

# Comparison Across Different Supervised Learning Algorithms For Classification Tasks

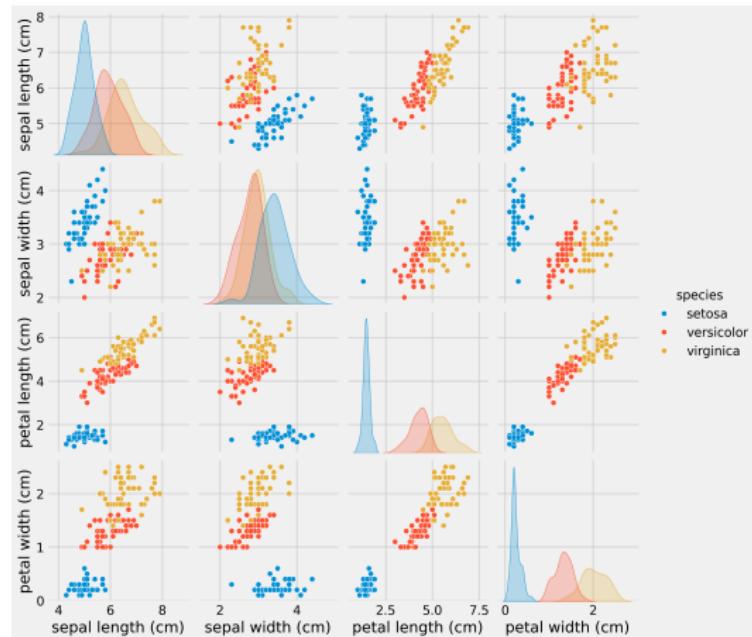
Julian Sampedro

2020 (updated 2025)

# Introduction

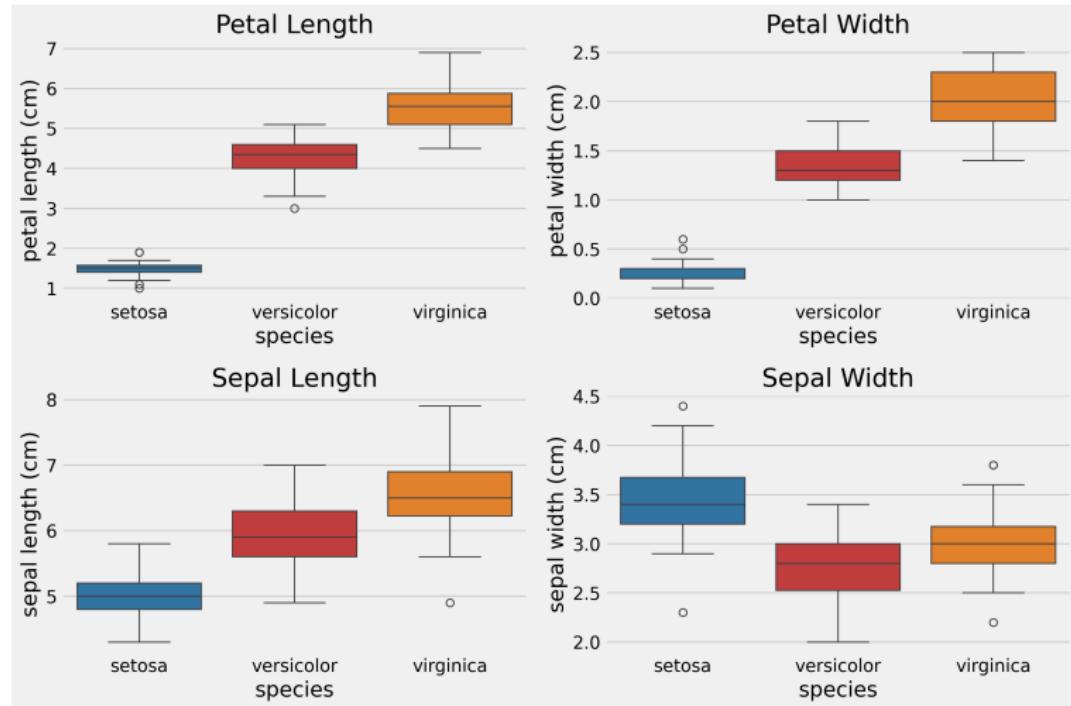
- (i) Analysis of the Iris dataset in Python.
- (ii) Supervised learning algorithms for classification with sklearn: Random Forest (RF), Multinomial Logistic Regression (MLR), Support Vector Machines (SVM), K-Nearest Neighbors (KNN) , XGBoost (XGB), Sequential Neural Network (SNN).
- (iii) Comparison of the performances of the different models and fine-tuning.
- (iv) Technical and domain conclusions.

# EDA - Scatterplots and Density Estimators (1/4)



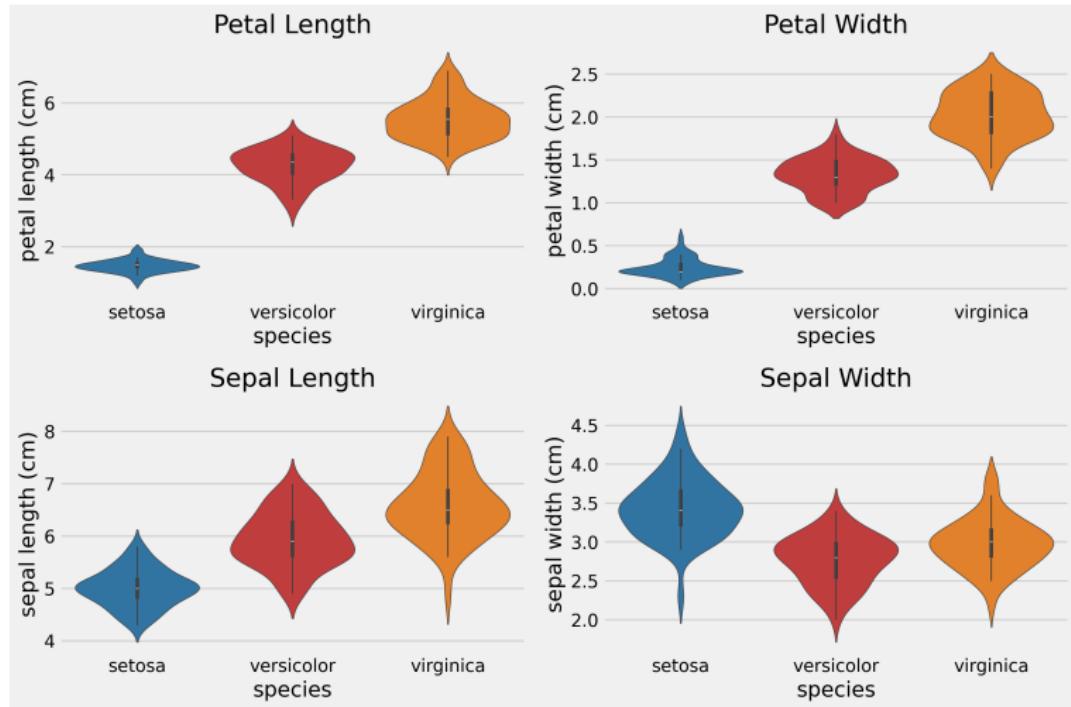
**Note:** Pairplot method in Python to display visualize distributions all at once.

## EDA - Boxplots (2/4)



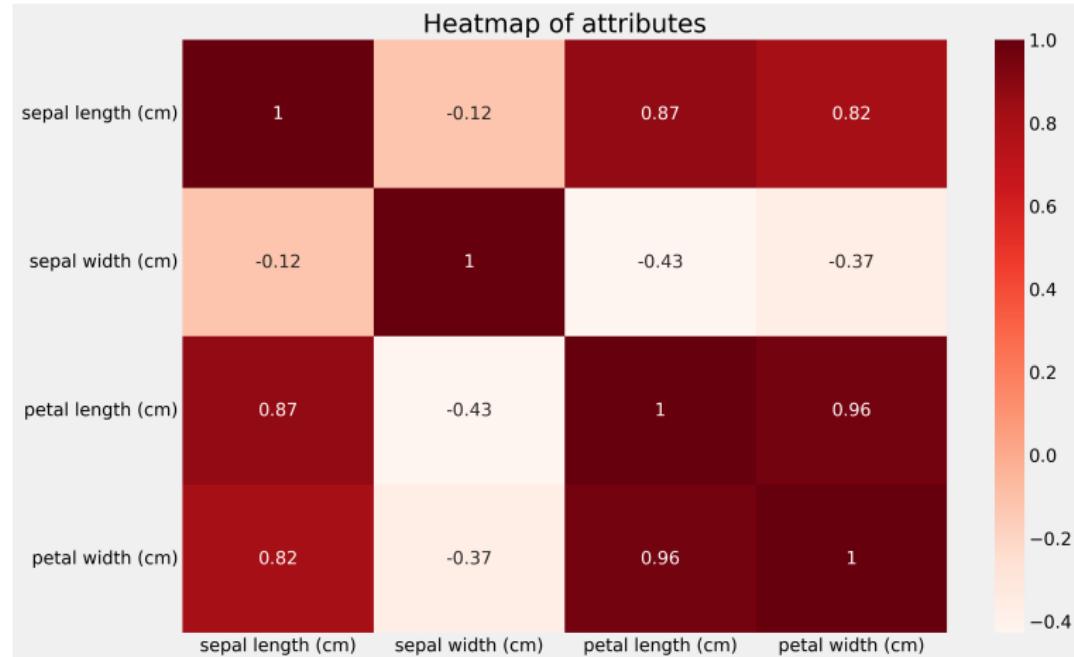
**Note:** Representation of the individual distributions for different attributes with extreme observations and the median.

# EDA - Violin Plots (3/4)



**Note:** Better representation of the shape of the individual distributions of the attributes.

# EDA - Heatmap (4/4)



**Note:** Heatmap of the correlations among attributes.

# Libraries and dataset

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import datasets
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from xgboost import XGBClassifier
from sklearn.svm import SVC

from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.model_selection import cross_val_score
from sklearn.metrics import accuracy_score, precision_score, recall_score,
    f1_score
import shap
from shap import plots
from keras.models import Sequential
from keras.layers import Dense
from keras.utils import to_categorical

from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_percentage_error
```

# Models

```
# RR
rf_model = RandomForestClassifier(n_estimators=100) #create

# MLR
lr_model = LogisticRegression(solver= 'lbfgs', max_iter=400, multi_class = 'multinomial') # create

# SVM
svm_model = SVC(kernel = 'linear', random_state = 0) # create

# KNN
knn_model = KNeighborsClassifier(n_neighbors = 5, weights = 'distance') # create

# XGBoost
xgb_model = XGBClassifier(n_estimators=100, learning_rate= 0.3) # create

# Sequential NN
model=Sequential()
model.add(Dense(100,activation='relu',input_shape=(4,)))
model.add(Dense(3,activation='softmax'))
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

**Note:** Other classification exists in scikit-learn, for example Naive Bayes or Multi-Layers Perceptron.

# Performance Metrics for Classification Algorithms (1/2)

## Confusion Matrix (3-Class Classification)

	Predicted 1	Predicted 2	Predicted 3
Actual 1	$C_{11}$	$C_{12}$	$C_{13}$
Actual 2	$C_{21}$	$C_{22}$	$C_{23}$
Actual 3	$C_{31}$	$C_{32}$	$C_{33}$

	Pred 1	Pred 2	Pred 3	Sum
Actual 1	$TP_1$	$FN_1$	$FN_1$	$C_{11} + C_{12} + C_{13}$
Actual 2	$FP_1$	$C_{22}$	$C_{23}$	$C_{21} + C_{22} + C_{23}$
Actual 3	$FP_1$	$C_{32}$	$C_{33}$	$C_{31} + C_{32} + C_{33}$
Sum	$C_{11} + C_{21} + C_{31}$	$C_{12} + C_{22} + C_{32}$	$C_{13} + C_{23} + C_{33}$	$N$

$$TP_1 = C_{11} \text{ and } TN_1 = C_{22} + C_{23} + C_{32} + C_{33}$$

# Performance Metrics for Classification Algorithms (2/2)

## Confusion Matrix (Binary Classification)

	Predicted Positive	Predicted Negative
Actual Positive	$TP$	$FN$
Actual Negative	$FP$	$TN$

## Core Metrics

- **Accuracy:**  $\frac{TP+TN}{TP+TN+FP+FN}$
- **Precision:**  $\frac{TP}{TP+FP}$
- **Recall (Sensitivity):**  $\frac{TP}{TP+FN}$
- **F1 Score (most important):**

$$2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \text{ or}$$

$$\frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$$

## Other Relevant Metrics

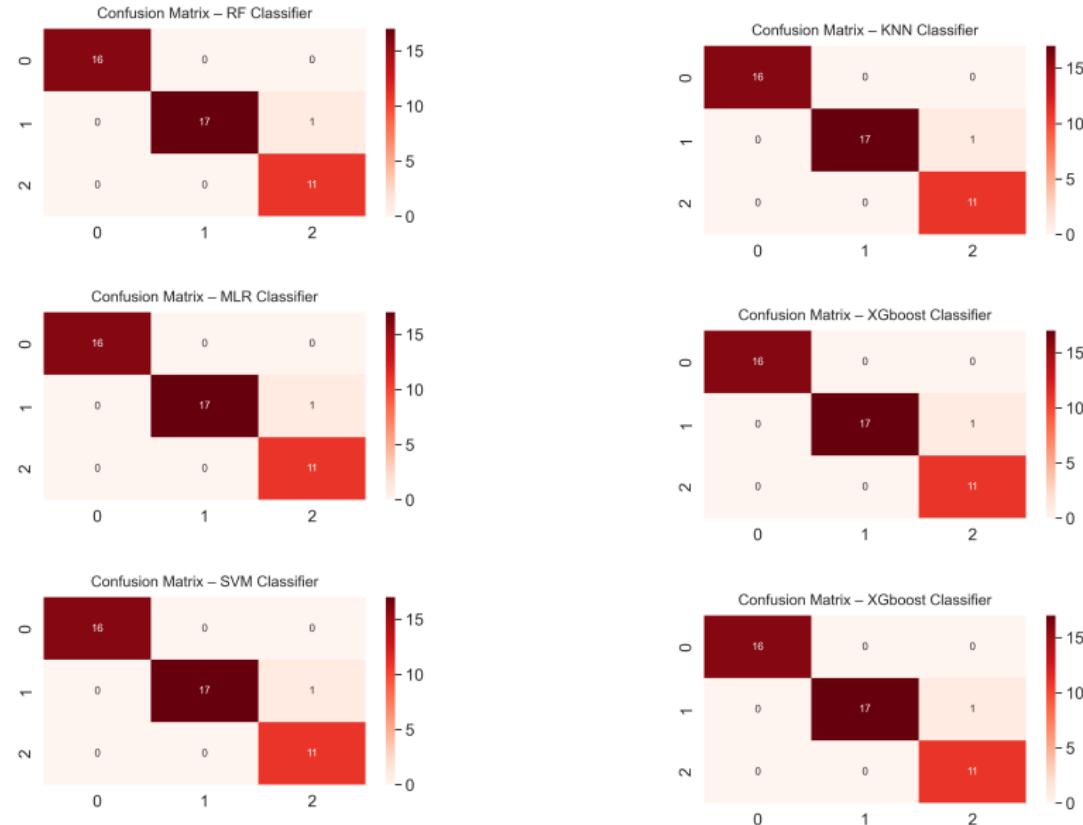
- **Specificity:**  $\frac{TN}{TN+FP}$
- **Balanced Accuracy:**  $\frac{\text{Sensitivity} + \text{Specificity}}{2}$
- **ROC Curve & AUC:** Discrimination ability across thresholds.

# Performance Comparison

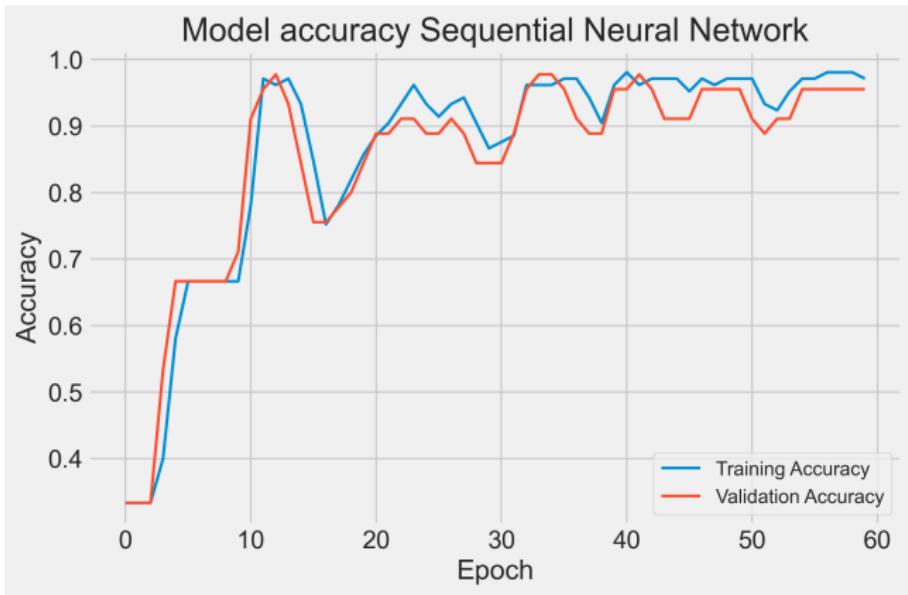
---

Algorithm	Acc. train	Acc. test	Precision	Recall	F1
<i>RF</i>	97.78	100	0.978	0.978	0.978
<i>MLR</i>	97.78	98.1	0.978	0.978	0.978
<i>SVM</i>	97.78	98.1	0.978	0.978	0.978
<i>KNN</i>	97.78	100	0.978	0.978	0.978
<i>XGboost</i>	97.78	100	0.978	0.978	0.978
<i>Sequential NN</i>	96.19	97.8	0.979	0.978	0.978

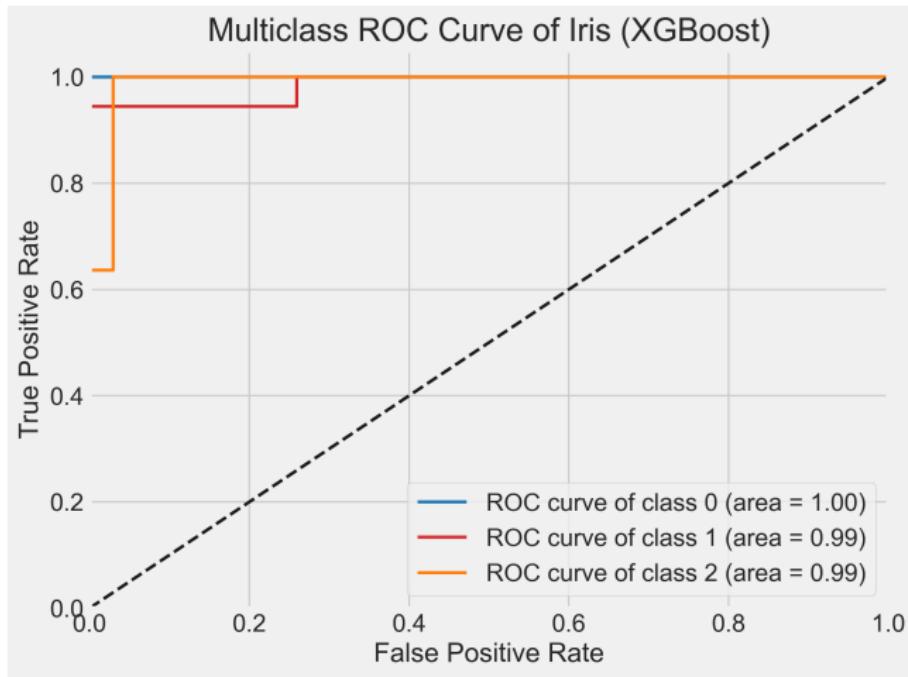
# Confusion Matrix



# Sequential Neural Network accuracy



# Roc Curves



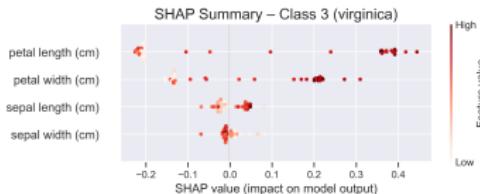
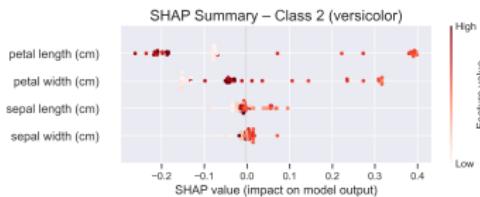
# Parameter Optimization Methods

Many family of optimization methods to fine-tune our models:

- **Grid Search**, which exhaustively tests all combinations of specified hyperparameter values to find the best model.  
*GridSearchCV()*.
- **Random Search** which samples a fixed number of random hyperparameter combinations from specified distributions to efficiently search large spaces.  
*RandomizedSearchCV()*.
- **Bayesian Search** which use past evaluation results to model the performance function and selects promising hyperparameters iteratively to optimize efficiently.  
*BayesSearchCV()*.

# SHAP Values

- SHAP plots of feature contributions to class predictions.
- Positive SHAP values increase the probability of the class and negative SHAP values decrease the probability of the class.
- Each plot corresponds to the following classes (from top to bottom): Setosa, Versicolor, Virginica.
- Model: Random Forest.
- Interpretation example: Petal length and petal width (both around  $-0.18$ ) strongly decrease the probability that an observation is setosa (setosa has small petals).



# Technical Conclusions

- (i) All models achieved high accuracy (train and test above 97%), which shows that the dataset is well-behaved and suitable for standard classification algorithms.
- (ii) Random Forest, XGBoost, and KNN slightly outperformed other models.
- (iii) Asequential Neural Network performed comparably but required more careful parameter tuning.
- (iv) Feature importance and SHAP values show clear interpretable contributions, though they make more sense in a regression context.

# Domain Conclusions

- (i) Petal measurements are the most critical biological indicators whereas sepal measurements contribute less to classification.
- (ii) One could argue that measuring only the most informative features (petal length and width) may be sufficient for accurate classification if resources for data acquisition are limited.
- (iii) Model interpretability methods (here for example SHAP values) help validate domain hypotheses.

# References

Brownlee, J. (2019). Master Machine Learning Algorithms.

Ressources on <https://www.python.org/>