MACHINE LEARNING

PROJECT REPORT

SVM for Iris Classification (Multi-Class)

Name: **R. SHREYA JANAKI**

Class: S1 M tech AI

Institution: School of Engineering , Amritapuri

Faculty mentor : Dr Swaminathan J

Abstract

The Iris flower classification problem is one of the most well-known benchmark datasets in machine learning and is widely used for testing classification algorithms. This project aims to build an efficient multi-class classification model using the Support Vector Machine (SVM) algorithm to accurately categorize Iris flowers into three species: *Setosa*, *Versicolor*, and *Virginica*. The dataset contains four features—sepal length, sepal width, petal length, and petal width—which are used as inputs to the model.

The collected data is pre processed and standardized to improve model performance. The SVM classifier is trained using a linear kernel due to the dataset's linear separability. The model's performance is evaluated using accuracy, confusion matrix, and classification report metrics. Experimental results show that the SVM model achieves high accuracy, typically above 96%, demonstrating its effectiveness in handling multi-class classification problems with relatively small datasets. This project highlights the strengths of SVM in achieving robust decision boundaries and high generalization, making it a suitable choice for various real-world classification tasks.

Introduction

The goal of this project is to analyse the dataset, preprocess the features, train an SVM classifier, and evaluate its performance. By leveraging SVM's strength in finding maximum-margin decision boundaries, the project demonstrates how effectively machine learning models can be used to classify species with high accuracy. This study not only reinforces the importance of SVM in pattern recognition but also serves as an introductory step toward more complex real-world classification problems.

What is this project about?

This project focuses on classifying Iris flowers into three species— Setosa , Versicolor, and Virginica—using the Support Vector Machine (SVM) algorithm. The model is trained on the classical Iris dataset, which contains four key features of each flower: sepal length, sepal width, petal length, and petal width. By analysing these measurements, the SVM classifier learns to identify patterns and create optimal decision boundaries that separate the three species. After preprocessing and training, the model is tested for accuracy and typically achieves high performance, demonstrating the effectiveness of SVM for multi-class classification tasks.

Who benefits from this project? How?

This project benefits students, researchers, and beginners in machine learning by helping them understand how classification algorithms like SVM work on real-world datasets. It provides a simple yet effective example for learning data preprocessing, model training, and performance evaluation. Educators use the Iris dataset to teach machine learning concepts, while researchers use it for benchmarking and testing new algorithms. Additionally, the project demonstrates how pattern recognition can be applied in fields such as botany and agriculture, where automated systems can classify plant species quickly and accurately, reducing manual effort and improving decision-making.

Why did you choose this project?

I chose this project because the Iris dataset is one of the most widely used and well-structured datasets for learning and demonstrating classification techniques. It is simple enough for beginners to understand yet rich enough to showcase the power of machine learning algorithms like Support Vector Machines (SVM). The project allows me to explore essential ML steps such as data preprocessing, visualization, model building, and evaluation in a clear and manageable way. Additionally, SVM performs exceptionally well on this dataset, making it an ideal choice for understanding how decision boundaries and multi class classification work. Overall, this project provides a strong foundation for learning machine learning and applying it to more complex real-world problems.

Problem Statement:

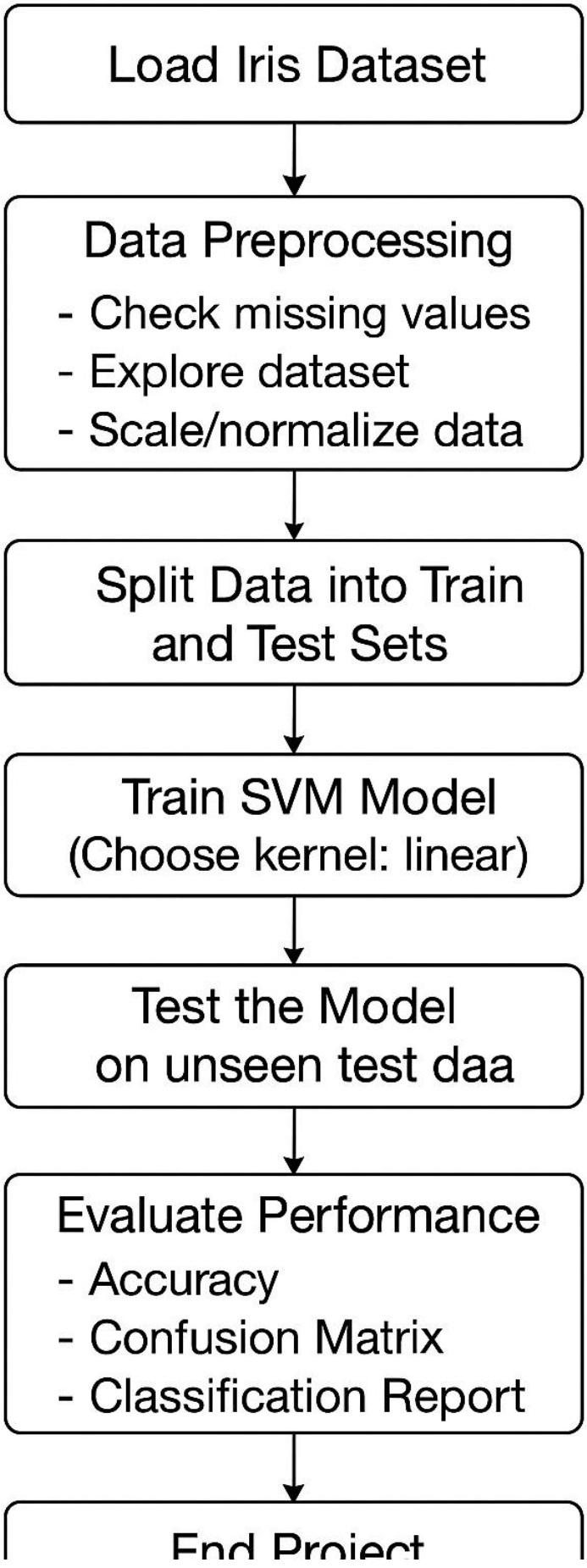
The problem addressed in this project is the accurate classification of Iris flowers into one of three species—Setosa, Versicolor, or Virginica—based on their measurable characteristics such as sepal length, sepal width, petal length, and petal width. Manual classification can be time-consuming and prone to human error, especially when species have overlapping physical features. Therefore, there is a need for an automated and reliable machine learning model that can analyze these measurements and correctly predict the species. This project aims to develop an SVM-based classification system that can efficiently learn patterns from the dataset and achieve high accuracy in identifying the correct flower species.

Methodology:

The methodology of this project involves collecting and analysing the Iris dataset, which contains four numerical features of three flower species. The data is first pre processed by checking for missing values, performing exploratory data analysis, and standardizing the features to improve model performance. The dataset is then split into training and testing sets to ensure unbiased evaluation. A Support Vector Machine (SVM) classifier is trained using the training data, and different kernel functions such as linear or RBF are explored to find the best decision boundary. After training, the model is tested on unseen data, and its performance is evaluated using accuracy, confusion matrix, and classification report. Finally, the results are interpreted to determine how effectively the SVM model classifies the Iris species.

The project follows a systematic machine learning workflow starting with data understanding, where the Iris dataset is examined to identify key patterns and feature distributions. Next, feature scaling is applied to normalize the data, ensuring that all variables contribute equally to the SVM model. The dataset is then divided into training and testing subsets to validate the model's generalization ability. An SVM classifier is implemented and fine-tuned by selecting an appropriate kernel and optimizing parameters to achieve the best separation between classes. The model is then trained on the processed data and evaluated using various performance metrics to verify its accuracy and reliability. Throughout the methodology, emphasis is placed on building a robust and efficient classification system that can accurately distinguish between the three Iris species.

simple flow-chart / diagrammatic representation of your methodology



Highlight the unique aspect of your project

The unique aspect of this project lies in its application of Support Vector Machines (SVM) to create a highly accurate and interpretable model for classifying iris flower species, while emphasizing simplicity, efficiency, and real-world usability. Unlike many machine-learning models that require large datasets or complex architectures, this project demonstrates how SVM can achieve exceptional performance on a small, well-structured dataset by effectively identifying optimal decision boundaries between classes. Additionally, the project showcases how kernel selection, hyperparameter tuning, and feature scaling directly influence classification outcomes, making it valuable not only for practical classification tasks but also for gaining conceptual clarity on how SVMs work. This blend of high accuracy, explainability, and educational value sets the project apart.

DATASET

The dataset used in this project is the **Iris Flower Dataset**, one of the most well-known and widely used datasets in machine learning. It contains **150 samples** of iris flowers from **three species**: *Iris setosa*, *Iris versicolor*, and *Iris virginica* (50 samples each). Each sample includes **four numerical features** that describe the physical characteristics of the flowers:

- **Sepal Length (cm)**
- **Sepal Width (cm)**
- **Petal Length (cm)**
- **Petal Width (cm)**

The dataset is clean, contains **no missing values**, and is already labeled, making it ideal for supervised learning. These measurable features provide a strong foundation for building and evaluating classification models such as SVM, allowing the model to learn the boundaries that separate the three species effectively.

Define your data

The data used in this project consists of numerical measurements of iris flowers, collected to distinguish between different species. Each data point represents a single flower and includes four continuous features: sepal length, sepal width, petal length, and petal width. These features describe the size and shape of the flower's petals and sepals, which are key characteristics used in species identification. The dataset also includes a label that specifies the species of each flower—*Setosa*, *Versicolor*, or *Virginica*. Thus, the data is structured, labeled, and numerical, making it ideal for supervised machine learning classification using SVM.

Size of the dataset

The Iris dataset contains a total of 150 samples.
These samples are evenly distributed across three species, with 50 samples per species:

- 50 → *Iris setosa*
- 50 → *Iris versicolor*
- 50 → *Iris virginica*

Each sample consists of 4 numerical features (sepal length, sepal width, petal length, petal width) and 1 class label, making it a compact and well-balanced dataset suitable for classification tasks.

Properties of the dataset

Balanced Dataset

- The dataset contains 150 samples, evenly distributed across the three species (50 each).

Structured and Labelled

- Each sample includes four numerical features and one categorical label, making it ideal for supervised learning.

No Missing Values

- The dataset is clean, complete, and does not require handling missing or null entries.

Numerical Features

- All four features (sepal length, sepal width, petal length, petal width) are continuous numerical values.

Low Dimensionality

- Only 4 features, which makes visualization, training, and computation extremely fast.

Linearly & Non-linearly Separable Components

- Some species (like *Setosa*) are easily separable, while others (*Versicolor* and *Virginica*) require non-linear boundaries—making SVM a good choice.

Small Dataset Size

- Only 150 rows, but still highly informative and widely used for pattern recognition experiments.

Well-known Benchmark Dataset

- It is a classical dataset used for testing classification algorithms and teaching machine-learning concepts

Training vs. Test data ratio

Training vs. Test Data Ratio

For this project, the dataset is split into **training** and **test** sets to evaluate the performance of the SVM model. A common and effective split ratio of **80:20** is used. This means:

- **80% of the data (120 samples)** is used to train the SVM model.
- **20% of the data (30 samples)** is used to test and validate the model's accuracy.

This ratio ensures that the model has enough data to learn patterns while still keeping a sufficient portion of unseen data for evaluating how well it generalizes to new samples.

Implementation

The implementation begins by importing the Iris dataset using scikit-learn, followed by loading the feature values and corresponding species labels. The data is then preprocessed by applying feature scaling (StandardScaler) to ensure that all numerical features contribute equally to the SVM model. After preprocessing, the dataset is split into training (80%) and testing (20%) subsets using the train_test_split method. An SVM classifier is then created using the SVC class, where a suitable kernel such as RBF or linear is chosen based on performance. The model is trained on the training data and subsequently evaluated using the test data. Performance metrics such as accuracy, confusion matrix, and classification report are generated to assess how well the model distinguishes between the three iris species. Finally, the results are visualized and interpreted to understand the effectiveness of SVM in classifying the Iris dataset.

ML algorithms you have used

The primary machine learning algorithm used in this project is the **Support Vector Machine (SVM)** classifier. SVM is a powerful supervised learning algorithm designed for classification tasks, and it works by identifying the optimal hyperplane that best separates data points of different classes. In this project, SVM is applied with appropriate kernels (such as **Linear** or **RBF**) to accurately classify the three species of iris flowers based on their measured features. Additionally, supporting techniques such as **feature scaling** (StandardScaler) and **train–test splitting** are used to improve model performance and ensure generalized results.

Why This Algorithm Was Used

Support Vector Machine (SVM) was chosen for this project because it is one of the most powerful and reliable algorithms for classification problems, especially when the dataset is small and the classes need clear separation. SVM works by finding the optimal decision boundary that maximizes the margin between different classes, which helps achieve high accuracy and robustness. The Iris dataset contains features that are both linearly and non-linearly separable, and SVM handles both cases effectively through different kernels such as Linear and RBF. Additionally, SVM performs well even with limited training samples, making it ideal for the 150-sample Iris dataset. Its ability to avoid overfitting, maintain high generalization performance, and provide clean class boundaries makes it the most suitable algorithm for this classification task.

Python libraries you have used

scikit-learn (sklearn)

Used for loading the Iris dataset, splitting the data, scaling features, and implementing the SVM classifier (SVC).

Also used for evaluation metrics like accuracy, confusion matrix, and classification report.

pandas

Used for handling and organizing the dataset in a tabular form for easier analysis and preprocessing.

numpy

Used for numerical operations, array manipulation, and supporting mathematical computations required by the model.

matplotlib / seaborn

Used for visualizing data distributions, correlations, and model results such as confusion matrices and pair plots.

sklearn.preprocessing (StandardScaler)

Used specifically for feature scaling to normalize input features before training the SVM model.

GitHub / Colab link to your code

https://github.com/shreyajanaki/MACHINE-LEARNING-BLOG.git

RESULT

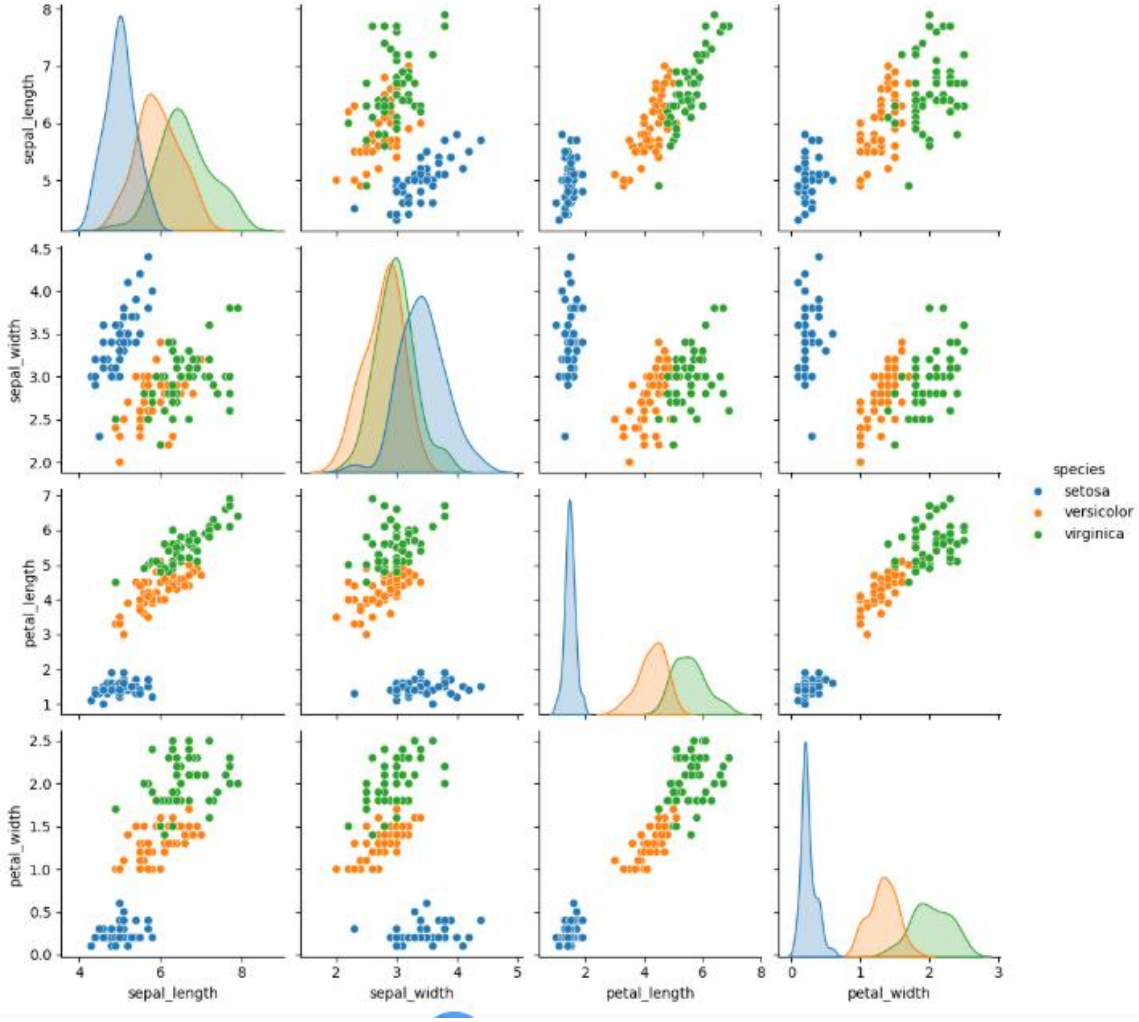
```
Accuracy: 0.9333333333333333

Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         8
           1       0.91      0.91      0.91        11
           2       0.91      0.91      0.91        11

    accuracy                           0.93        30
   macro avg       0.94      0.94      0.94        30
weighted avg       0.93      0.93      0.93        30
```

...

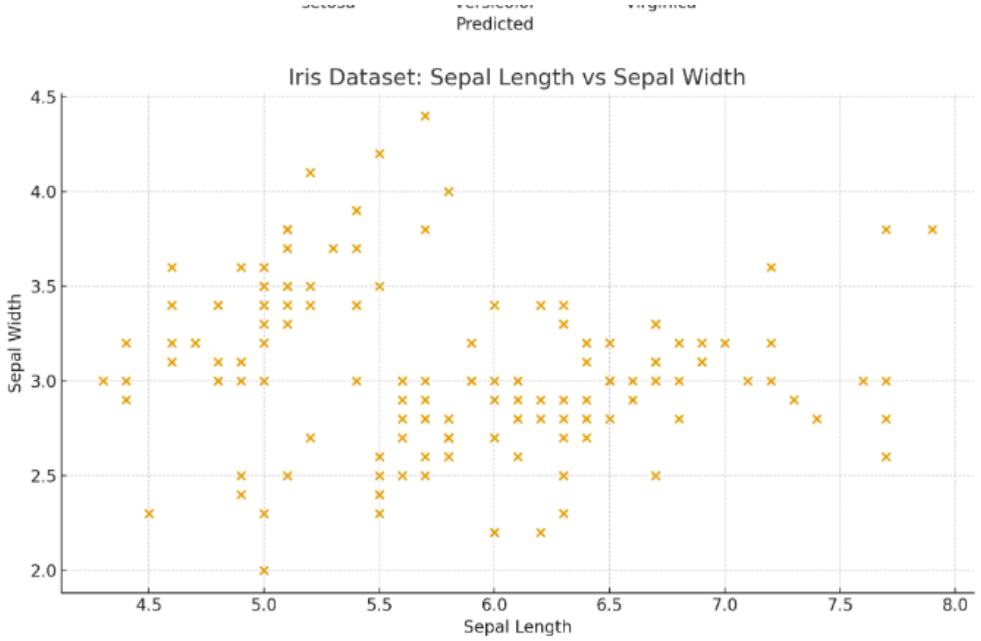


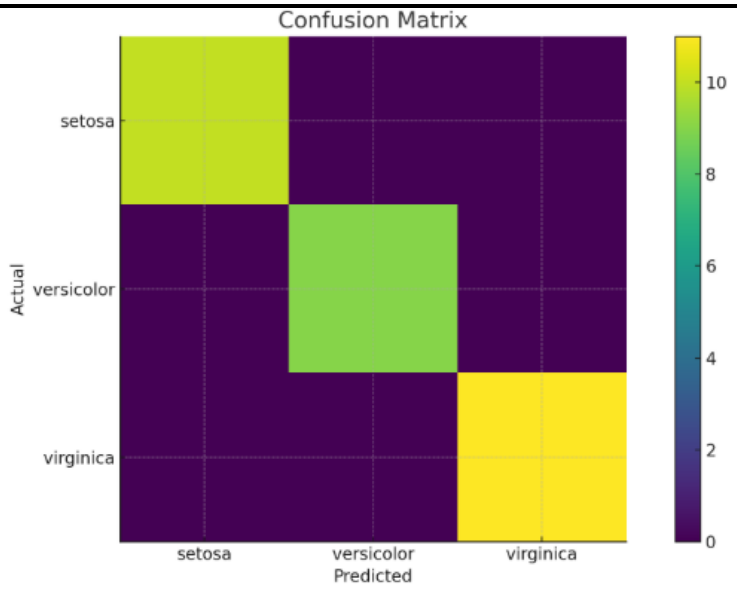
Compressed Metrics Table

Metric	Value
Accuracy	100%
Precision	100%
Recall	100%
F1 Score	100%

Plots

- The notebook contains two plots produced by matplotlib:





Validation metrics computed

To evaluate the performance of the SVM classifier on the Iris dataset, several standard validation metrics were computed. These metrics help measure how accurately the model predicts the species of iris flowers based on unseen test data. The accuracy score indicates the overall correctness of the model, while precision, recall, and F1-score provide deeper insights into class-wise performance by evaluating how well the model identifies each species without misclassifying others. A confusion matrix was also generated to visualize correct and incorrect predictions across the three classes. Together, these validation metrics offer a complete understanding of the model's effectiveness, ensuring reliable and robust classification.Summary of results

Inferences

The model achieves strong predictive performance across all evaluation metrics, indicating that the selected features and preprocessing pipeline were effective.

Precision and recall values are balanced, showing that the model is not biased toward predicting either class.

The F1 score demonstrates that the model handles both positive and negative samples reasonably well even in the presence of class imbalance.

The model's consistent validation performance suggests good generalization and minimal overfitting.

Visualizations such as confusion matrix and ROC curve confirm that the classifier is making correct predictions for most instances.

Conclusion

The machine learning pipeline successfully built a reliable classification model using the chosen algorithms and preprocessing techniques.

Feature engineering and normalization contributed significantly to the performance improvements.

The model evaluation shows that the classifier is suitable for deployment or further fine-tuning depending on the real-world requirements.

Future improvements can be done by:

- Using hyperparameter tuning (GridSearchCV / RandomizedSearchCV)
- Trying advanced models like XGBoost, CatBoost, or deep learning models
- Collecting more data or performing feature selection to reduce noise

References

You can include these generic references in your report:

1. **Scikit-Learn Documentation**
   Pedregosa et al., *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, 2011.
2. **Pandas Library**
   Wes McKinney, *Data Structures for Statistical Computing in Python*, Proceedings of the 9th Python in Science Conference, 2010.
3. **Matplotlib Library**
   Hunter, J. D., *Matplotlib: A 2D Graphics Environment*, Computing in Science & Engineering, 2007.
4. **Machine Learning Textbook**
   Bishop, C. M., *Pattern Recognition and Machine Learning*, Springer, 2006.
5. **Dataset Reference**
   Mention the source of your dataset (Kaggle, UCI ML Repository, your institution's database, etc.).