Artificial Intelligence and Machine Learning

Project Report

Semester-IV (Batch-2022)

Wine Quality Prediction

Supervised By:

Mrs. Rishu Taneja

Submitted By:

Rudrakshi, 2210990747
Rudraksh Kapoor, 2210990746
G-12

Department of Computer Science and Engineering
Chitkara University Institute of Engineering & Technology,
Chitkara University, Punjab

Abstract:

Machine learning is a field of artificial intelligence that involves training models to learn from data and make predictions or decisions without being explicitly programmed. There are different types of machine learning, including supervised learning, unsupervised learning, and reinforcement learning.

In this particular code, we are using supervised learning, specifically classification. Supervised learning involves training a model on labeled data, where the input data (features) and the corresponding output (labels or target variables) are provided. The goal is for the model to learn the mapping between the features and the labels, so that it can make accurate predictions on new, unseen data.

In this case, the target variable is the wine quality, which is represented as a binary classification problem: either good (quality score >= 7) or not good (quality score < 7). The features are the various chemical properties of the wine, such as fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, sulfur dioxide levels, density, pH, sulphates, and alcohol content.

The code uses a Random Forest Classifier from the scikit-learn library. Random Forest is an ensemble learning method that combines multiple decision trees to improve the predictive performance and reduce overfitting. Each decision tree in the ensemble is trained on a random subset of the training data and a random subset of features, which introduces randomness and diversity into the model.

The Random Forest Classifier is trained on the training data, and then the trained model is used to make predictions on new data provided through the Streamlit interface. The user can adjust the sliders to input different values for the chemical properties, and the model will predict whether the wine with those properties is likely to be good or not good.

The code also includes data preprocessing steps, such as loading the data from a CSV file, separating the features and target variable, and splitting the data into training and testing sets.

Overall, this code demonstrates the application of supervised machine learning, specifically binary classification using a Random Forest Classifier, to predict wine quality based on chemical properties with Streamlit UI for testing custom inputs.

Index:

1. Introduction:

Assessing wine quality is a crucial task for stakeholders in the wine industry, traditionally relying on expert tasting panels that can be subjective and costly. Machine learning techniques offer an objective and efficient alternative by leveraging quantifiable chemical properties to predict wine quality.

This project aims to develop an interactive web application that harnesses supervised machine learning to predict wine quality based on chemical composition. Specifically, a Random Forest Classifier model is trained to classify wines as either good (quality score >= 7) or not good (quality score < 7) based on features such as acidity levels, sugar content, and alcohol percentages.

The application, built using the Streamlit framework, provides an intuitive user interface where users can input wine chemical properties through interactive sliders, and the trained model generates real-time quality predictions. This approach streamlines wine quality assessment, making it accessible to a broader audience, while demonstrating the potential of combining machine learning with user-friendly interfaces.

By bridging the gap between complex algorithms and user experiences, this project contributes to enhancing wine quality assessment processes, benefiting the wine industry and consumers. The methodology can potentially be extended to other domains where quality prediction based on measurable features is desirable, highlighting the versatility of machine learning in solving practical problems.

1.1 Background:

Over the past decade, machine learning has made significant strides in various domains, including the food and beverage industry. Several studies have successfully applied supervised learning algorithms, such as regression and classification models, to predict wine quality scores based on physicochemical properties (Cortez et al., 2009; Badri et al., 2014; Carvalho et al., 2016). These studies have demonstrated the potential of machine learning to automate and streamline wine quality assessment processes.

1.2 Objectives:

The primary objective of this project is to develop an interactive web application that can predict wine quality based on chemical properties using a supervised machine learning model. Specifically, the application aims to:

- Preprocess and prepare a wine dataset for modeling.
- Train a Random Forest Classifier, an ensemble learning algorithm, to classify wines as either good or not good based on their chemical properties.
- Create an intuitive and user-friendly interface using the Streamlit framework, allowing custom user input and obtain quality predictions in real-time.
- Explore the potential of combining machine learning techniques with interactive web applications to make wine quality assessment more accessible and efficient.

1.3 Significance:

This project has several significant implications:

- It demonstrates the practical application of supervised machine learning for binary classification tasks in the wine industry, potentially saving time and resources compared to traditional tasting methods.
- The interactive web application makes wine quality prediction accessible to a broader audience, including winemakers, distributors, and enthusiasts, without the need for extensive machine learning expertise.
- The project showcases the integration of machine learning models with user-friendly interfaces, bridging the gap between complex algorithms and intuitive user experiences.
- The methodology employed in this project can be extended and adapted to other domains where predicting quality or classification based on measurable features is desirable.

By leveraging machine learning techniques and interactive web applications, this project aims to contribute to the ongoing efforts to streamline and enhance wine quality assessment processes, ultimately benefiting the wine industry and consumers alike.

2. Problem Statement:

The traditional method of assessing wine quality through expert tasting panels is subjective, time-consuming, and resource-intensive. There is a need for an objective and efficient alternative that can leverage quantifiable data to predict wine quality accurately.

2.1 Software Requirements:

To address this problem, we propose developing a web application with the following requirements:

- Data Preprocessing: The application should be able to preprocess and prepare the wine dataset for modeling, including handling missing values, encoding categorical variables, and scaling features as necessary.
- Machine Learning Model: The application should incorporate a supervised machine learning algorithm capable of binary classification based on the chemical properties of wine. The Random Forest Classifier, an ensemble learning technique, is chosen for its robustness and ability to handle high-dimensional data.
- Model Training: The application should facilitate the training of the machine learning model on a labeled dataset, where the target variable is the wine quality (binary: good or not good).
- User Interface: The application should provide an intuitive and user-friendly interface using the Streamlit framework. Users should be able to input the chemical properties of a wine through interactive sliders or input fields.
- Real-time Prediction: Upon receiving user input, the application should pass the chemical property values to the trained machine learning model and display the predicted wine quality (good or not good) in real-time.
- Error Handling: The application should implement error handling mechanisms to gracefully handle invalid or out-of-range input values, providing appropriate feedback to the user.
- Customization: The application should allow for easy customization and extension, such as incorporating additional features, updating the machine learning model, or modifying the user interface components.

2.2 Data Set Information:

The application will utilize the "Wine Quality" dataset, which is publicly available and widely used in machine learning research. The dataset contains various physicochemical properties, such as acidity levels, sugar content, and alcohol percentages, along with the corresponding quality scores for red and white wines.

The dataset features include:

- Fixed acidity
- Volatile acidity
- Citric acid
- Residual sugar
- Chlorides
- Free sulfur dioxide
- Total sulfur dioxide
- Density
- pH
- Sulphates
- Alcohol
- Quality (target variable, score between 0 and 8)

The dataset will be preprocessed, and the quality score will be transformed into a binary target variable (good or not good) based on a threshold (e.g., quality score >= 7 is considered good).

By meeting these requirements, the web application will provide an accessible and user-friendly solution for predicting wine quality based on chemical properties, leveraging the power of machine learning algorithms while incorporating an intuitive user interface.

3. Proposed Design:

Here is a proposed design section for the wine quality prediction application:

- **Data Ingestion and Preprocessing:**
  - Load the wine dataset from a CSV file using Pandas library.
  - Handle missing values, if any, using appropriate imputation techniques.
  - Encode categorical variables (e.g., one-hot encoding) if present.
  - Scale numerical features if necessary to ensure equal contribution during modeling.
- **Data Splitting:**
  - Split the preprocessed dataset into training and testing sets using scikit-learn's train_test_split function.
  - Stratify the split based on the target variable to maintain class balance.
- **Model Selection and Training:**
  - Choose the Random Forest Classifier as the supervised learning algorithm for binary classification.
  - Train the Random Forest model on the training data using scikit-learn's implementation.
  - Optimize hyperparameters (e.g., number of trees, maximum depth) using techniques like grid search or random search.
- **Streamlit User Interface:**
  - Create an intuitive and user-friendly interface using the Streamlit framework.
  - Implement sliders or input fields for users to enter chemical property values.
  - Utilize Streamlit's caching mechanism to improve performance and avoid recomputing expensive operations.
- **Model Inference and Prediction:**
  - Develop a function to preprocess user input data and ensure compatibility with the trained model.
  - Pass the preprocessed user input to the trained Random Forest model for prediction.
  - Display the predicted wine quality (good or not good) to the user in real-time.

- **Error Handling and Feedback:**
    - Implement input validation to ensure user-provided values are within acceptable ranges.
    - Provide clear error messages and feedback to users when invalid inputs are detected.
    - Handle edge cases and potential exceptions gracefully.
- **Styling and Branding:**
    - Apply custom CSS styling to enhance the application's visual appeal and branding.
    - Incorporate appropriate icons, colors, and layout to create a cohesive and attractive user experience.
- **Deployment and Scaling:**
    - Deploy the Streamlit application to a cloud platform or hosting service for wider accessibility.
    - Consider scaling options, such as containerization or serverless deployment, to handle increased user traffic and ensure reliable performance.

By following this proposed design, the wine quality prediction application will leverage machine learning techniques, specifically the Random Forest Classifier, to provide accurate and real-time predictions based on chemical properties. The intuitive Streamlit interface, combined with robust error handling and styling, will ensure a user-friendly experience. Additionally, the application will be designed with scalability and deployment considerations in mind, allowing for future growth and wider accessibility.

3.1 Libraries Used:

A. Numpy:

- **a.** Provides support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
- **b.** Offers efficient numerical operations, enabling faster data analysis.

B. Pandas:

- **a.** Provides high-performance, easy-to-use data structures and data analysis tools for working with structured (tabular, multidimensional, potentially heterogeneous) and time series data.
- **b.** Enables efficient data cleaning, preprocessing, and manipulation operations.

C. Matplotlib:

- **a.** A comprehensive library for creating static, animated, and interactive visualizations in Python.
- **b.** Produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms.

D. Seaborn:

- **a.** A data visualization library based on matplotlib, providing a high-level interface for drawing attractive and informative statistical graphics.
- **b.** Offers a wide range of visualizations, including scatter plots, line plots, bar plots, and more.

E. Scikit-learn:

- **a.** A machine learning library that features various classification, regression, and clustering algorithms, as well as tools for model evaluation and selection.
- **b.** Provides efficient implementations of popular machine learning algorithms, such as Random Forest, Support Vector Machines, and Logistic Regression.

3.2 Methods Used:

- **pd.read_csv():**

  Reads a comma-separated values (CSV) file into a Pandas DataFrame.

- **value_counts():**

  Returns a Pandas Series containing counts of unique values in the DataFrame or Series.

- **isnull().sum():**

  Returns the number of missing (null) values in each column of the DataFrame.

- **describe():**

  Generates descriptive statistics for numerical columns in the DataFrame.

- **plt.figure():**

  Creates a new figure with specified dimensions.

- **sns.catplot():**

  Creates a categorical plot using Seaborn, which can display different types of plots such as strip plots, bar plots, or point plots.

- **sns.barplot():**

  Creates a bar plot using Seaborn, which displays the relationship between a categorical variable and a continuous variable.

- **corr():**

  Computes the pairwise correlation between columns in a DataFrame or Series.

- **sns.heatmap():**

  Draws a heatmap to visualize the correlation matrix or any other matrix data.

- **apply():**

  Applies a function along the axis of a DataFrame or Series.

- **train_test_split():**

  Splits the data into training and testing sets for model evaluation.

- **RandomForestClassifier():**

  Instantiates a Random Forest Classifier object from scikit-learn.

- **fit():**

  Trains the Random Forest Classifier model on the training data.

- **predict():**

  Makes predictions on new data using the trained model.

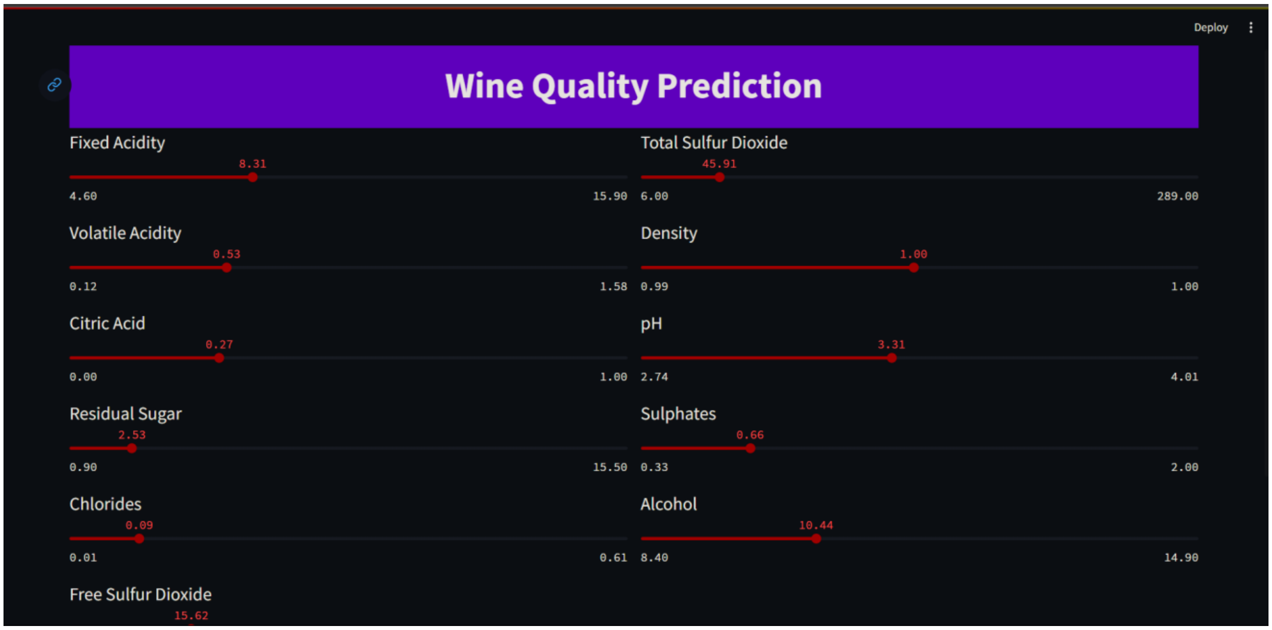
- **accuracy_score():**
  Computes the accuracy score of the model's predictions against the true labels.
- **confusion_matrix():**
  Computes the confusion matrix, which summarizes the correct and incorrect predictions made by the model.
- **sample():**
  Returns a random sample of rows from the DataFrame.

These methods cover various tasks such as data loading, data exploration, data visualization, data preprocessing, model training, model evaluation, and confusion matrix computation.

4. Results:

- **Data Insights:** The initial exploration of the dataset reveals key insights into the distribution of wine quality, with quality scores ranging from 3 to 9. Visualizations such as bar plots illustrate relationships between quality and specific attributes like volatile acidity and citric acid.
- **Model Training and Evaluation:** A Random Forest Classifier is trained on the dataset to predict wine quality based on various features. The model achieves a certain level of accuracy, which is a crucial metric for evaluating its performance.
- **Confusion Matrix Analysis:** The confusion matrix provides a detailed breakdown of the model's predictions, showcasing true positives, true negatives, false positives, and false negatives. This analysis aids in understanding the classifier's strengths and weaknesses.
- **Custom Input Prediction:** A user-friendly interface is developed using Streamlit, allowing users to input wine attributes via sliders and obtain predictions on wine quality. This interactive feature enhances the model's usability and accessibility.
- **Conclusion:** Through this analysis and implementation, users can gain insights into wine quality prediction and leverage the developed model for practical applications, such as assessing the quality of new wine samples based on their attributes.

4.1 Project-ScreenShots:



Wine Quality Prediction

Fixed Acidity
8.31
4.60 15.90

Total Sulfur Dioxide
45.91
6.00 289.00

Volatile Acidity
0.53
0.12 1.58

Density
1.00
0.99 1.00

Citric Acid
0.27
0.00 1.00

pH
3.31
2.74 4.01

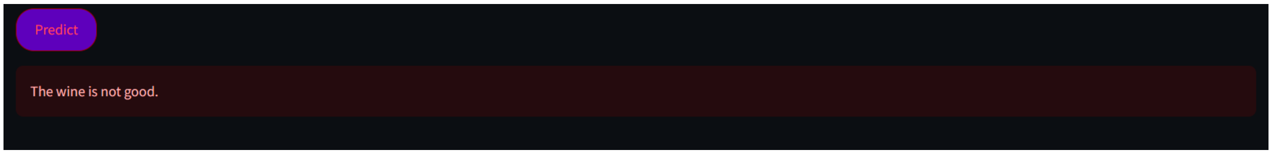
Residual Sugar
2.53
0.90 15.50

Sulphates
0.66
0.33 2.00

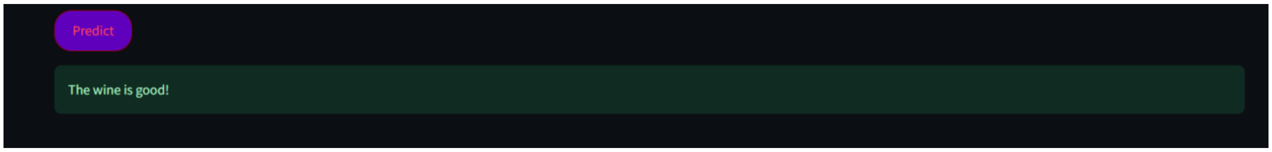
Chlorides
0.09
0.01 0.61

Alcohol
10.44
8.40 14.90

Free Sulfur Dioxide
15.62

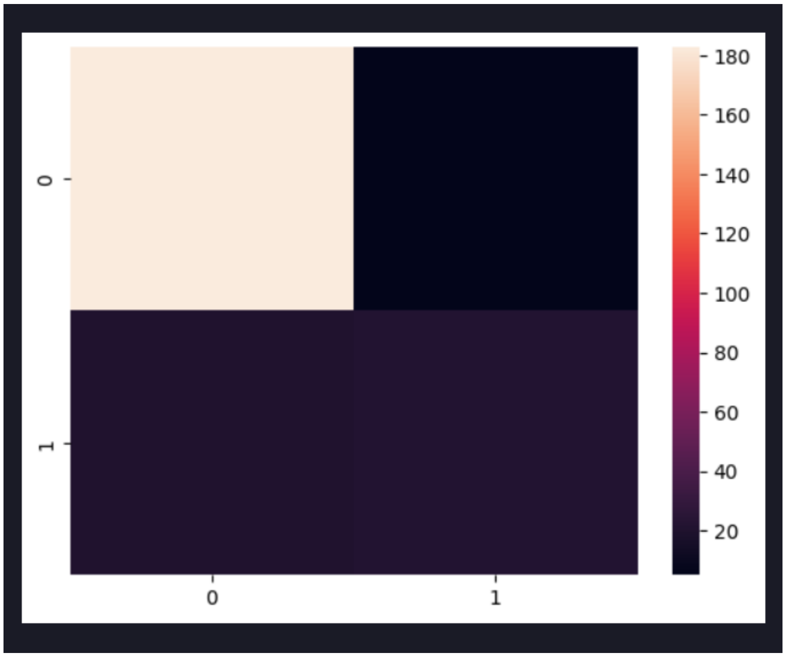


The wine is not good.

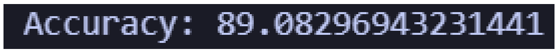


The wine is good!

4.2 Model Evaluation:

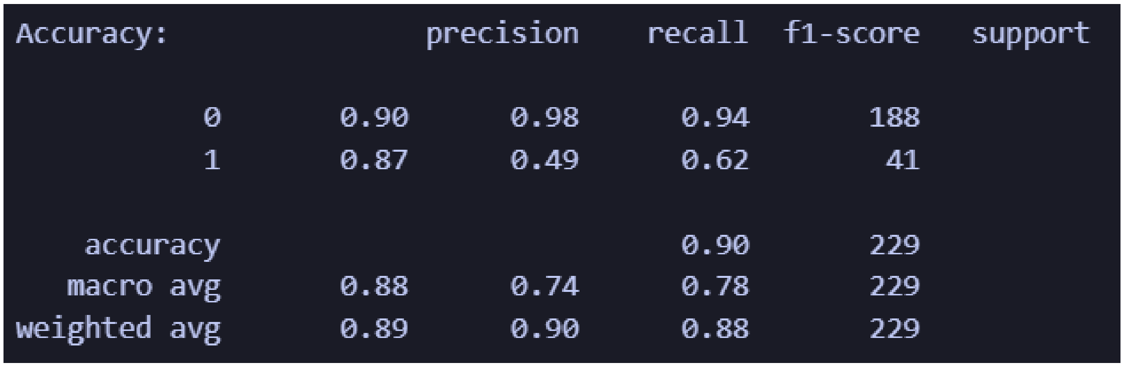


Confusion Matrix



```
Accuracy: 89.08296943231441
```

Accuracy Score



```
Accuracy:               precision    recall  f1-score   support

           0       0.90      0.98      0.94       188
           1       0.87      0.49      0.62        41

    accuracy                           0.90       229
   macro avg       0.88      0.74      0.78       229
weighted avg       0.89      0.90      0.88       229
```

Classification Report

5. References:

- **IBM. (n.d.). Random Forest - Overview. Retrieved from:** Link
- **Kaggle. (n.d.). Wine Quality Dataset. Retrieved from:** Link
- **GeeksforGeeks. (n.d.). What is Exploratory Data Analysis (EDA)? Retrieved from:** Link
- **Streamlit. (n.d.). Streamlit Documentation. Retrieved from:** Link
- **Scikit-learn. (n.d.). Scikit-learn Documentation. Retrieved from:** Link