Machine Learning in life sciences.

Classification & Cross-validation

Ignacio Sánchez-Gendriz[1]

[1] Departamento de Engenharia de Computação e Automação (DCA/UFRN)

September 8, 2024

Summary

1. Classification Tasks
2. ML Classifiers
   - Classical Machine Learning (ML) Classifiers
   - Artificial Neural Networks
   - Introduction to the ANN Ecosystem
3. Overfitting & Underfitting
4. Validating ML Classifiers
   - Classification Metrics
   - Cross-Validation Techniques
5. Looking Ahead

Definition and Background

- **Classification:** A supervised learning task that predicts discrete labels or categories from input data.
- **Goal:** To assign new observations to predefined categories based on patterns learned from labeled data (training set).
- **Generalization:** The algorithm should be able to accurately predict the class of samples that were not seen during the training phase (test set).

Dataset partitioned into training and test sets

Samples	Set	X_1	X_2	X_3	X_4	Class
1	Train	0.777	0.000	0.777	0.000	C_1
3	Train	0.587	0.778	0.587	0.778	C_1
5	Train	0.728	0.396	0.728	0.396	C_1
7	Train	0.700	0.944	0.700	0.944	C_1
9	Train	1.000	0.472	1.000	0.472	C_1
11	Train	0.484	0.450	0.484	0.450	C_2
13	Train	0.000	0.557	0.000	0.558	C_2
15	Test	0.342	0.544	0.341	0.544	?
17	Test	0.291	0.711	0.290	0.711	?
19	Test	0.230	0.469	0.230	0.469	?

Importance and Applications in Life Sciences

- **Disease Diagnosis:** Detecting cancer, diabetes, and neurological disorders using gene expression profiles or medical images.
- **Biomarker Identification:** For diagnosis, treatment, and monitoring of diseases, including neurodegenerative conditions.
- **Species Classification:** Using genetic, morphological, or acoustic data for biodiversity assessments, conservation, and behavior studies.
- **Neuroscience Applications:** Classifying neural and behavioral data to study cognitive functions like memory and learning, and to evaluate responses in experiments.

K-Nearest Neighbors (KNN) Algorithm

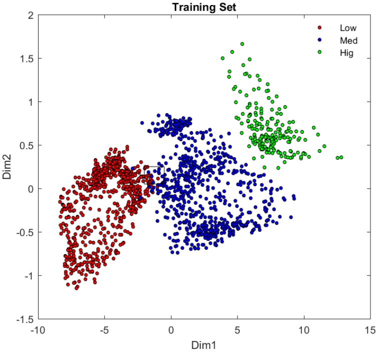
- **Objective:** To classify a new example x_0.
  - Calculate the distance between x_0 and all points in the training set.
  - Identify the classes of the k nearest neighbors.
  - Assign x_0 the majority class among these k neighbors.
- **Distance Measures0**: Mahalanobis, Cosine, Euclidean, othres.
- **Lazy Learning:**
  - KNN is an instance-based algorithm.
  - It does not create a model; it uses the training data directly for each prediction.

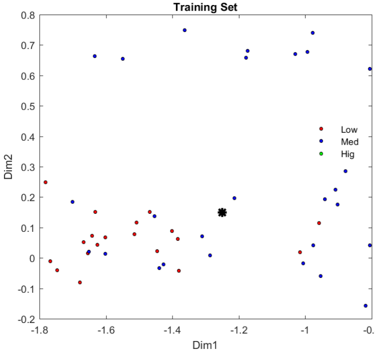
K-Nearest Neighbors (KNN) Algorithm

Problem: Given a dataset of patients categorized into different levels of infection (low, medium, high), the task is to train a classifier to predict the infection level of new patients. (ML-based diagnosis).

K-Nearest Neighbors (KNN) Algorithm

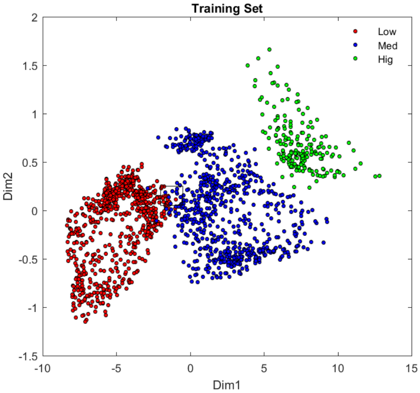
Problem: Given a dataset of patients categorized into different levels of infection (low, medium, high), the task is to train a classifier to predict the infection level of new patients. (ML-based diagnosis).

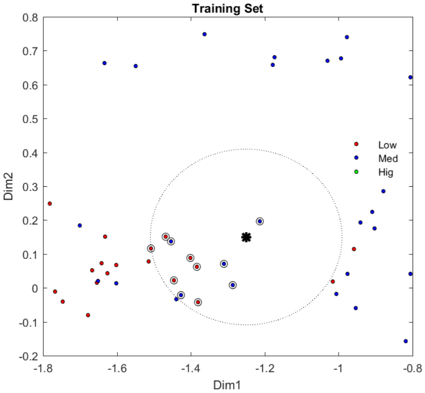




K-Nearest Neighbors (KNN) Algorithm

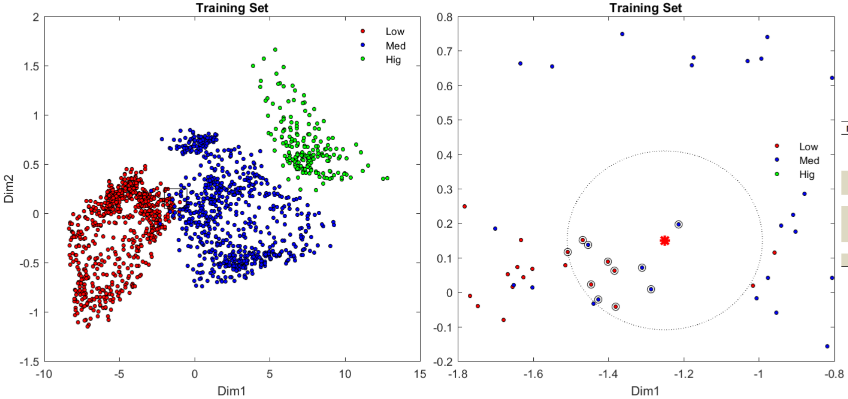
- Select the k nearest neighbors to the new point (e.g., $k = 11$).
- The majority class within this subset is determined.





K-Nearest Neighbors (KNN) Algorithm

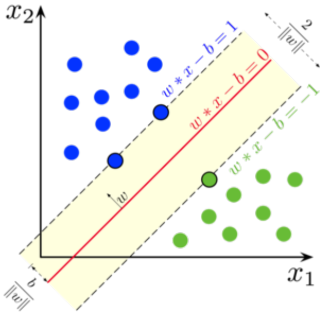
- Assign the majority class among these neighbors as the class of the new sample.



Neighboors	distance	X1	X2	Class
1	0.0597	-1.2125	0.1965	'Med'
2	0.0992	-1.3099	0.0709	'Med'
3	0.1467	-1.2871	0.0080	'Med'
4	0.1603	-1.3842	0.0623	'Low'
5	0.1635	-1.4011	0.0877	'Low'
6	0.2043	-1.4539	0.1370	'Med'
7	0.2179	-1.4679	0.1507	'Low'
8	0.2327	-1.3798	-0.0432	'Low'
9	0.2338	-1.4455	0.0218	'Low'
10	0.2456	-1.4263	-0.0210	'Med'
11	0.2595	-1.5072	0.1153	'Low'

Support Vector Machines (SVM)

- **SVM:** Finds the optimal decision boundary (hyperplane) that maximizes the margin between classes.
- **Support Vectors:** The data points closest to the decision boundary, crucial for defining the margin.
- **Margin:** The distance between the hyperplane and the nearest support vectors, which SVM aims to maximize for better separation.
- In 2D, the boundary is a line. For data with more than 2 dimensions, the boundary is a hyperplane.



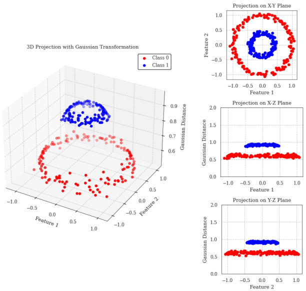
Source: Wikipedia

Support Vector Machines (SVM)

- **Linear SVM:** Originally designed for linear classification by finding the optimal hyperplane that separates classes with the maximum margin.
- **Non-Linear SVM:** SVM can be extended to handle non-linear classification problems using the **kernel trick**, which implicitly maps input data into higher-dimensional spaces.
- **Kernels:** Kernels allow SVM to find complex decision boundaries by transforming data into higher dimensions where it becomes linearly separable.
- **Polynomial Kernel** and **Radial Basis Function (RBF) Kernel** are comuns strategies used in this cases.

Support Vector Machines (SVM)

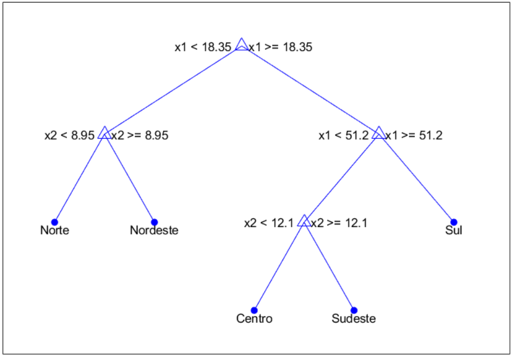
As an example, we applied a Gaussian transformation to project 2D data of concentric circular clusters into 3D. This was achieved by adding a third dimension using a Gaussian function based on the distance from each point to the origin. This transformation lifts the data into a higher-dimensional space, creating a separable structure and illustrating how SVM can handle non-linear data by making it linearly separable in 3D through the *kernel trick*.

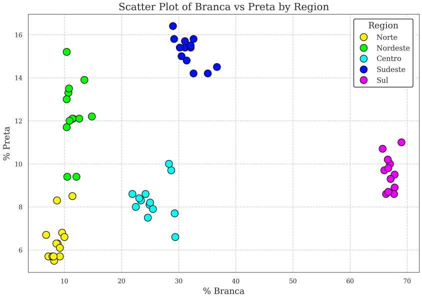


Source: Notebook

Decision Trees

- **Decision Trees:** Classify data by splitting it into branches based on feature values.
- **Intuitive:** Easy to understand and visualize with simple decision rules.





Ensemble Methods - Fundamentals

- **Ensemble Methods:** Combine multiple models, known as ”weak learners,” to enhance overall performance and robustness.
- **Key Concept:** By aggregating predictions, ensemble methods reduce variance, correct biases, and generally improve accuracy compared to any single model.
- **Why Ensembles Work:** They leverage the strengths of diverse models, where the errors of some models are compensated by the correct predictions of others.
- **Theoretical Foundation:** A group of weak learners, which individually perform slightly better than random, can be combined into a strong learner with significantly improved accuracy.
- **Example:** Like a committee of experts making a decision, ensemble methods combine the predictions of multiple models to achieve a more reliable outcome.

Types of Ensembles

- **Bagging (Bootstrap Aggregating):** Reduces variance by training multiple instances of the same algorithm on different subsets of the data. Example: *Random Forests*.
- **Boosting:** Sequentially trains models, where each model focuses on correcting the errors of the previous ones, thereby reducing bias. Examples: *AdaBoost, Gradient Boosting*.
- **Stacking:** Combines multiple models using a meta-model that learns the best way to aggregate the predictions of base models.

Ensemble Methods - Random Forests

- **Random Forests:** An ensemble method that combines multiple decision trees to improve stability, accuracy, and robustness. Each tree is built using a random subset of the training data and a random subset of features at each split, which reduces overfitting and enhances the model's generalizability.
- **Key Mechanisms:**
  - **Bootstrap Sampling (Bagging):** Each decision tree is trained on a randomly sampled subset of the training data with replacement, ensuring diversity among the trees.
  - **Random Feature Selection:** At each split in a tree, a random subset of features is considered rather than evaluating all features, which decorrelates the trees and improves the ensemble's performance.

Advantages and Applications of Random Forests

- **Advantages:**
  - **Reduces Overfitting:** By averaging multiple trees, random forests minimize overfitting that is common in individual decision trees.
  - **Enhances Predictive Performance:** The aggregation of diverse trees improves both classification and regression accuracy.
  - **Robust to Noise:** Less sensitive to noise in the data due to the averaging of predictions across many trees.
  - **Feature Selection:** Provides a measure of feature importance, allowing random forests to be used for identifying the most relevant features in the dataset.
- **Applications:**
  - Widely used across various fields, including finance, healthcare, and bioinformatics.
  - Common tasks include classification, regression, and feature importance ranking.

Introduction to Neural Networks

Artificial Neural Networks (ANNs) are computational models inspired by the neural structure of the human brain. They are designed to learn and store knowledge through interconnected processing units, known as artificial neurons, which mimic the functioning of biological neurons.

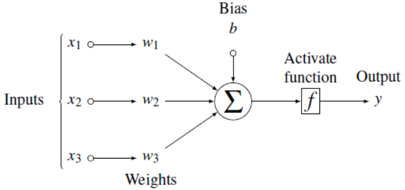
- **Concept:** ANNs consist of layers of artificial neurons that simulate the activation and connectivity of biological neurons. Each neuron processes inputs, applies a weight (synaptic strength), and transmits the output to the next layer, enabling complex pattern recognition and decision-making.
- **Structure:** Neurons are organized into layers: an input layer, one or more hidden layers, and an output layer. Connections between neurons are represented by weights, which are adjusted during training to minimize prediction errors.
- **Learning Mechanism:** ANNs learn by adjusting weights through algorithms like backpropagation, which minimizes the difference between predicted and actual outputs by iteratively updating the weights.

McCulloch-Pitts Model

- **Historical Context:** Introduced by Warren McCulloch and Walter Pitts in 1943, it was the first mathematical model of a neuron, foundational for neural network theory.
- **Concept:** Models a neuron as a binary threshold unit that sums weighted inputs and fires (outputs 1) if the sum exceeds a threshold; otherwise, it outputs 0.
- **Functioning:**
  - Uses binary inputs with associated weights.
  - Outputs 1 if the weighted sum meets or exceeds the threshold; otherwise, outputs 0.
- **Significance:** Demonstrated how basic logic gates (AND, OR, NOT) can be implemented with neurons, paving the way for more complex neural models.

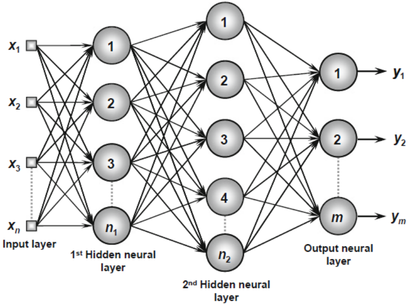
Perceptron Model

- **Perceptron:** A fundamental linear classifier that serves as the basic unit of neural networks.
- **Function:** Combines inputs linearly with weights, adds a bias, and applies an activation function to produce the output.
- **Usage:** Best suited for linearly separable data, laying the groundwork for more complex neural network structures.



Multi-Layer Perceptron (MLP)

- **MLP:** An extension of the perceptron that includes one or more hidden layers and non-linear activation functions.
- **Function:** Each layer transforms inputs through weighted sums and activation functions, allowing the network to learn complex, non-linear patterns.
- **Importance:** Crucial for deep learning, MLPs can approximate a wide range of functions, making them powerful for classification and regression tasks.



Overview of ANN Ecosystem

- **Role:** ANNs are pivotal in AI, learning complex patterns from data and forming the core of deep learning models.
- **Key Features:**
  - **Flexibility:** Adaptable to tasks like classification, regression, and generative modeling.
  - **Scalability:** Handles large datasets, ideal for big data.
  - **Feature Extraction:** Deep NNs can learn features from raw data, minimizing manual effort.
- **Applications:**
  - **Image Recognition:** Object detection and segmentation.
  - **Speech Processing:** Voice recognition and speech-to-text.
  - **Predictive Modeling:** Forecasting and risk assessment in various sectors.

Key Architectures in ANN

- **Convolutional Neural Networks (CNNs):**
  - **Purpose:** Handles grid-like data (e.g., images).
  - **Mechanism:** Convolutional layers extract spatial patterns.
  - **Applications:** Image classification, object detection.
- **Recurrent Neural Networks (RNNs):**
  - **Purpose:** Processes sequential data.
  - **Mechanism:** Loops maintain sequence information.
  - **Applications:** NLP, time-series forecasting.
- **Other Architectures:**
  - **GANs:** Generate new data samples.
  - **Autoencoders:** Dimensionality reduction, denoising.

Specific Applications of ANN Architectures

- **Medical Imaging:**
  - Detects and classifies diseases from scans (X-rays, MRIs).
  - Aids in early diagnosis and personalized treatment.
- **Species Classification:**
  - Identifies species using audio or images.
  - Supports biodiversity and conservation efforts.
- **Financial Forecasting:**
  - Predicts stock prices and market trends.
  - Improves decision-making in investments.
- **Autonomous Vehicles:**
  - Performs object detection and navigation.
  - Enhances safety in self-driving technologies.

Overfitting

- **Definition:** Overfitting occurs when a model learns the noise and details of the training data to the extent that it negatively impacts the model's performance on new, unseen data.
- **Symptoms:** High accuracy on training data but poor generalization to test data.
- **Visual Example:** The learning curve shows a large gap between training and validation performance.
- **Causes:** Excessively complex models, too many features, or insufficient training data.

Underfitting

- **Definition:** Underfitting occurs when a model is too simple to capture the underlying patterns in the data.
- **Symptoms:** Poor performance on both training and test data.
- **Visual Example:** The learning curve shows both training and validation errors remaining high.
- **Causes:** Overly simplistic models, inadequate feature selection, or insufficient model training.

Impact on Model Performance

- **Overfitting:** Leads to a model that is highly tuned to the training data but fails to generalize to new data, resulting in poor predictive performance.
- **Underfitting:** Results in a model that does not capture the underlying trends, providing limited predictive power.
- **Balancing Act:** The goal is to find the optimal complexity where the model performs well on both training and unseen data.
- **Mitigation Strategies:** Regularization, cross-validation, pruning, simplifying the model, or gathering more data.

Introduction to Model Validation

- **Objective:** To evaluate the performance of classifiers, we need well-defined metrics.
- **Challenge:** Metrics must be computed in a manner that avoids pitfalls like overfitting or underfitting.
- **Solution:** Systematic evaluations through cross-validation techniques help in assessing model generalizability.

Confusion Matrix

- **Definition:** A table that summarizes the performance of a classification model by comparing predicted vs. actual values.
- **Components:**
  - **True Positives (TP):** Correctly predicted positive cases.
  - **True Negatives (TN):** Correctly predicted negative cases.
  - **False Positives (FP):** Incorrectly predicted as positive (Type I error).
  - **False Negatives (FN):** Incorrectly predicted as negative (Type II error).
- **Importance:** Provides insight into the types of errors made by the classifier.

Accuracy, Precision, Recall, and F1-Score

- **Accuracy:** The ratio of correctly predicted instances out of all instances.
- **Precision:** The ratio of correctly predicted positive observations to the total predicted positives.
- **Recall (Sensitivity):** The ratio of correctly predicted positive observations to all actual positives.
- **F1-Score:** The harmonic mean of precision and recall, balancing both concerns.
- **When to Use:**
  - Accuracy is suitable when classes are balanced.
  - Precision is critical when the cost of false positives is high.
  - Recall is important when the cost of false negatives is high.
  - F1-Score is used when you need a balance between precision and recall.

Concepts and Importance of Cross-Validation

- **Purpose:** To assess how a model generalizes to an independent dataset, preventing overfitting.
- **Process:** The data is split into subsets; the model is trained on some subsets and tested on others.
- **Benefit:** Provides a more robust evaluation of the model's performance compared to a single split of data.

Hold-Out Method

- **Definition:** A simple technique where the dataset is split into training and test sets.
- **Usage:** Fast and straightforward, but performance estimates can vary depending on how the data is split.
- **Limitations:** Can suffer from high variance if the split is not representative.

K-Fold Cross-Validation

- **Definition:** The dataset is divided into K equally sized folds. The model is trained on K-1 folds and tested on the remaining fold, repeated K times.
- **Advantage:** Reduces variance in performance estimates by averaging results across all folds.
- **Typical Choice:** K is often set to 5 or 10.

Leave-One-Out Cross-Validation (LOOCV)

- **Definition:** A special case of K-fold where K equals the number of samples, leaving one sample out in each iteration.
- **Advantage:** Utilizes all data points, but can be computationally expensive for large datasets.
- **When to Use:** Effective when the dataset is small, ensuring maximum data utilization.

Resampling and Bootstrapping

- **Definition:** Techniques that involve repeatedly sampling from the dataset with replacement to create multiple training sets.
- **Purpose:** Allows estimation of model accuracy and stability, particularly useful when data is limited.
- **Benefit:** Provides insights into the variability of model performance.

Continuing with Practical Applications for the Next Class

- **Next Steps:** In the next class, we will extend our understanding by applying the concepts learned so far.
- **Focus:** Hands-on practice with Python, utilizing real datasets to deepen your grasp of machine learning classifiers and validation techniques.

Practical Examples in Python

- **Objective:** Develop practical examples to reinforce theoretical concepts.
- **Activities:** We will work with real datasets, enabling you to see how classifiers perform in real-world scenarios.
- **Outcome:** Gain a deeper understanding of model performance, validation techniques, and the application of machine learning in life sciences.

Preparation for the Next Class

- **Reminder to Students:** Review the material covered in previous classes to familiarize yourself with the concepts discussed.
- **Hands-On Practice:** Use the datasets provided or bring your own data to apply the techniques learned. This is an excellent opportunity to put theory into practice.
- **Engagement:** Engage with the material actively—practice coding, explore datasets, and prepare questions for deeper discussion in the next class.