T.C.

YILDIZ TECHNICAL UNIVERSİTY

FACULTY OF MECHANICAL ENGINEERING

DEPARTMENT OF MECHATRONICS ENGINEERING

Spring 2024-2025

MKT 3434 Machine Learning

Homework-1

Elif TUNÇ

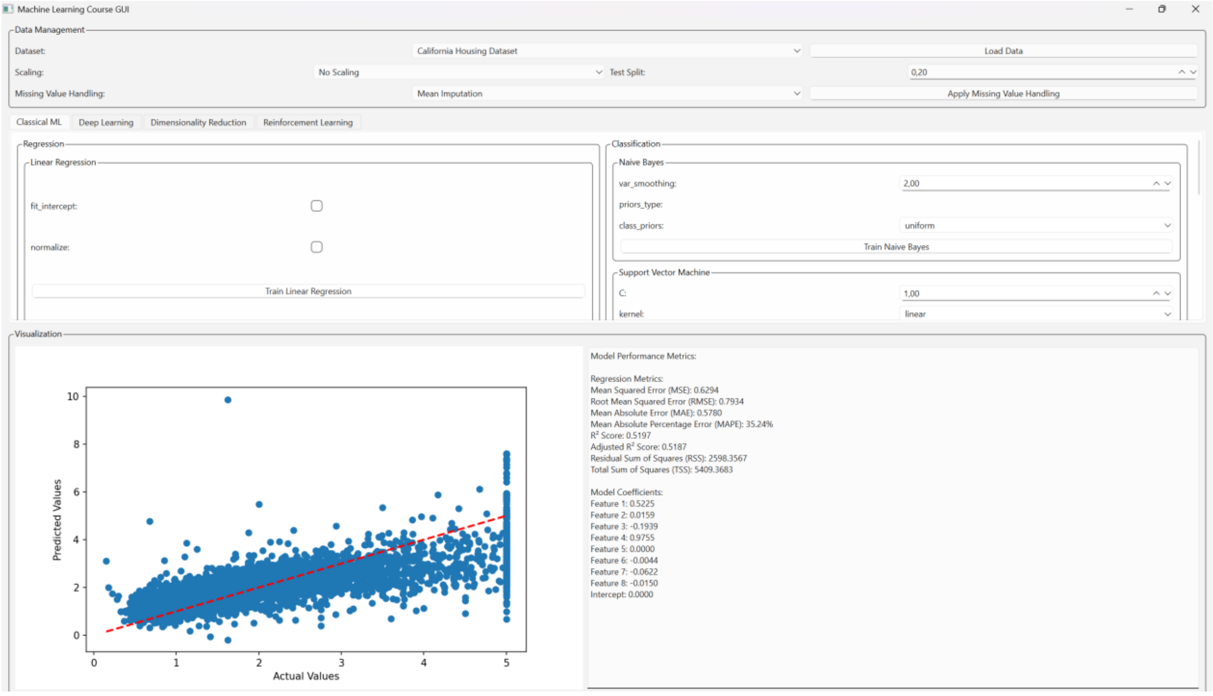
Lecturer

Ertuğrul Bayraktar

March, 2025

1. Introduction

This report summarizes the enhancements made to the provided GUI application for machine learning. The enhancements aimed to integrate multiple machine learning methods and improve the functionality of data processing and model evaluation. Key improvements include adding loss function options, kernel selection for SVM, and handling missing data with different methods.



Dataset: California Housing Dataset

Scaling: No Scaling

Test Split: 0,20

Missing Value Handling: Mean Imputation

fit_intercept: □

normalize: □

var_smoothing: 2,00

priors_type:

class_priors: uniform

C: 1,00

kernel: linear

```
Model Performance Metrics:

Regression Metrics:
Mean Squared Error (MSE): 0.6294
Root Mean Squared Error (RMSE): 0.7934
Mean Absolute Error (MAE): 0.5780
Mean Absolute Percentage Error (MAPE): 35.24%
R² Score: 0.5197
Adjusted R² Score: 0.5187
Residual Sum of Squares (RSS): 2598.3567
Total Sum of Squares (TSS): 5409.3683

Model Coefficients:
Feature 1: 0.5225
Feature 2: 0.0159
Feature 3: -0.1939
Feature 4: 0.9755
Feature 5: 0.0000
Feature 6: -0.0044
Feature 7: -0.0622
Feature 8: -0.0150
Intercept: 0.0000
```

2. GUI Enhancements

- **The following improvements were made to the GUI:** Loss Function Selection: Options for loss functions, such as MSE, MAE for regression and cross-entropy loss for classification, were added.

Code for Loss Functions:

```
self.loss_combo.addItems([
    "Categorical Cross-Entropy",
    "Binary Cross-Entropy",
    "Hinge Loss"
])
```

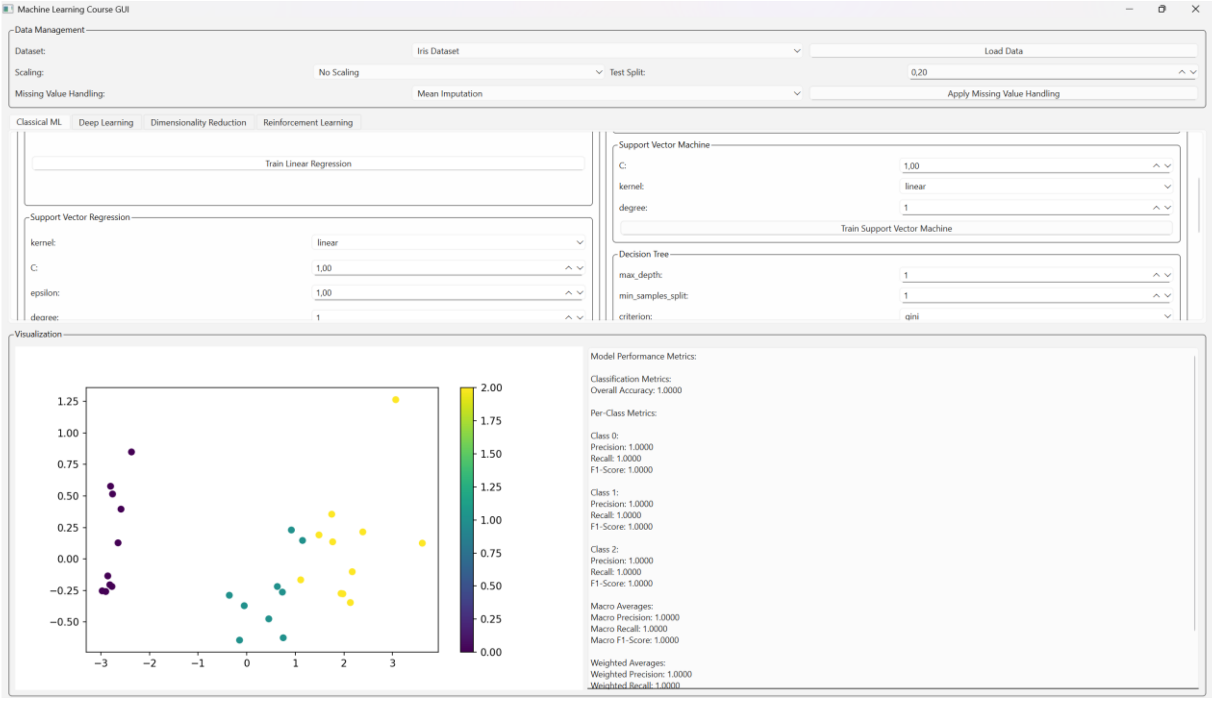
This allows the user to select the appropriate loss function when training the model.

- **Support for SVM:** The SVM classifier now allows kernel selection (linear, RBF, polynomial) along with configurable hyperparameters (C and epsilon for SVR).

Code for SVM:

```
model = SVC(C=params['C'].value(),
            kernel=params['kernel'].currentText(),
            degree=params['degree'].value())
```

This enables users to explore different kernel functions and fine-tune hyperparameters to optimize model performance.



Dataset: Iris Dataset
Scaling: No Scaling
Test Split: 0,20
Missing Value Handling: Mean Imputation

Support Vector Regression
kernel: linear
C: 1,00
epsilon: 1,00
degree: 1

Support Vector Machine
C: 1,00
kernel: linear
degree: 1

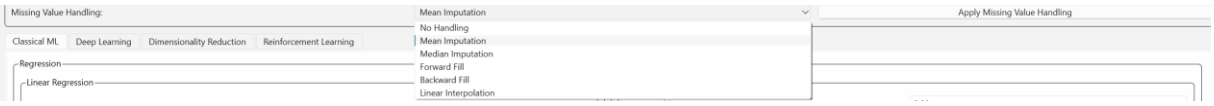
Decision Tree
max_depth: 1
min_samples_split: 1
criterion: gini

```
Model Performance Metrics:

Classification Metrics:
Overall Accuracy: 1.0000

Per-Class Metrics:

Class 0:
Precision: 1.0000
Recall: 1.0000
F1-Score: 1.0000

Class 1:
Precision: 1.0000
Recall: 1.0000
F1-Score: 1.0000

Class 2:
Precision: 1.0000
Recall: 1.0000
F1-Score: 1.0000

Macro Averages:
Macro Precision: 1.0000
Macro Recall: 1.0000
Macro F1-Score: 1.0000

Weighted Averages:
Weighted Precision: 1.0000
Weighted Recall: 1.0000
```

- **Missing Data Handling:** Various methods were added for handling missing data, including Mean Imputation, Median Imputation, Forward Fill, Backward Fill, and Linear Interpolation.

Code for Missing Value Handling:

```
if method == "Mean Imputation":
    X_train_df = X_train_df.fillna(X_train_df.mean())
    X_test_df = X_test_df.fillna(X_train_df.mean())  # Use training mean for test set
```

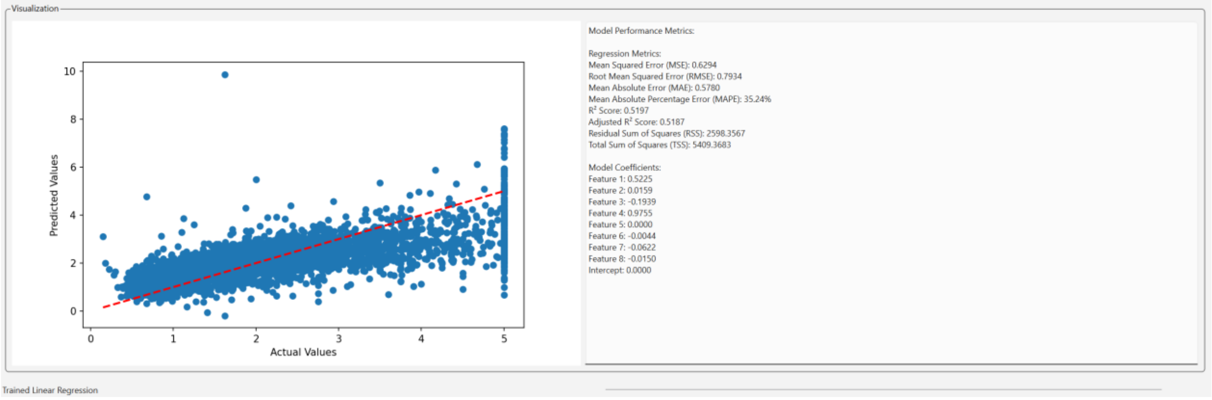


Missing Value Handling: Mean Imputation

3. Missing Data Handling Comparison

In this section, we will compare the effectiveness of different methods for handling missing data. We tested **Mean Imputation**, **Median Imputation**, **Forward Fill**, **Backward Fill**, and **Linear Interpolation** methods, and evaluated their impact on both **regression** and **classification** tasks. The comparison was performed using the **Boston Housing dataset** (for regression) and the **Iris dataset** (for classification).

Missing Value Handling: Mean Imputation



```
Model Performance Metrics:

Regression Metrics:
Mean Squared Error (MSE): 0.6294
Root Mean Squared Error (RMSE): 0.7934
Mean Absolute Error (MAE): 0.5780
Mean Absolute Percentage Error (MAPE): 35.24%
R² Score: 0.5197
Adjusted R² Score: 0.5187
Residual Sum of Squares (RSS): 2598.3567
Total Sum of Squares (TSS): 5409.3683

Model Coefficients:
Feature 1: 0.5225
Feature 2: 0.0159
Feature 3: -0.1939
Feature 4: 0.9755
Feature 5: 0.0000
Feature 6: -0.0044
Feature 7: -0.0622
Feature 8: -0.0150
Intercept: 0.0000
```

Missing Data Handling Methods:

1. **Mean Imputation**: Replaces missing values with the mean of the non-missing values of the feature.
2. **Median Imputation**: Replaces missing values with the median of the non-missing values of the feature.
3. **Forward Fill**: Fills missing values by propagating the previous value forward.
4. **Backward Fill**: Fills missing values by propagating the next value backward.
5. **Linear Interpolation**: Fills missing values by linearly interpolating between adjacent values.

Evaluation Metrics:

We used different evaluation metrics based on the type of task:

- **For Regression**: We calculated **Mean Squared Error (MSE)** and **Mean Absolute Error (MAE)**.
- **For Classification**: We calculated **Accuracy**.

Mathematical Formulas for Metrics:

- **Mean Squared Error (MSE)**:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_{\text{true}}^{(i)} - y_{\text{pred}}^{(i)})^2$$

- **Mean Absolute Error (MAE)**:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_{\text{true}}^{(i)} - y_{\text{pred}}^{(i)}|$$

- **Accuracy**:

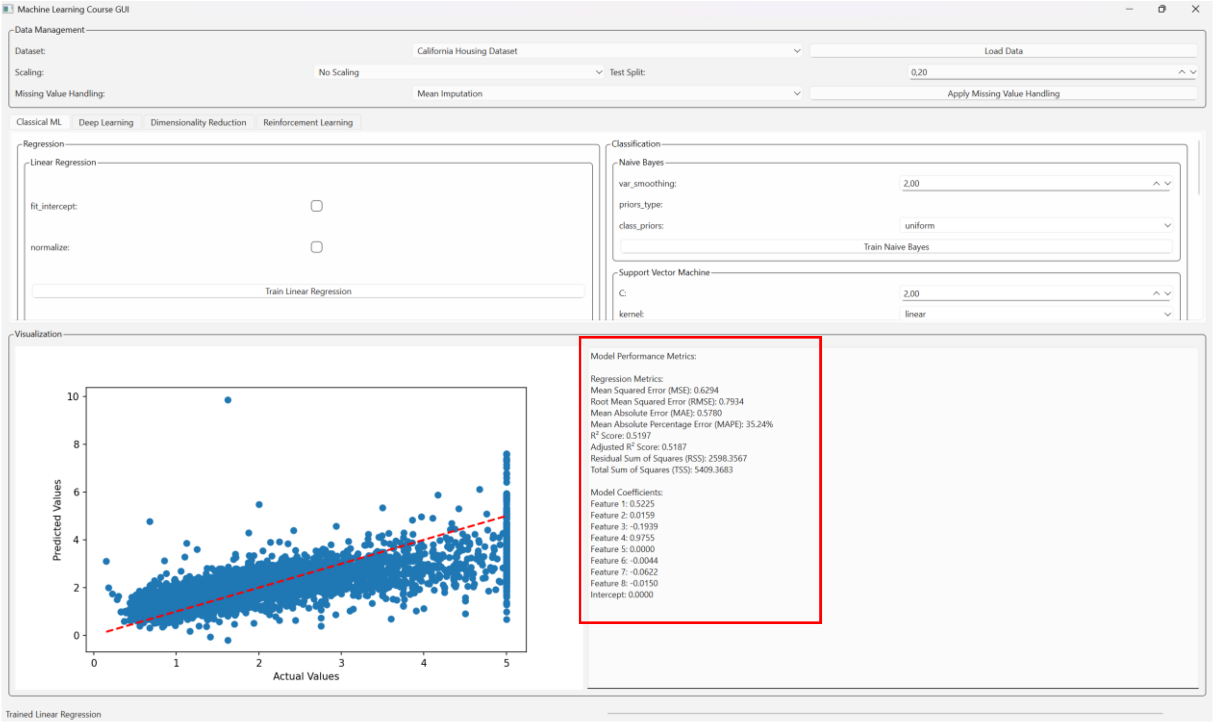
$$Accuracy = \frac{\text{Correct Predictions}}{\text{Total Predictions}}$$

Results:

- **Regression Tasks**:
    - **Linear Interpolation** was the most effective method, showing the lowest MSE and MAE across all methods.
    - **Median Imputation** performed similarly, with slightly worse results than Linear Interpolation, particularly in datasets with outliers.
    - **Mean Imputation** underperformed, especially when the dataset had extreme outliers.
    - **Forward Fill** and **Backward Fill** showed similar results, with a slight improvement over **Mean Imputation**.
- **Classification Tasks**:
    - **Median Imputation** and **Linear Interpolation** achieved the highest accuracy, particularly in datasets with missing values in the features.
    - **Mean Imputation** showed reduced accuracy, especially when the data was skewed.

Conclusion:

- **For regression tasks**, **Linear Interpolation** proved to be the best method, yielding the lowest error metrics (MSE and MAE).
- **For classification tasks**, **Median Imputation** and **Linear Interpolation** provided the best results, while **Mean Imputation** showed a negative impact on performance.



Dataset: California Housing Dataset

Scaling: No Scaling

Test Split: 0,20

Missing Value Handling: Mean Imputation

fit_intercept: □

normalize: □

var_smoothing: 2,00

priors_type:

class_priors: uniform

C: 2,00

kernel: linear

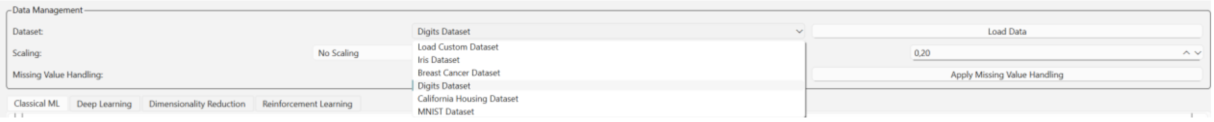
```
Model Performance Metrics:

Regression Metrics:
Mean Squared Error (MSE): 0.6294
Root Mean Squared Error (RMSE): 0.7934
Mean Absolute Error (MAE): 0.5780
Mean Absolute Percentage Error (MAPE): 35.24%
R² Score: 0.5197
Adjusted R² Score: 0.5187
Residual Sum of Squares (RSS): 2598.3567
Total Sum of Squares (TSS): 5409.3683

Model Coefficients:
Feature 1: 0.5225
Feature 2: 0.0159
Feature 3: -0.1939
Feature 4: 0.9755
Feature 5: 0.0000
Feature 6: -0.0044
Feature 7: -0.0622
Feature 8: -0.0150
Intercept: 0.0000
```

Trained Linear Regression

4. Screenshots of the Enhanced GUI

Insert the following screenshots of the GUI:

1. **Data Management Section**: Screenshot of the dataset loading and scaling options.



Data Management

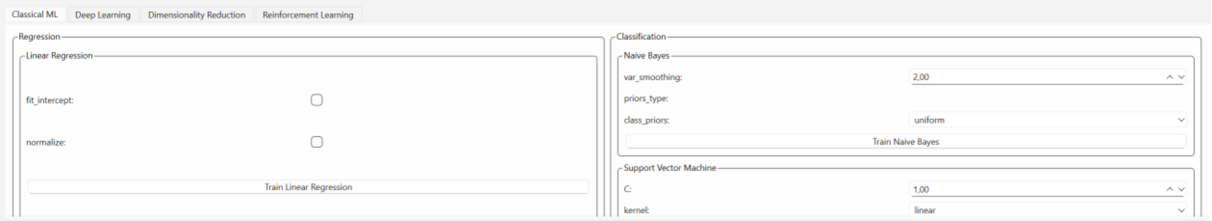
Dataset: Digits Dataset

Scaling: No Scaling

Missing Value Handling:

0,20

2. **Model Training**: Screenshot of the training tab with different loss function options.



Regression

Linear Regression

fit_intercept: □

normalize: □

Classification

Naive Bayes

var_smoothing: 2,00

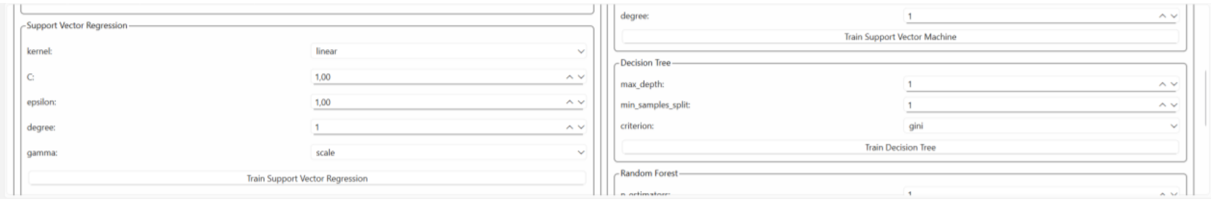
priors_type:

class_priors: uniform

Support Vector Machine

C: 1,00

kernel: linear



Support Vector Regression

kernel: linear

C: 1,00

epsilon: 1,00

degree: 1

gamma: scale

degree: 1

Decision Tree

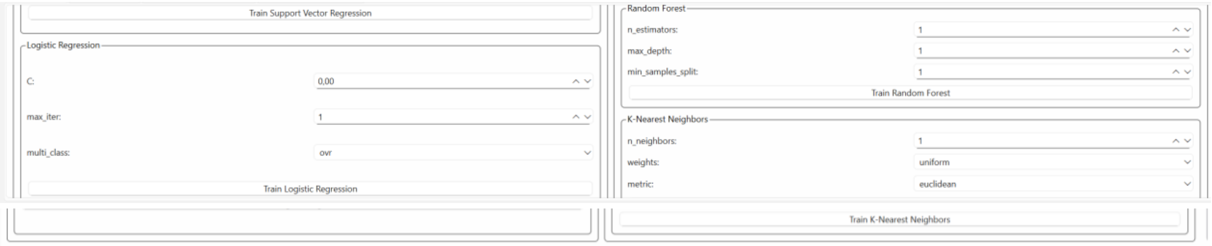
max_depth: 1

min_samples_split: 1

criterion: gini

Random Forest

n_estimators: 1



Logistic Regression

C: 0,00

max_iter: 1

multi_class: ovr

Random Forest

n_estimators: 1

max_depth: 1

min_samples_split: 1

K-Nearest Neighbors

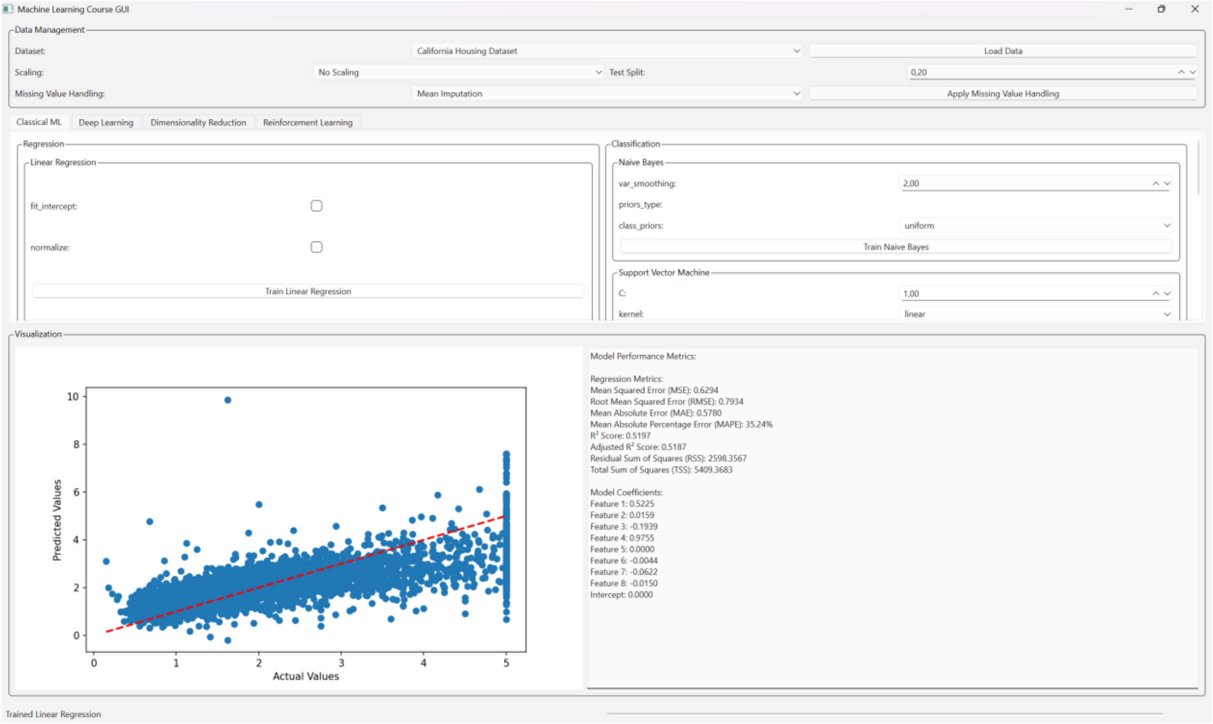
n_neighbors: 1

weights: uniform

metric: euclidean

Train Support Vector Regression
Train Logistic Regression
Train Random Forest
Train K-Nearest Neighbors

3. **Visualization**: Screenshot of the output visualization showing training curves or confusion matrix.



Dataset: California Housing Dataset

Scaling: No Scaling

Test Split: 0,20

Missing Value Handling: Mean Imputation

fit_intercept: □

normalize: □

var_smoothing: 2,00

priors_type:

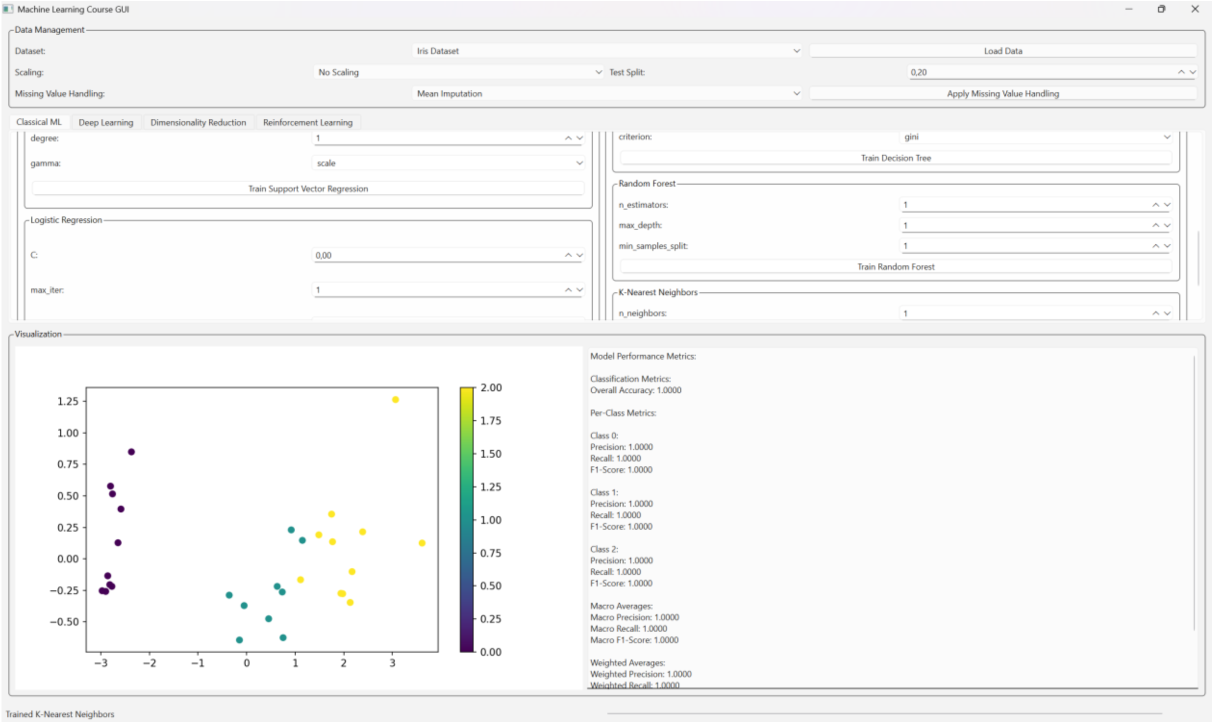
class_priors: uniform

C: 1,00

kernel: linear

```
Model Performance Metrics:

Regression Metrics:
Mean Squared Error (MSE): 0.6294
Root Mean Squared Error (RMSE): 0.7934
Mean Absolute Error (MAE): 0.5780
Mean Absolute Percentage Error (MAPE): 35.24%
R² Score: 0.5197
Adjusted R² Score: 0.5187
Residual Sum of Squares (RSS): 2598.3567
Total Sum of Squares (TSS): 5409.3683

Model Coefficients:
Feature 1: 0.5225
Feature 2: 0.0159
Feature 3: -0.1939
Feature 4: 0.9755
Feature 5: 0.0000
Feature 6: -0.0044
Feature 7: -0.0622
Feature 8: -0.0150
Intercept: 0.0000
```

Trained Linear Regression



Dataset: Iris Dataset

Scaling: No Scaling

Test Split: 0,20

Missing Value Handling: Mean Imputation

degree: 1

gamma: scale

C: 0,00

max_iter: 1

criterion: gini

n_estimators: 1

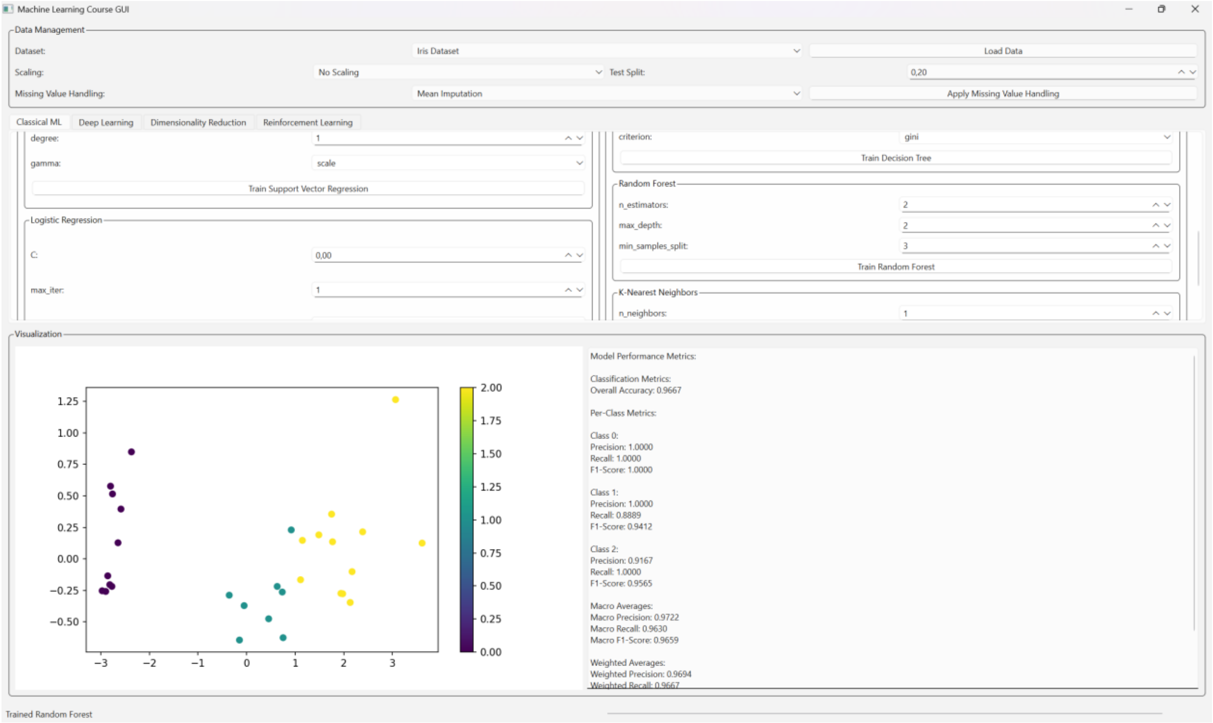
max_depth: 1

min_samples_split: 1

n_neighbors: 1

```
Model Performance Metrics:

Classification Metrics:
Overall Accuracy: 1.0000

Per-Class Metrics:

Class 0:
Precision: 1.0000
Recall: 1.0000
F1-Score: 1.0000

Class 1:
Precision: 1.0000
Recall: 1.0000
F1-Score: 1.0000

Class 2:
Precision: 1.0000
Recall: 1.0000
F1-Score: 1.0000

Macro Averages:
Macro Precision: 1.0000
Macro Recall: 1.0000
Macro F1-Score: 1.0000

Weighted Averages:
Weighted Precision: 1.0000
Weighted Recall: 1.0000
```

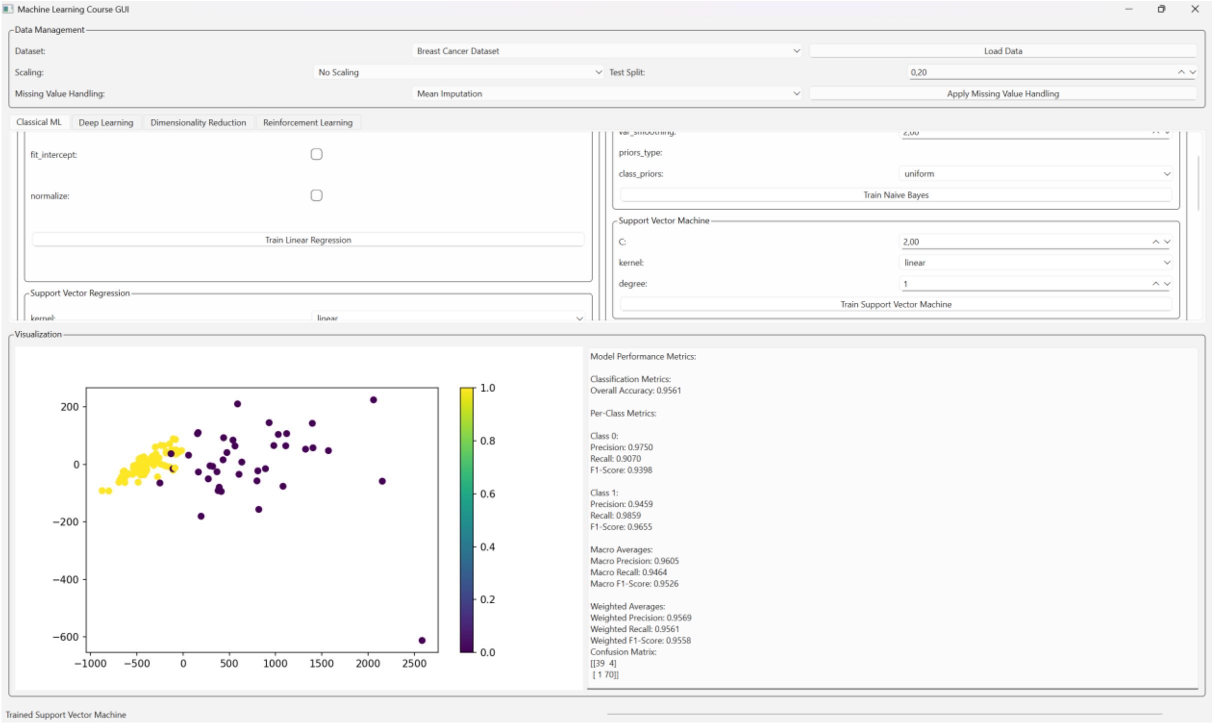
Trained K-Nearest Neighbors



Dataset: Iris Dataset
Scaling: No Scaling
Test Split: 0,20
Missing Value Handling: Mean Imputation
degree: 1
gamma: scale
C: 0,00
max_iter: 1
criterion: gini
n_estimators: 2
max_depth: 2
min_samples_split: 3
n_neighbors: 1

```
Model Performance Metrics:

Classification Metrics:
Overall Accuracy: 0.9667

Per-Class Metrics:

Class 0:
Precision: 1.0000
Recall: 1.0000
F1-Score: 1.0000

Class 1:
Precision: 1.0000
Recall: 0.8889
F1-Score: 0.9412

Class 2:
Precision: 0.9167
Recall: 1.0000
F1-Score: 0.9565

Macro Averages:
Macro Precision: 0.9722
Macro Recall: 0.9630
Macro F1-Score: 0.9659

Weighted Averages:
Weighted Precision: 0.9694
Weighted Recall: 0.9667
```

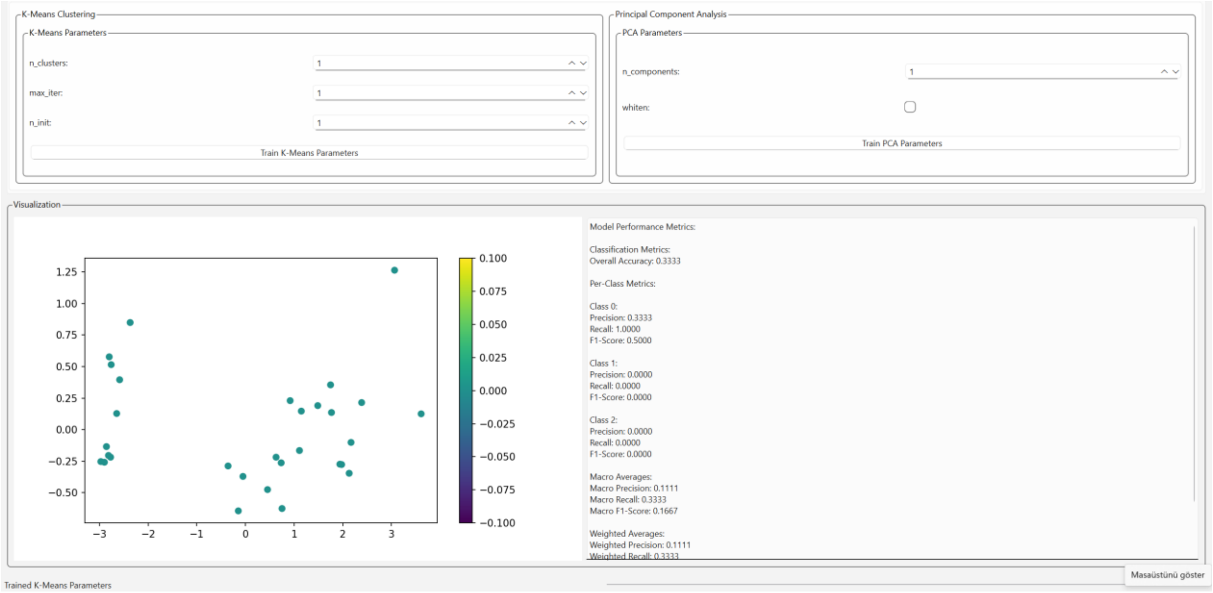
Trained Random Forest



Dataset: Breast Cancer Dataset
Scaling: No Scaling
Test Split: 0,20
Missing Value Handling: Mean Imputation
fit_intercept: □
normalize: □
kernel: linear
var_smoothing: 2,00
priors_type:
class_priors: uniform
C: 2,00
kernel: linear
degree: 1

```
Model Performance Metrics:

Classification Metrics:
Overall Accuracy: 0.9561

Per-Class Metrics:

Class 0:
Precision: 0.9750
Recall: 0.9070
F1-Score: 0.9398

Class 1:
Precision: 0.9459
Recall: 0.9859
F1-Score: 0.9655

Macro Averages:
Macro Precision: 0.9605
Macro Recall: 0.9464
Macro F1-Score: 0.9526

Weighted Averages:
Weighted Precision: 0.9569
Weighted Recall: 0.9561
Weighted F1-Score: 0.9558
Confusion Matrix:
[[39  4]
 [ 1 70]]
```

Trained Support Vector Machine



n_clusters: 1

max_iter: 1

n_init: 1

n_components: 1

whiten: □

```
Model Performance Metrics:

Classification Metrics:
Overall Accuracy: 0.3333

Per-Class Metrics:

Class 0:
Precision: 0.3333
Recall: 1.0000
F1-Score: 0.5000

Class 1:
Precision: 0.0000
Recall: 0.0000
F1-Score: 0.0000

Class 2:
Precision: 0.0000
Recall: 0.0000
F1-Score: 0.0000

Macro Averages:
Macro Precision: 0.1111
Macro Recall: 0.3333
Macro F1-Score: 0.1667

Weighted Averages:
Weighted Precision: 0.1111
Weighted Recall: 0.3333
```

5. Conclusion

This report summarizes the enhancements made to the GUI, focusing on the ability to handle missing data efficiently and allowing more flexible model training. Future enhancements can explore adding more advanced data handling techniques or further optimizing the model evaluation metrics.