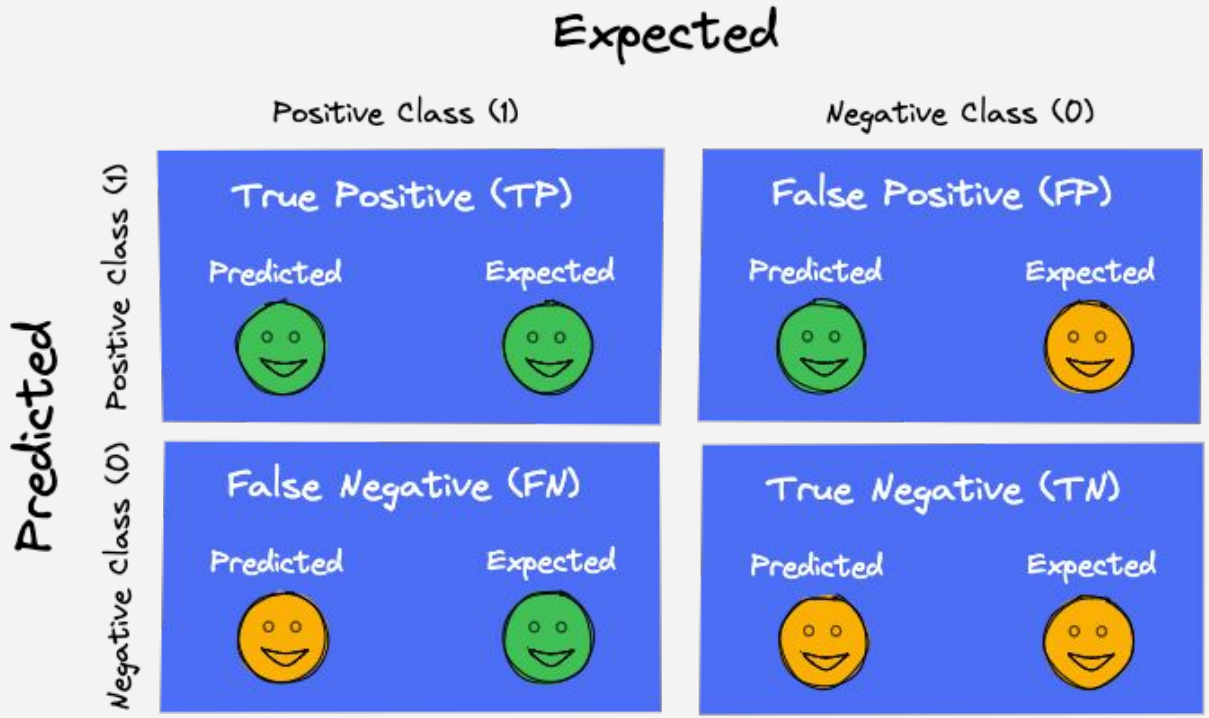


Precision (positive predicte value - PPV) $= \dfrac{TP}{TP + FP}$

Precision (negative predicte value - NPV) $= \dfrac{TN}{TN + FN}$

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

Confusion Matrix



$$\text{Fbeta-measure} = \frac{(1+\beta^2) \times \text{Precision} \times \text{Recall}}{\beta^2 \times \text{Precision} + \text{Recall}}$$

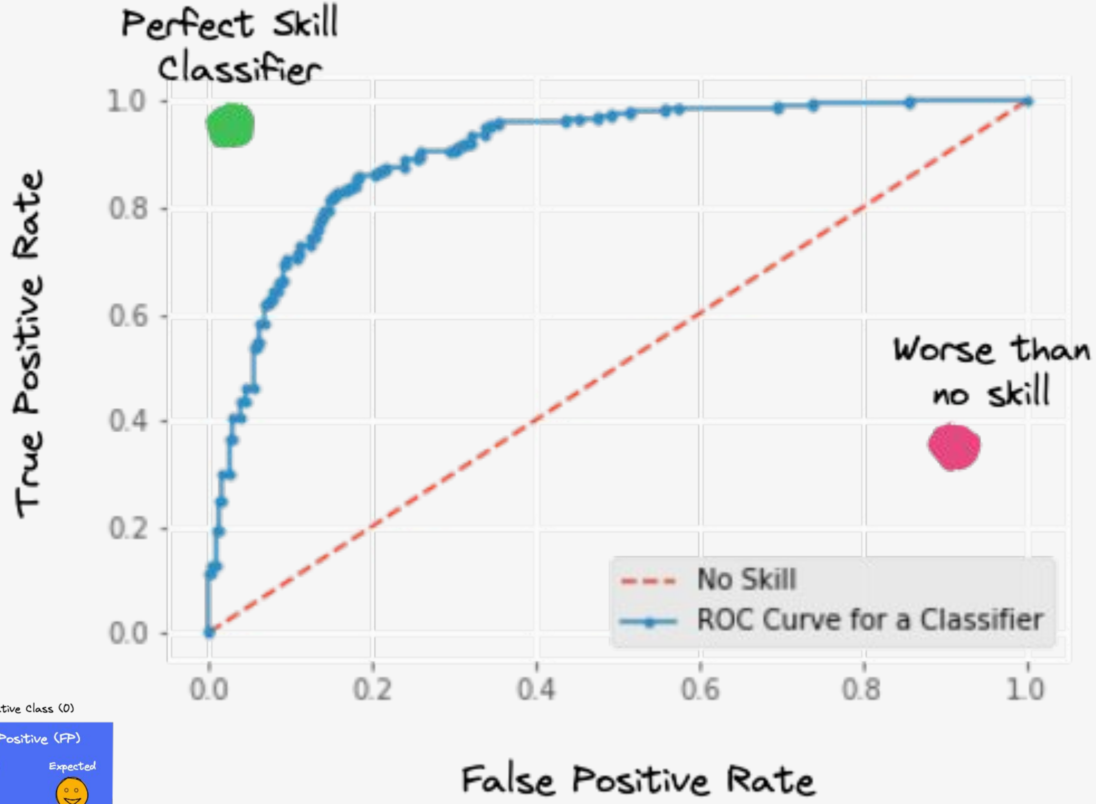
$$\beta == \begin{cases} 0.5, & \text{more weight on precision} \\ 1.0, & \text{balance on weight PR and RE} \\ 2.0, & \text{less weight on precision} \end{cases}$$

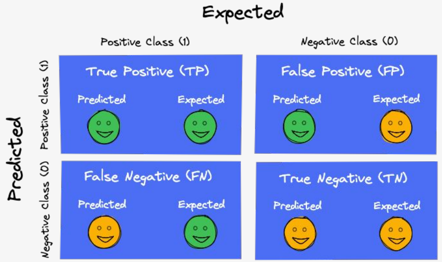
Rank metrics are more concerned with evaluating classifiers based on **how effective** they are at separating classes.

These metrics require that a **classifier predicts a score** or a probability of class membership. From this score, **different thresholds** can be applied to **test the effectiveness of classifiers**. Those models that maintain a good score across a range of thresholds will have good class separation and will be ranked higher.

Receiver Operating Characteristic (ROC)

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

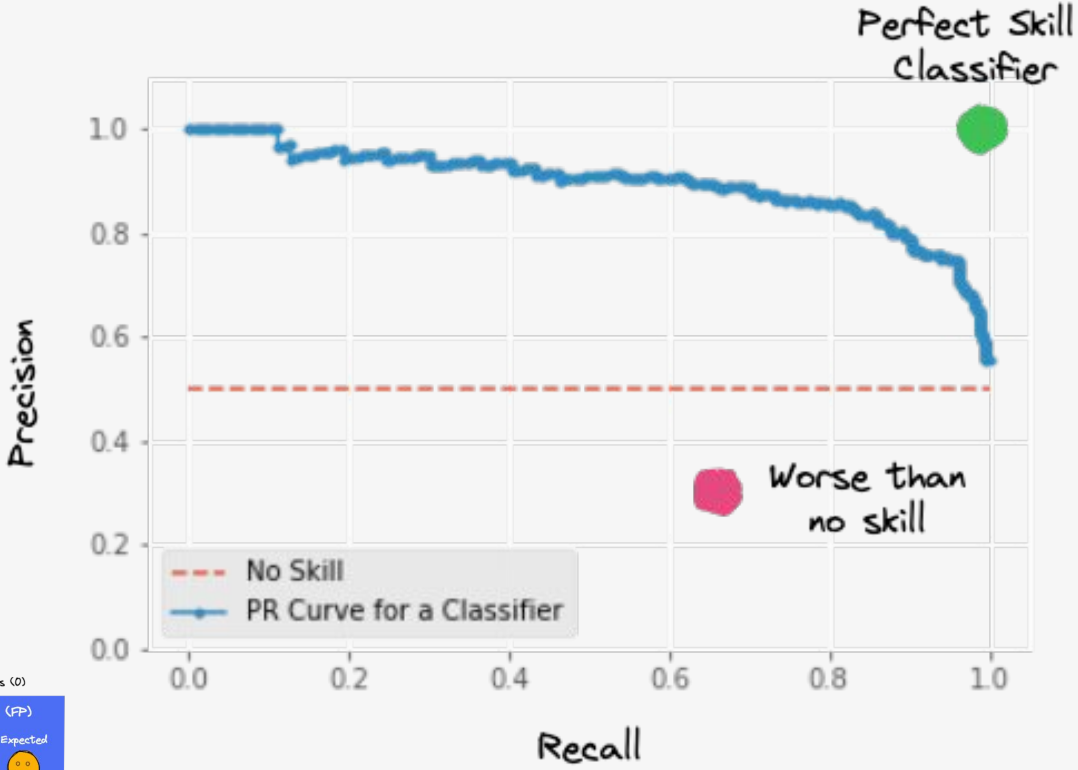


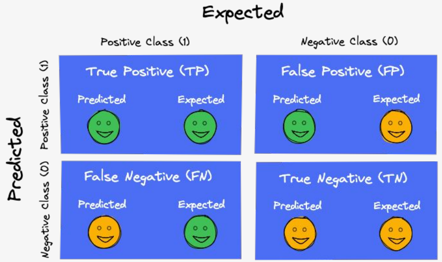


$$FPR = \frac{FP}{FP + TN}$$

Precision-Recall (PR) Curve

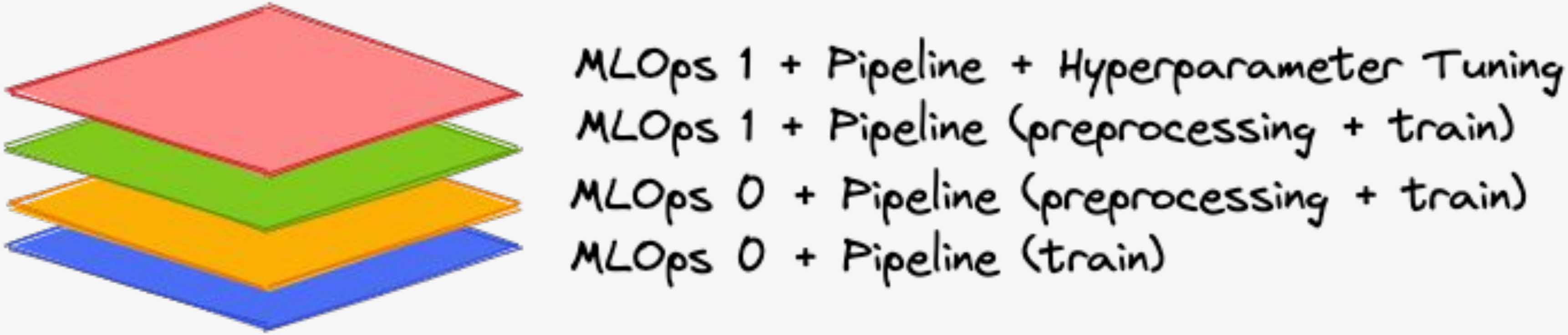
$$\text{Precision} = \frac{TP}{TP + FP}$$



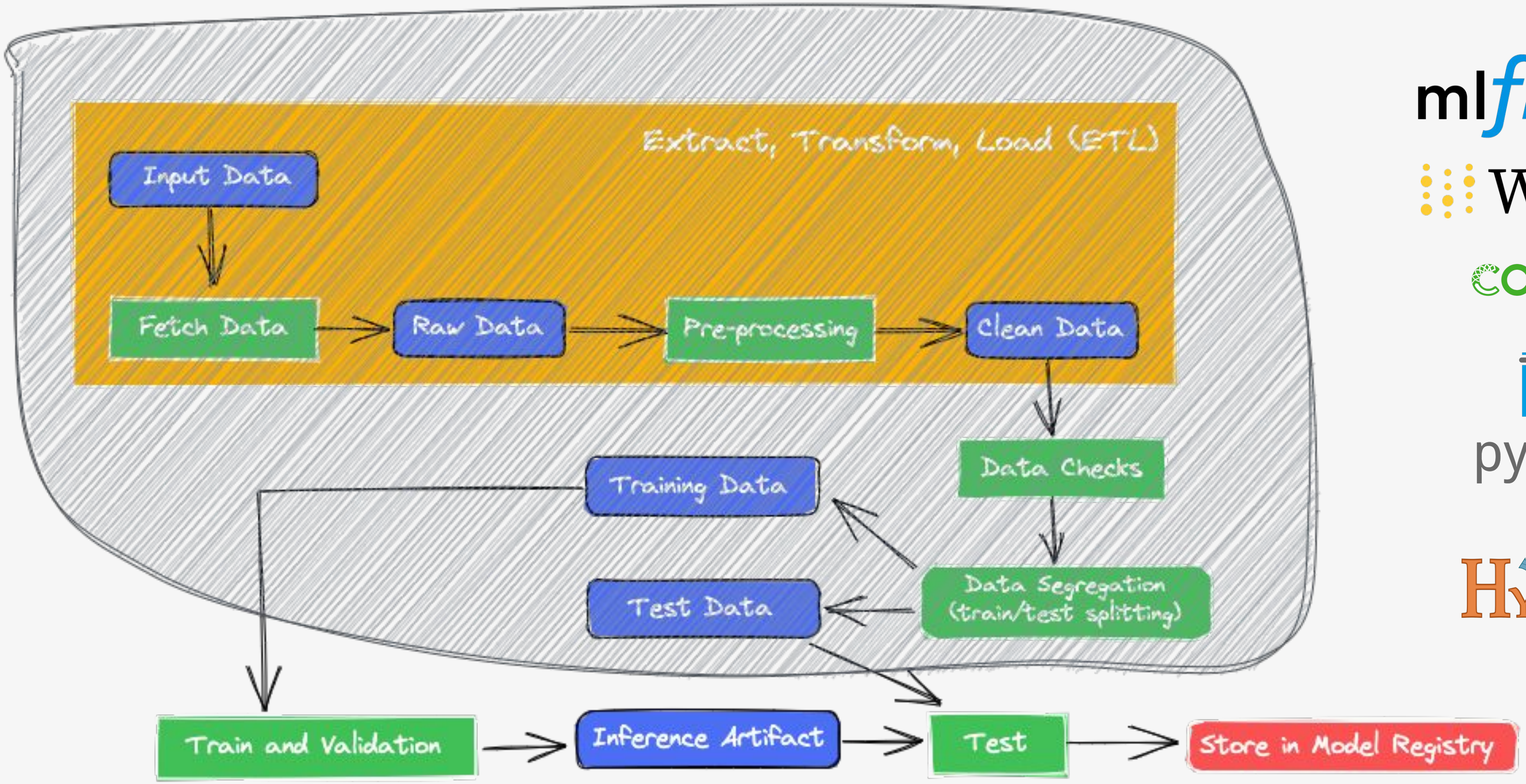


$$\text{Recall} = \frac{TP}{TP + FN}$$

Case Study - Hands on



Final Stage


Extract, Transform, Load (ETL)
Input Data
Fetch Data
Raw Data
Pre-processing
Clean Data
Data Checks
Training Data
Data Segregation (train/test splitting)
Test Data
Train and Validation
Inference Artifact
Test
Store in Model Registry