VSB TECHNICAL UNIVERSITY OF OSTRAVA | FACULTY OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE | DEPARTMENT OF COMPUTER SCIENCE

Introduction to Machine Learning

EFREI/VSB - 2024

Radek Svoboda

November 14, 2024

Evaluation Metrics for Machine Learning

Why Classification Metrics Matter

- Classification metrics help evaluate how effectively a model can categorize instances.
- Different metrics highlight strengths and weaknesses in model performance.
- Understanding these metrics ensures that the model meets the application-specific needs.

Training and Testing Sets

- **Training Set**: The data used to train the model, where it learns patterns and relationships.
- **Testing Set**: Separate data to evaluate model performance, ensuring that it generalizes beyond training data.

Why Use Separate Sets?

- Prevents overfitting by assessing on unseen data.
- Provides a realistic estimate of model accuracy in practical applications.

Classification Metrics

- Evaluates how well a model can classify samples into categories.
- Common metrics include precision, precision, recall, F1 score, and AUC-ROC.
- Each metric provides unique insights, useful in different applications.

Key Classification Metrics

- **Accuracy**: Overall correctness of the predictions.
- **Precision**: Correctness among positive predictions.
- **Recall**: Completeness of positive class predictions.
- **F1-Score**: Balances Precision and Recall.
- **AUC-ROC**: Measures class separation capability.

Confusion Matrix and Terminology

	Predicted Positive	Predicted Negative
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

- The confusion matrix provides a complete summary of the classification performance.
- Key for calculating metrics such as precision, recall, and F1 score.

Confusion Matrix and Terminology

- **True Positive (TP)**: Correctly predicted positive instance.
- **True Negative (TN)**: Correctly predicted negative instance.
- **False Positive (FP)**: Incorrectly predicted as positive (Type I error).
- **False Negative (FN)**: Incorrectly predicted as negative (Type II error).

Classification Example: Disease Detection

Positive Class: Disease Present - **Negative Class**: Disease Absent

Confusion Matrix Outcomes

- **True Positive (TP)**: The patient has the disease, and the test correctly identifies it as positive.
- **True Negative (TN)**: The patient does not have the disease and the test correctly identifies it as negative.
- **False Positive (FP)**: The patient does not have the disease, but the test incorrectly identifies it as positive (Type I error).
- **False Negative (FN)**: The patient has the disease, but the test incorrectly identifies it as negative (Type II error).

Accuracy

- **Accuracy**: Proportion of correct predictions among total predictions.
- Can be misleading if classes are imbalanced, as it does not differentiate between errors.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total samples}}$$

Use Case

- Most effective for balanced datasets where all errors are equally important.
- It may not reflect model performance well when data is imbalanced.

Sample Confusion Matrix

Example Scenario

- **Positive Class**: Disease present
- **Negative Class**: Disease absent

Confusion Matrix

	Predicted Positive	Predicted Negative
Actual Positive	True Positive (TP) = 50	False Negative (FN) = 10
Actual Negative	False Positive (FP) = 5	True Negative (TN) = 35

Calculating Accuracy

- True Positives (TP) = 50
- True Negatives (TN) = 35
- False Positives (FP) = 5
- False Negatives (FN) = 10

Accuracy Calculation

- Substitute values into formula:
- Accuracy $= \frac{50+35}{50+10+5+35} = \frac{85}{100} = 0.85$
- Result: 85% accuracy

Interpretation of Accuracy

- An accuracy of 85% means the model correctly classified 85% of instances.
- Useful as a general overview of model performance.

Limitations

- Accuracy does not reflect the impact of false positives and false negatives.
- Can be misleading in imbalanced datasets where one class dominates.

Precision Definition

- **Precision** is the proportion of true positive predictions among all positive predictions.
- Indicates the accuracy of positive predictions, showing how often positive predictions are correct.

Formula

- **Precision** = $\frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Positives (FP)}}$

Calculating Precision

- True Positives (TP) = 50
- False Positives (FP) = 5

Precision Calculation

- Substitute values into the precision formula:
- Precision $= \frac{50}{50+5} = \frac{50}{55} \approx 0.91$
- **Result**: Precision is approximately 91%.

Interpretation of Precision

- A precision of 91% means that 91% of instances predicted as positive are actually positive.
- Reflects the model's ability to minimize false positives.
- High precision is critical in scenarios where false positives are costly or harmful, such as in medical diagnosis or fraud detection.

Recall Definition

- **Recall** is the proportion of true positive instances among all actual positive instances.
- Measures the model's ability to capture all relevant (positive) instances.

Formula

- **Recall** = $\frac{\text{True Positives (TP)}}{\text{True Positives (TP) + False Negatives (FN)}}$

Calculating Recall

- True Positives (TP) = 50
- False Negatives (FN) = 10

Recall Calculation

- Substitute values into the recall formula:
- Recall $= \frac{50}{50+10} = \frac{50}{60} \approx 0.83$
- **Result**: The recall is approximately 83%.

Interpretation of Recall

- Recall of 83% means the model captures 83% of all actual positive instances.
- Reflects the model's ability to identify as many positive cases as possible.
- High recall is crucial in applications where missing positive instances has severe consequences, such as in disease screening or fraud detection.

F1 Score Definition

- The **F1 Score** is the harmonic mean of Precision and Recall.
- Provides a balance between Precision and Recall, especially useful when you need to consider both false positives and false negatives.

Formula

- **F1 Score** $= 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

Calculating Precision and Recall for F1 Score

- Precision = $\frac{TP}{TP+FP} = \frac{50}{50+5} = \frac{50}{55} \approx 0.91$
- Recall = $\frac{TP}{TP+FN} = \frac{50}{50+10} = \frac{50}{60} \approx 0.83$

F1 Score Calculation

- Substitute values into the F1 Score formula:
- F1 Score = $2 \times \frac{0.91 \times 0.83}{0.91+0.83} = 2 \times \frac{0.7553}{1.74} \approx 0.87$
- **Result**: F1 Score is approximately 87%.

Interpretation of F1 Score

- An F1 score of 87% means that the model is in good balance between precision and recall.
- Especially useful when both false positives and false negatives are important.
- The F1 score is helpful in imbalanced datasets where high precision or high recall alone are insufficient.
- Examples: Medical diagnosis, spam detection, fraud detection.

AUC-ROC Definition

- **ROC (Receiver Operating Characteristic)** curve is a graphical representation of the model's ability to discriminate between classes.
- The curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings.
- **AUC (Area Under the Curve)** is the area under the ROC curve and provides a single value that summarizes the model's performance.

Key Concepts

- TPR (True Positive Rate) = $\frac{TP}{TP+FN}$ = Recall
- FPR (False Positive Rate) = $\frac{FP}{FP+TN}$

Predicted Probability and Threshold in Classification

- The predicted probability is the model's estimate of the likelihood that a given observation belongs to a certain class.
- In binary classification, the predicted probability ranges from 0 to 1.
- Example: A probability of 0.8 means that the model predicts an 80% chance that the observation belongs to the positive class.

- The threshold is a decision boundary that converts predicted probabilities into class labels.
- If the predicted probability is greater than or equal to the threshold, classify as a positive class.
- If the predicted probability is less than the threshold, classify as a negative class.

Example: Classification with Threshold 0.5

Email	Predicted Probability (Spam)	Threshold = 0.5
Email 1	0.8	Spam
Email 2	0.3	Not Spam
Email 3	0.6	Spam
Email 4	0.2	Not Spam

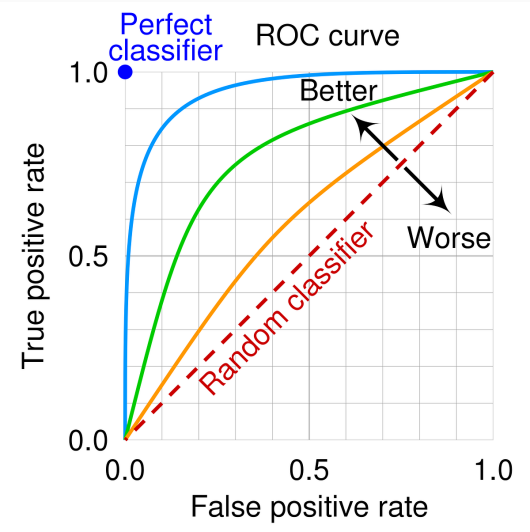
ROC Curve Components

- The x-axis represents the **False Positive Rate (FPR)**.
- The y-axis represents the **True Positive Rate (TPR)** or **Recall**.
- Each point on the curve corresponds to a different classification threshold.

Important Points on ROC Curve

- The diagonal line from (0, 0) to (1, 1) represents a random classifier (50% chance of correct classification).
- The curve above this line indicates a better-than-random classifier.

Example of ROC Curve



What is Cross-Validation?

- Cross-validation is a technique used to assess the performance of a model on unseen data.
- It helps to identify how well the model generalizes to an independent dataset.
- Cross-validation involves splitting the dataset into multiple subsets and using different subsets for training and testing.

Why Use Cross-Validation?

- To avoid overfitting or underfitting.
- To make better use of limited data by training and evaluating the model multiple times.
- To get a more reliable estimate of model performance.

Types of Cross-Validation - K-Fold Cross-Validation

- The data is divided into K subsets (folds).
- The model is trained on $K - 1$ subsets and tested on the remaining subset.
- This process is repeated K times, each time with a different subset as the test set.

Types of Cross-Validation - Leave-One-Out Cross-Validation (LOO)

- A special case of the K-Fold where *K* equals the total number of data points.
- Each data point is used once as the test set and the model is trained on all other data points.

Types of Cross-Validation - Stratified K-Fold Cross-Validation

- Ensures that each fold has a similar distribution of the target variable (especially important for imbalanced datasets).

K-Fold Cross-Validation Example

- Suppose that we have a dataset with 10 data points.
- In 5-Fold Cross-Validation, the dataset will be split into 5 folds.
- Performance is averaged over the five different test sets.

- Fold 1: Train on folds 2-5, test on fold 1.
- Fold 2: Train on folds 1, 3-5, test on fold 2.
- Fold 3: Train on folds 1-2, 4-5, test on fold 3.
- Fold 4: Train on folds 1-3, 5, test on fold 4.
- Fold 5: Train on folds 1-4, test on fold 5.

Evaluating Cross-Validation Results

- After cross-validation, we compute the average of the model's performance metrics (e.g., accuracy, precision, recall) across all folds.
- This gives a more reliable estimate of the performance of the model than a single train-test split.

Advantages

- Less variance in the estimate compared to a single train-test split.
- Makes better use of the available data.

Disadvantages

- More computationally expensive, especially for large datasets.
- May be unnecessary for very large datasets.

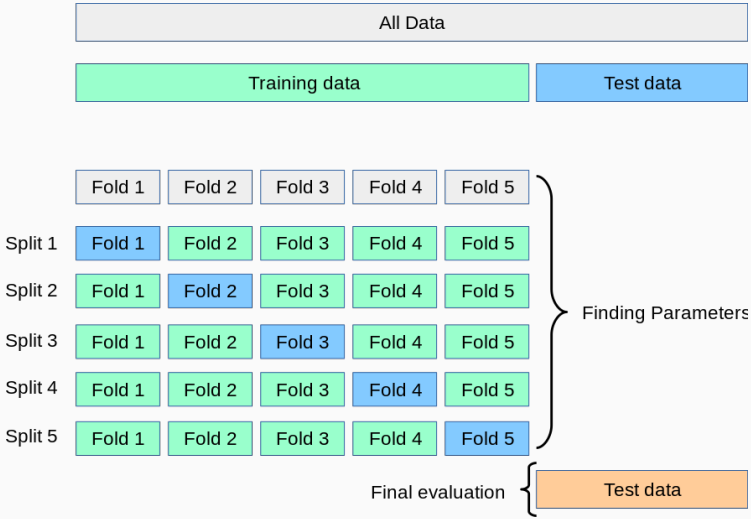
Cross-Validation in Practice

- Split the dataset into K folds.
- For each fold, train the model on the remaining $K - 1$ folds and evaluate it on the current fold.
- Repeat for all folds and compute average performance.

When to Use Cross-Validation

- Use when you have limited data.
- When you want to ensure that your model is not overfitting or underfitting.
- Cross-validation is also useful for model selection and hyperparameter tuning.

k-Fold Cross-Validation



What is Stratified Cross-Validation?

- Stratified Cross-Validation is a variation of K-Fold Cross-Validation.
- It ensures that each fold used for training and testing has a **similar distribution of the target variable** (i.e., the class labels).
- This is especially important when dealing with **imbalanced datasets**, where one class may be underrepresented.
- Unlike standard K-Fold Cross-Validation, Stratified K-Fold ensures that each fold contains approximately the same proportion of each class.

Example of Stratified Cross-Validation

Dataset Composition

- Suppose we have a dataset of 100 samples with the following class distribution:
  - 80% Positive (80 samples) and 20% Negative (20 samples)
- Without stratification, one fold might have too many negative samples, causing training/testing imbalances.

Stratified K-Fold (3-Fold Example)

- **Fold 1:** 27 positive samples, 7 negative samples (80% positive, 20% negative).
- **Fold 2:** 27 positive samples, 7 negative samples.
- **Fold 3:** 26 positive samples, 6 negative samples.

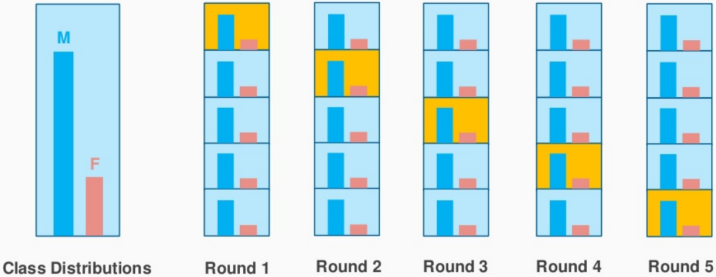
Why Use Stratified Cross-Validation?

- **Maintains Class Distribution**: Ensures that each fold is representative of the overall class distribution, especially important for imbalanced datasets.
- **Better Performance Estimation**: Helps in providing a more reliable estimate of the model's performance, particularly in cases where certain classes are underrepresented.
- **Reduces Bias**: Avoids the risk of overfitting on certain class distributions by ensuring each fold is balanced.

When to Use Stratified Cross-Validation?

- Use when dealing with imbalanced datasets where one class significantly outnumbers the other.
- For binary classification tasks, stratified cross-validation helps ensure that both classes are consistently represented in all folds.
- Especially useful when evaluating classifiers like logistic regression, SVM, or decision trees on imbalanced data.

Stratified Cross-Validation



Questions?