Machine Learning Education Environment

Dr. Alp Arslan Bayrakçı

Computer Engineering

Gebze Teknik Üniversitesi Mühendislik Fakültesi, Lisans Bitirme Projeleri Sergisi, 25 Haziran 2025

Summary

This project, the ”ML Education Environment,” addresses the challenges of machine learning experimentation by providing an interactive, web-based platform designed to demystify the algorithm training and evaluation process. It targets the common barriers faced by students and practitioners, such as complex environment setups and the need for extensive boilerplate code, which can obscure the nuanced impact of different models, datasets, and hyperparameters.

The system is architected with a decoupled frontend and backend, containerized using Docker for easy deployment and accessibility. A model factory design pattern dynamically dispatches tasks to the appropriate machine learning model implementation, which includes a suite of classification algorithms like DT, LR, SVM, K-NN, and ANN. A key feature is the caching mechanism that stores the results of completed experiments in a JSON file, avoiding redundant computations by retrieving past results for identical configurations, thus speeding up the user workflow.

MATERIALS AND METHODS

The project was developed with a client-server architecture using modern web technologies.

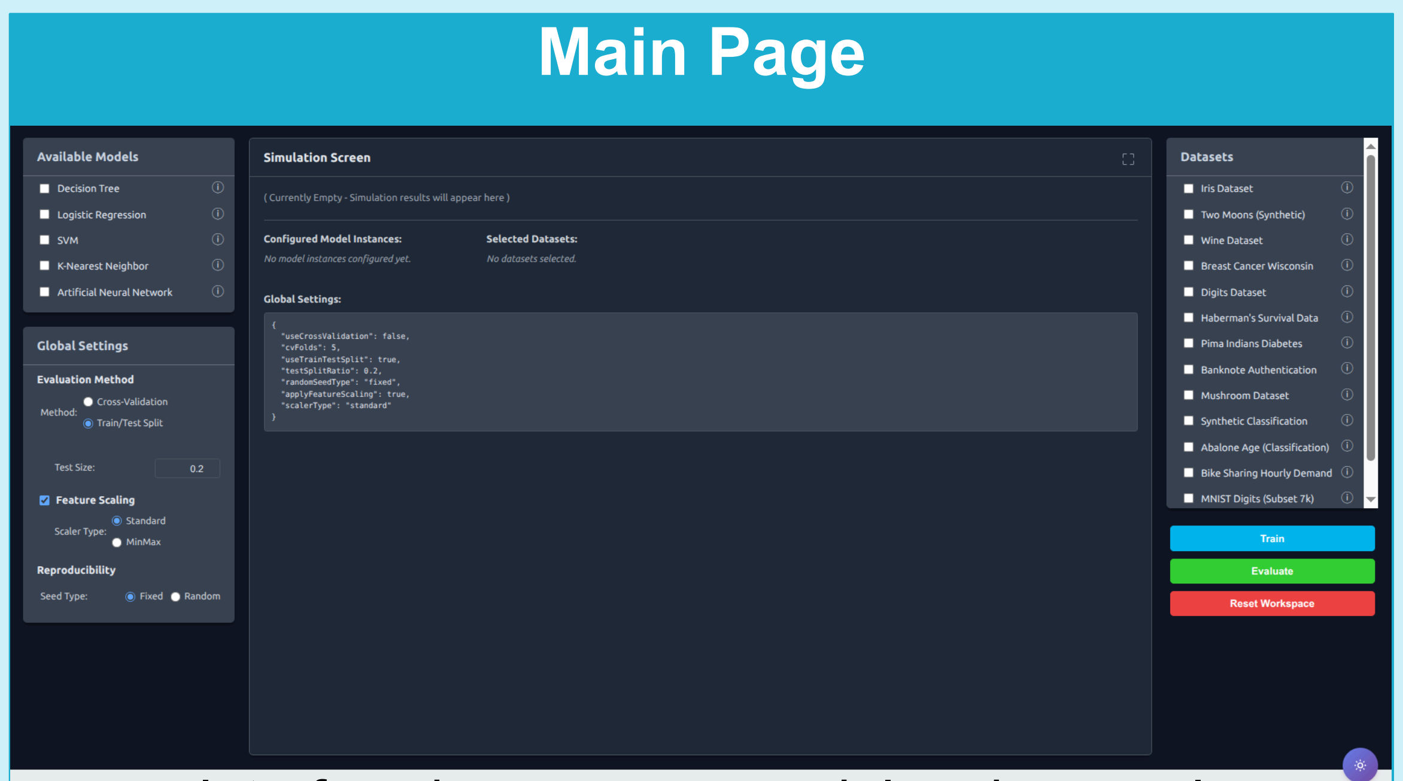
Backend:

- Framework: A high-performance API server was created using Python-based FastAPI.
- ML Models: LR, KNN, SVM, DT, ANN classification models were integrated using scikit-learn library.
- Data processing: The CSV format data uploaded by the user was processed and made suitable for the models using the Pandas library.
- Containerization. Portability and easy installation were provided by transforming the application’s services into containers with Docker.

Frontend:

- Library: A user-friendly, dynamic and interactive interface was designed with React.js
- Interface Components: Contains components where users can load a dataset, select the model and its parameters, track the training process, and view the results (metrics, comparison charts, generated code). Metrics are visualized with Chart.js

Main Page


Available Models
☐ Decision Tree
☐ Logistic Regression
☐ SVM
☐ K-Nearest Neighbor
☐ Artificial Neural Network
Global Settings
Evaluation Method
Method: ○ Cross-Validation ● Train/Test Split
Test Size: 0.2
☑ Feature Scaling
Scaler Type: ● Standard ○ MinMax
Reproducibility
Seed Type: ● Fixed ○ Random
Simulation Screen
Datasets
☐ Iris Dataset
☐ Two Moons (Synthetic)
☐ Wine Dataset
☐ Breast Cancer Wisconsin
☐ Digits Dataset
☐ Haberman's Survival Data
☐ Pima Indians Diabetes
☐ Banknote Authentication
☐ Mushroom Dataset
☐ Synthetic Classification
☐ Abalone Age (Classification)
☐ Bike Sharing Hourly Demand
☐ MNIST Digits (Subset 7k)
Train
Evaluate
Reset Workspace

(Currently Empty - Simulation results will appear here)

Configured Model Instances: *No model instances configured yet.*

Selected Datasets: *No datasets selected.*

Global Settings:

```
{
  "useCrossValidation": false,
  "cvFolds": 5,
  "useTrainTestSplit": true,
  "testSplitRatio": 0.2,
  "randomSeedType": "fixed",
  "applyFeatureScaling": true,
  "scalerType": "standard"
}
```

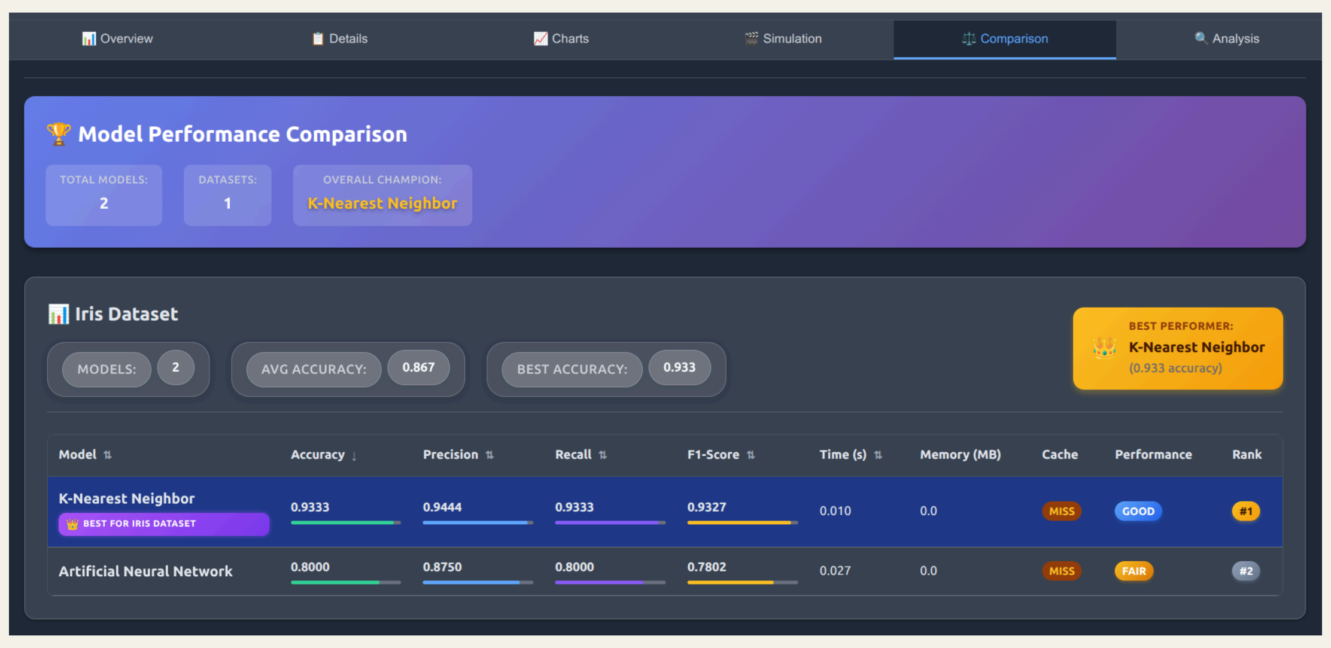
Interface home page: model and general configurations on the left and datasets on the right.

FINDINGS

The developed platform allows users to train different machine learning models simultaneously on a dataset and see their results in a single interface.

- **Comparative Performance Analysis:** Users can compare metrics such as accuracy, precision, recall and F1-score calculated for each model in tables and graphs. In this way, they can observe which model is more successful for a particular problem based on concrete data.
- **Automatic Code Generation:** The system successfully generates clean and executable Python code that includes data preprocessing and model training steps according to the choices made by the user in the interface (model, hyperparameters, etc.).
- **AI-supported Explanations:** Gemini API integration contributes to the reinforcement of theoretical knowledge by explaining what the generated code and the selected algorithm do in a language that a beginner can understand.

Interface



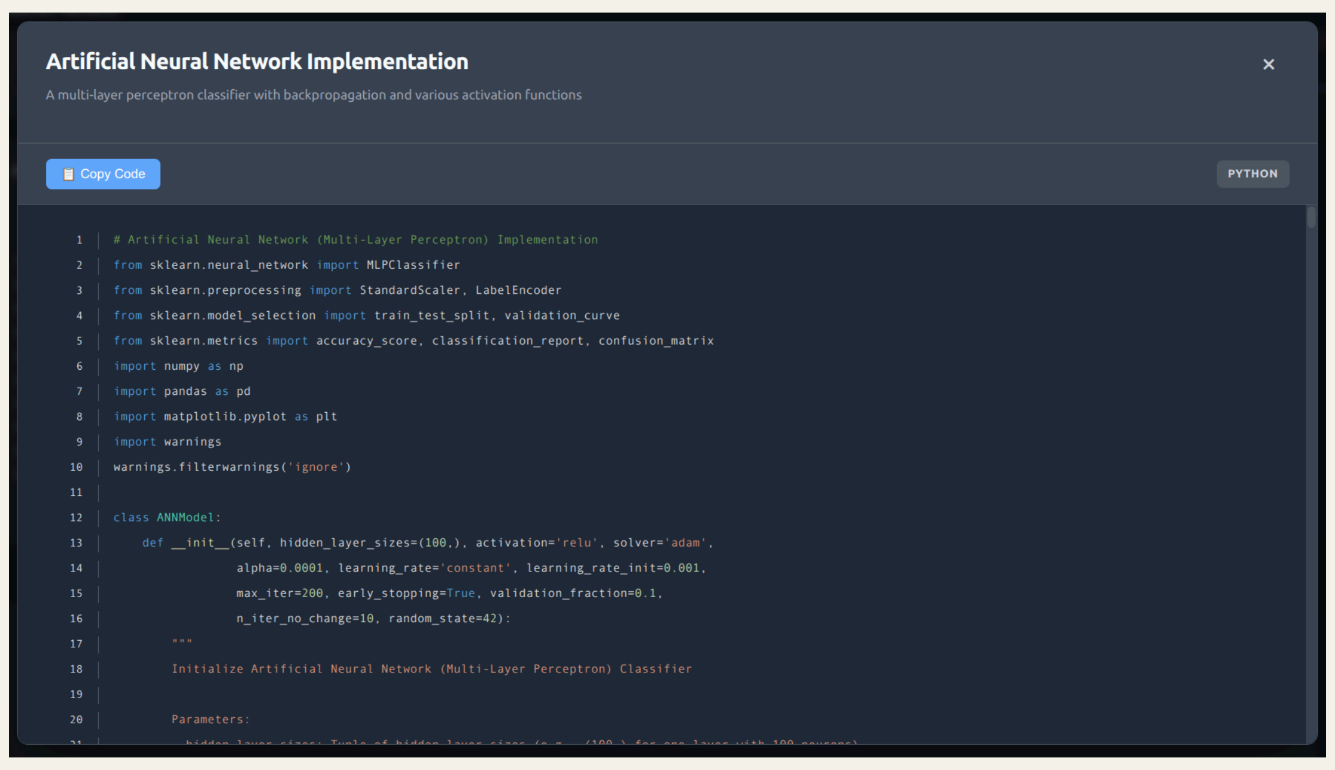
TOTAL MODELS: 2 | DATASETS: 1 | OVERALL CHAMPION: K-Nearest Neighbor

Iris Dataset

MODELS: 2 | AVG ACCURACY: 0.867 | BEST ACCURACY: 0.933

BEST PERFORMER: K-Nearest Neighbor (0.933 accuracy)

Model	Accuracy	Precision	Recall	F1-Score	Time (s)	Memory (MB)	Cache	Performance	Rank
K-Nearest Neighbor	0.9333	0.9444	0.9333	0.9327	0.010	0.0	MISS	GOOD	#1
Artificial Neural Network	0.8000	0.8750	0.8000	0.7802	0.027	0.0	MISS	FAIR	#2



Artificial Neural Network Implementation

A multi-layer perceptron classifier with backpropagation and various activation functions

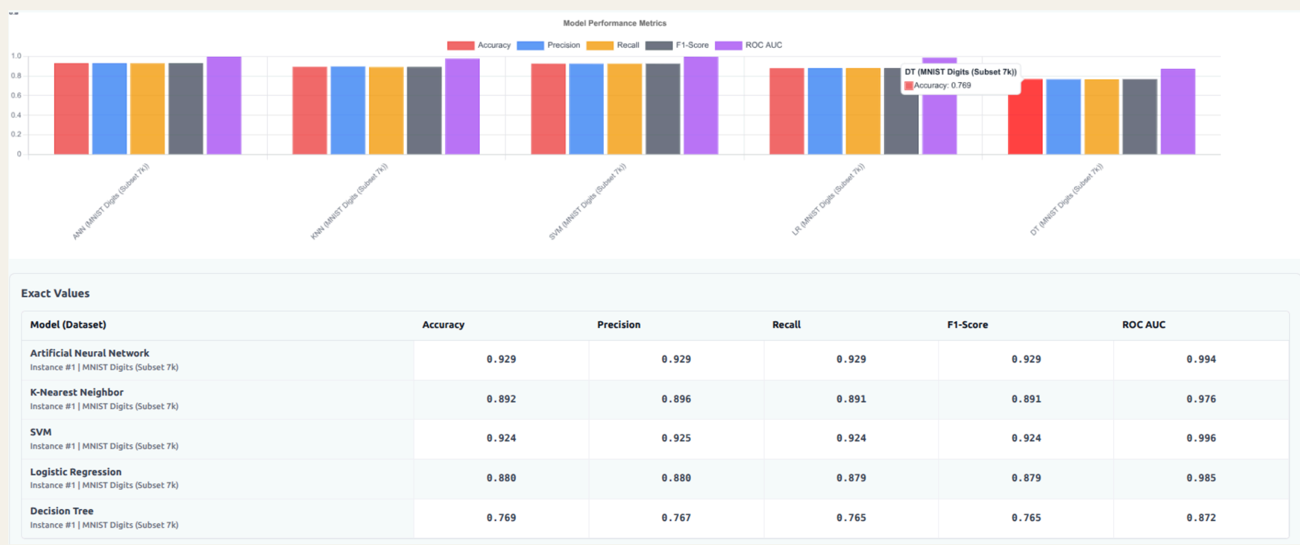
```python
# Artificial Neural Network (Multi-Layer Perceptron) Implementation
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.model_selection import train_test_split, validation_curve
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')

class ANNModel:
    def __init__(self, hidden_layer_sizes=(100,), activation='relu', solver='adam',
                 alpha=0.0001, learning_rate='constant', learning_rate_init=0.001,
                 max_iter=200, early_stopping=True, validation_fraction=0.1,
                 n_iter_no_change=10, random_state=42):
        """
        Initialize Artificial Neural Network (Multi-Layer Perceptron) Classifier

        Parameters:
```

Tables and Graphs

Available Datasets

Dataset Name	Number of Sample	Number of Features	Number of Class
Iris	150	4	3
Two Moons (Synthetic)	400	2	2
Wine	178	13	3
Breast Cancer Wisconsin	569	30	2
Digits	1797	64	10
Haberman’s Survival	306	3	2
Pima Indians Diabetes	768	8	2
Banknote Authentication	1372	4	2
Mushroom	8124	22	2
Synthetic Classification	1000	20	2
Abalone	4177	8	28
Bike Sharing Hourly	17379	16	Regression Task
MNIST (Subset)	7,000	784	10
MNIST (Full)	70,000	784	10

Sample Output of Accuracy Metrics

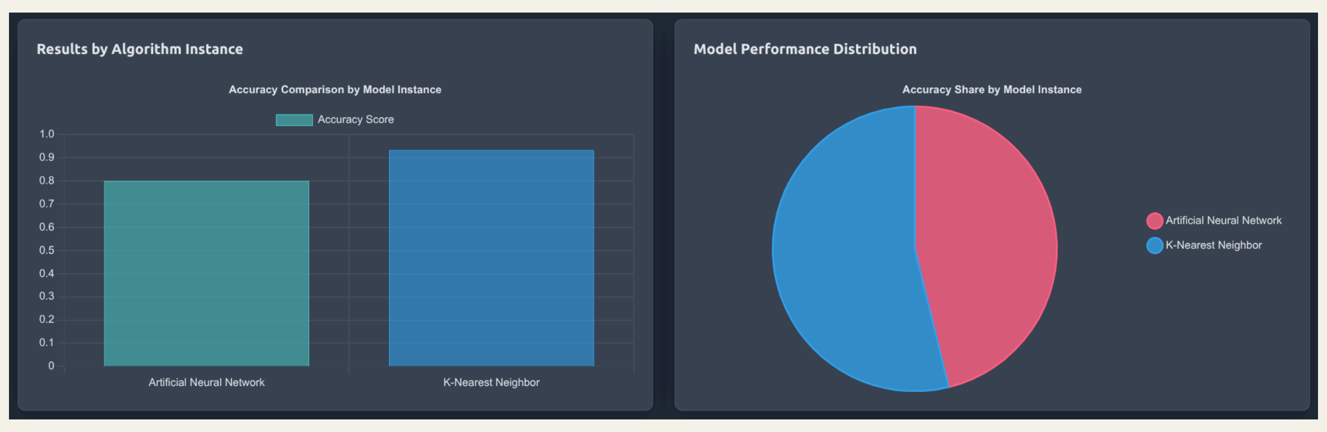


Exact Values

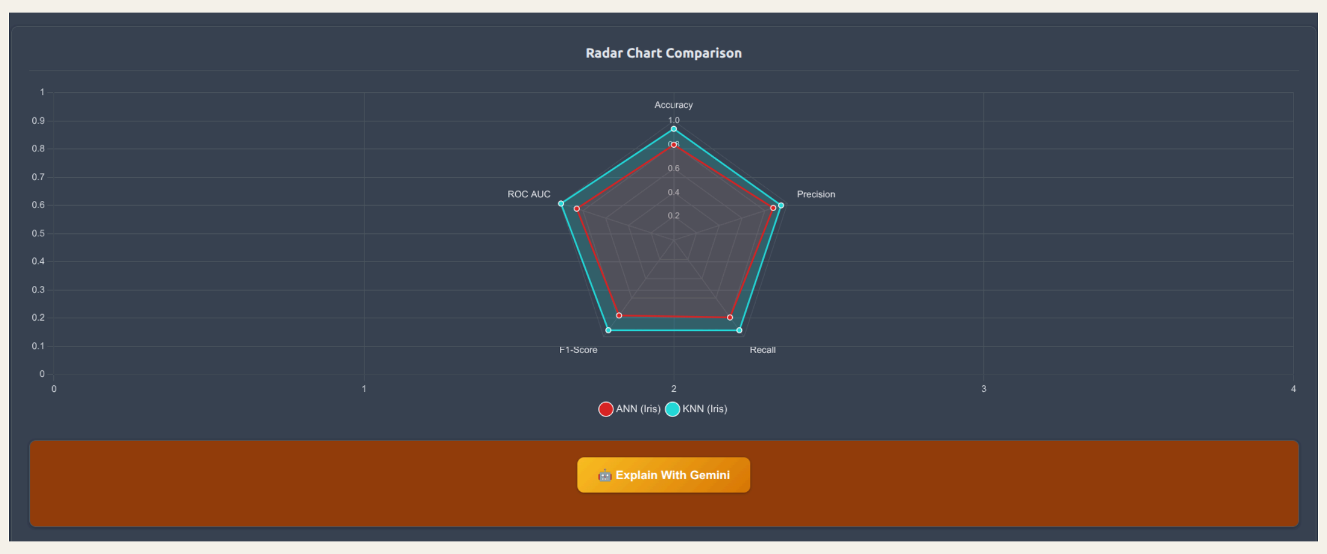
Model (Dataset)	Accuracy	Precision	Recall	F1-Score	ROC AUC
Artificial Neural Network Instance #1 | MNIST Digits (Subset 7k)	0.929	0.929	0.929	0.929	0.994
K-Nearest Neighbor Instance #1 | MNIST Digits (Subset 7k)	0.892	0.896	0.891	0.891	0.976
SVM Instance #1 | MNIST Digits (Subset 7k)	0.924	0.925	0.924	0.924	0.996
Logistic Regression Instance #1 | MNIST Digits (Subset 7k)	0.880	0.880	0.879	0.879	0.985
Decision Tree Instance #1 | MNIST Digits (Subset 7k)	0.769	0.767	0.765	0.765	0.872

Simulation Results

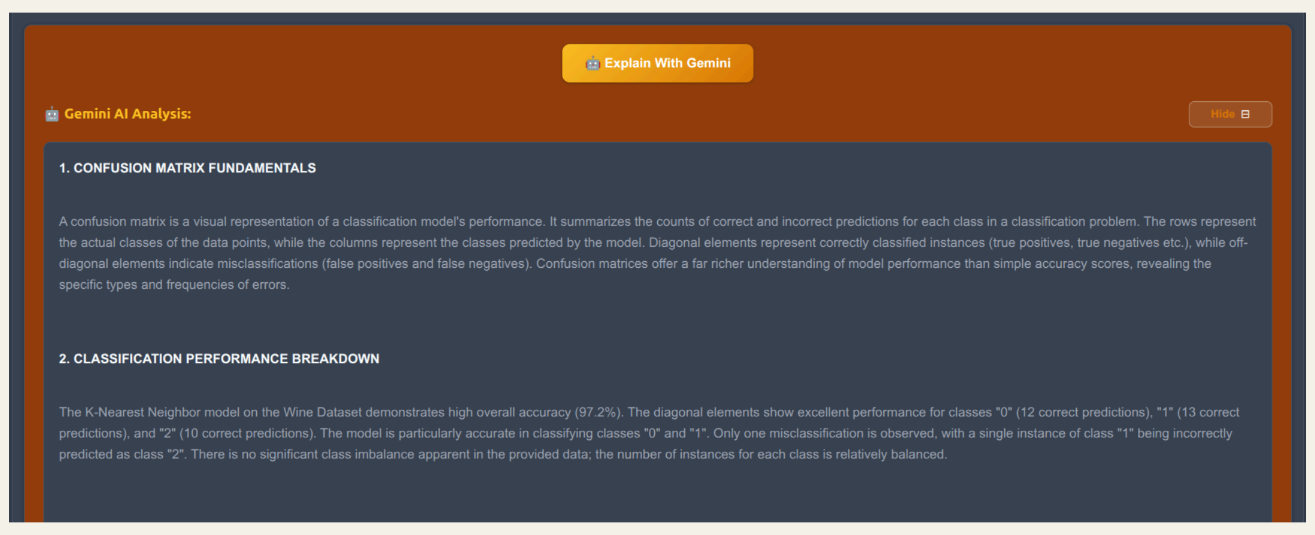
From Overview Tab:



From Charts Tab:



Asking Gemini for explain:



Gemini AI Analysis:

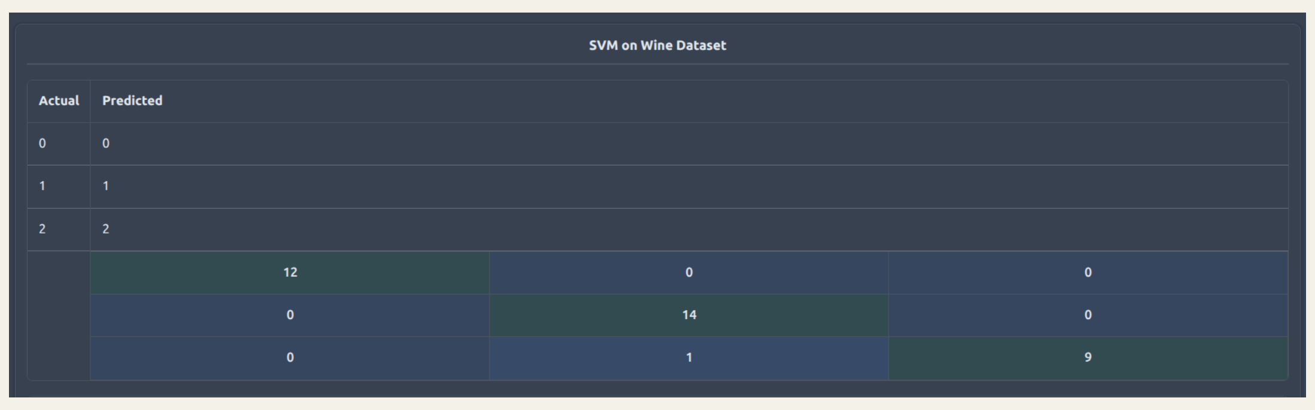
1. CONFUSION MATRIX FUNDAMENTALS

A confusion matrix is a visual representation of a classification model's performance. It summarizes the counts of correct and incorrect predictions for each class in a classification problem. The rows represent the actual classes of the data points, while the columns represent the classes predicted by the model. Diagonal elements represent correctly classified instances (true positives, true negatives etc.), while off-diagonal elements indicate misclassifications (false positives and false negatives). Confusion matrices offer a far richer understanding of model performance than simple accuracy scores, revealing the specific types and frequencies of errors.

2. CLASSIFICATION PERFORMANCE BREAKDOWN

The K-Nearest Neighbor model on the Wine Dataset demonstrates high overall accuracy (97.2%). The diagonal elements show excellent performance for classes "0" (12 correct predictions), "1" (13 correct predictions), and "2" (10 correct predictions). The model is particularly accurate in classifying classes "0" and "1". Only one misclassification is observed, with a single instance of class "1" being incorrectly predicted as class "2". There is no significant class imbalance apparent in the provided data; the number of instances for each class is relatively balanced.

Example of Confusion Matrix:



SVM on Wine Dataset

Actual	Predicted		
0	0		
1	1		
2	2		
	12	0	0
	0	14	0
	0	1	9

Communication Information

Ahmet Özdemir
Computer Engineering

Adres:

E-posta:
a.ozdemir2020@gtu.edu.tr