Data Mining
Course Project

Kush Goel
23BCE2117

Iris Flower Classification Web App with Machine Learning

GitHub Repository Link (Source Code):
https://github.com/kushforges/iris-flower-classifier.git

Live App Link (Demo):
https://iris-flower-classifier-y35s7iemyzpcuanwyzpci2.streamlit.app/

Introduction

The Iris Flower Classification web application is designed to classify flowers into one of three species based on their sepal and petal measurements. This app utilizes machine learning algorithms to predict the species of an iris flower based on the user's input of flower dimensions. The app allows users to interactively explore different machine learning models, visualize the dataset, and view model performance metrics. This solution makes use of the popular **Iris dataset**, which is widely used for machine learning classification tasks.

The app is built using **Streamlit**, a Python framework that enables rapid development of interactive applications for data science and machine learning. In this project, users can:

- Select a machine learning model.
- Input flower measurements to receive a species prediction.
- Visualize and explore the Iris dataset through various plots.

Machine Learning Models

Three machine learning models are implemented in this app for comparison and experimentation:

1. **Random Forest Classifier**: A robust, ensemble learning method that uses multiple decision trees to make predictions.
2. **Logistic Regression**: A simple and effective linear model for binary and multi-class classification tasks.

3. **K-Nearest Neighbors (KNN)**: A non-parametric method used for classification by comparing the distances between data points.

Users can select the model they want to use via the sidebar, and the app will train the model on the Iris dataset, display performance metrics, and make predictions based on user input.

How the App Works

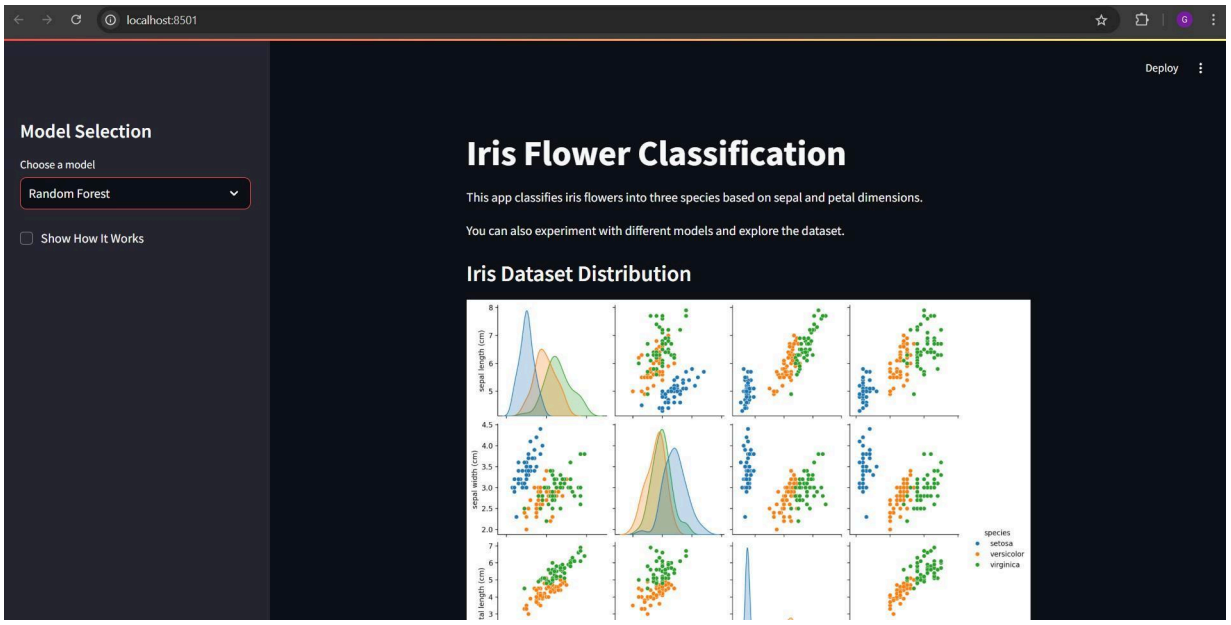
Upon running the app, users can:

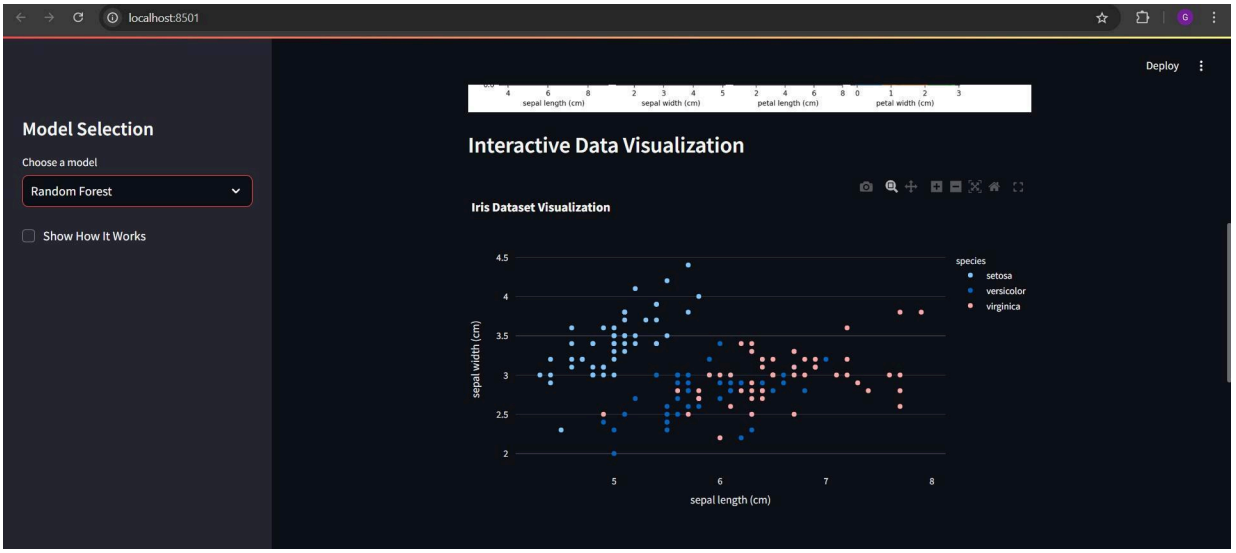
1. **Choose a Model**: The sidebar allows users to select one of the available models.
2. **Enter Flower Measurements**: Users can input the sepal and petal length and width for a flower.
3. **View the Prediction**: After entering the flower's measurements and clicking the “Classify Flower” button, the app predicts the species of the flower and displays the result.

The app also shows the **model accuracy** and a **classification report** (precision, recall, F1-score), giving users insights into how well the chosen model performs. Additionally, for the **Random Forest model**, the app displays a **feature importance** chart, showing the contribution of each feature to the model’s decision-making process.

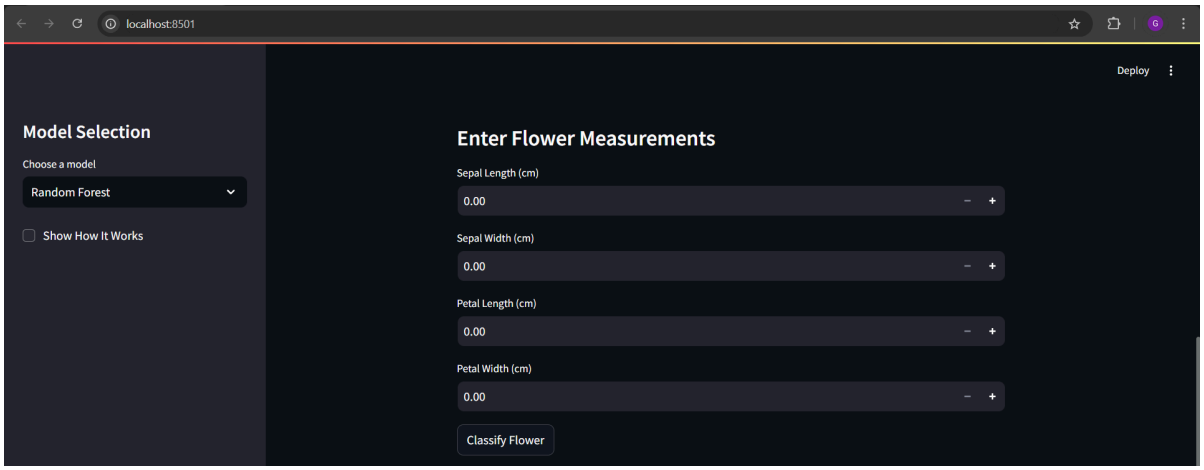
Interactive Features

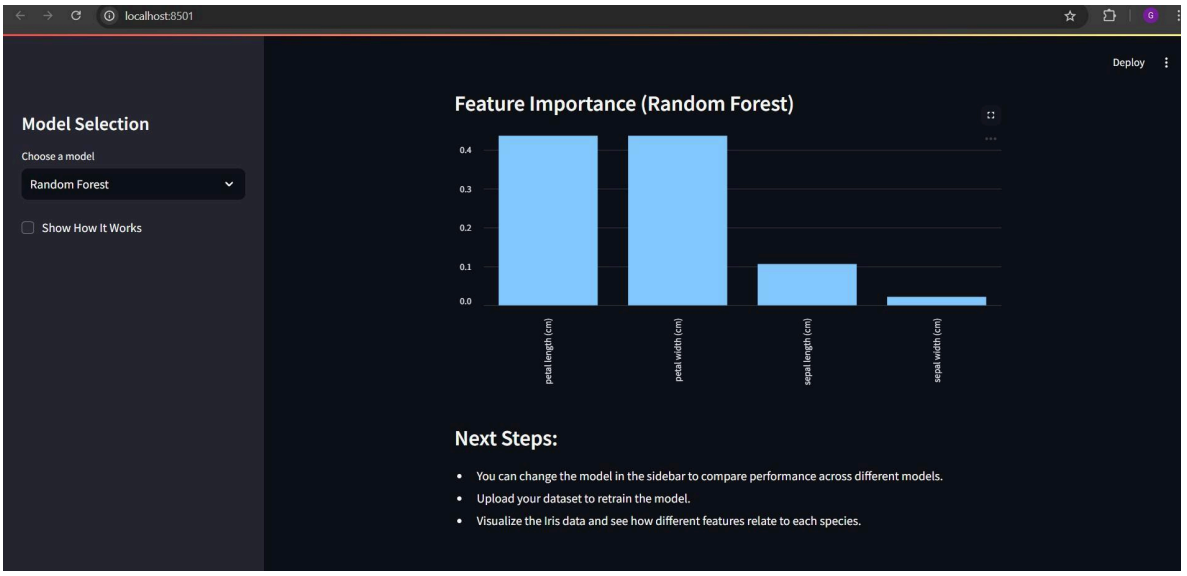
1. **Data Visualization**:
    - A **pairplot** of the Iris dataset is displayed to show the relationship between features and how the species are distributed.
    - An **interactive scatter plot** is implemented using **Plotly** to allow users to explore the data in an interactive manner. Users can zoom in and hover over the points to examine specific flower measurements.
2. **Performance Metrics**: For each model selected, the app provides detailed performance metrics including:
    - **Accuracy Score**: How accurate the model is on the entire dataset.
    - **Classification Report**: Precision, recall, and F1-score for each class.
3. **Feature Importance (for Random Forest)**: The **Random Forest** model provides an importance score for each feature, which helps users understand which measurements (sepal length, petal width, etc.) are most useful in predicting the species.

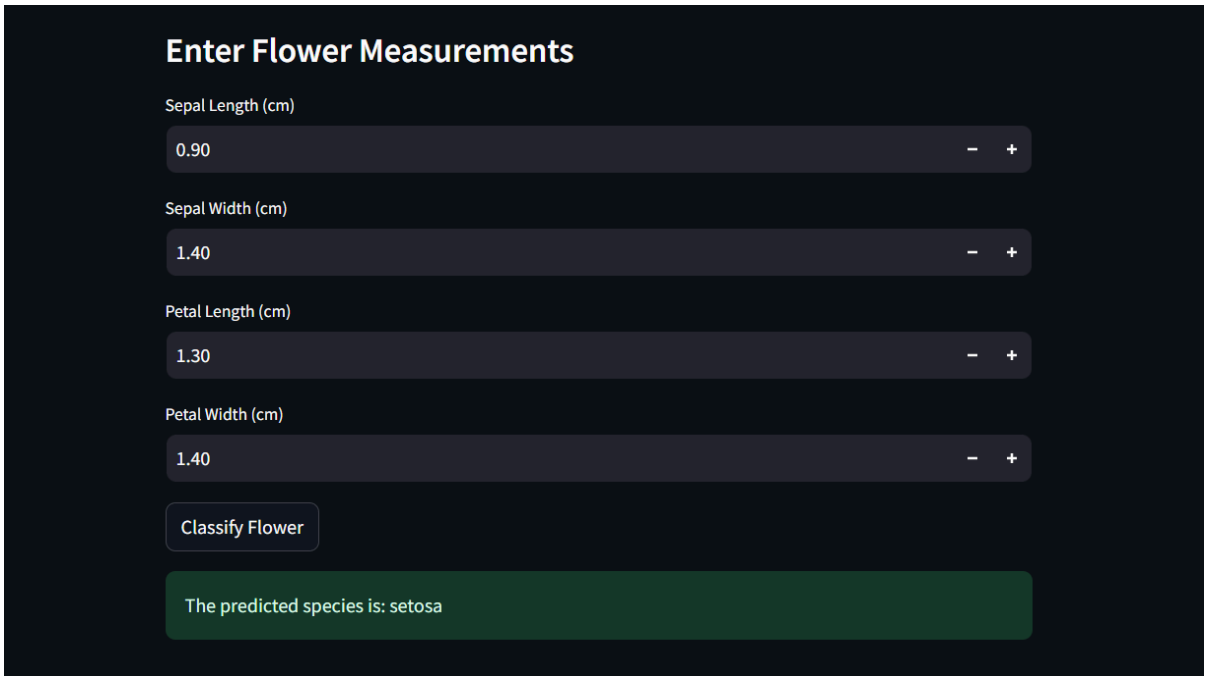

localhost:8501
Deploy
Model Selection
Choose a model
Random Forest
Show How It Works
Iris Flower Classification
This app classifies iris flowers into three species based on sepal and petal dimensions.
You can also experiment with different models and explore the dataset.
Iris Dataset Distribution
sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)
species
setosa
versicolor
virginica

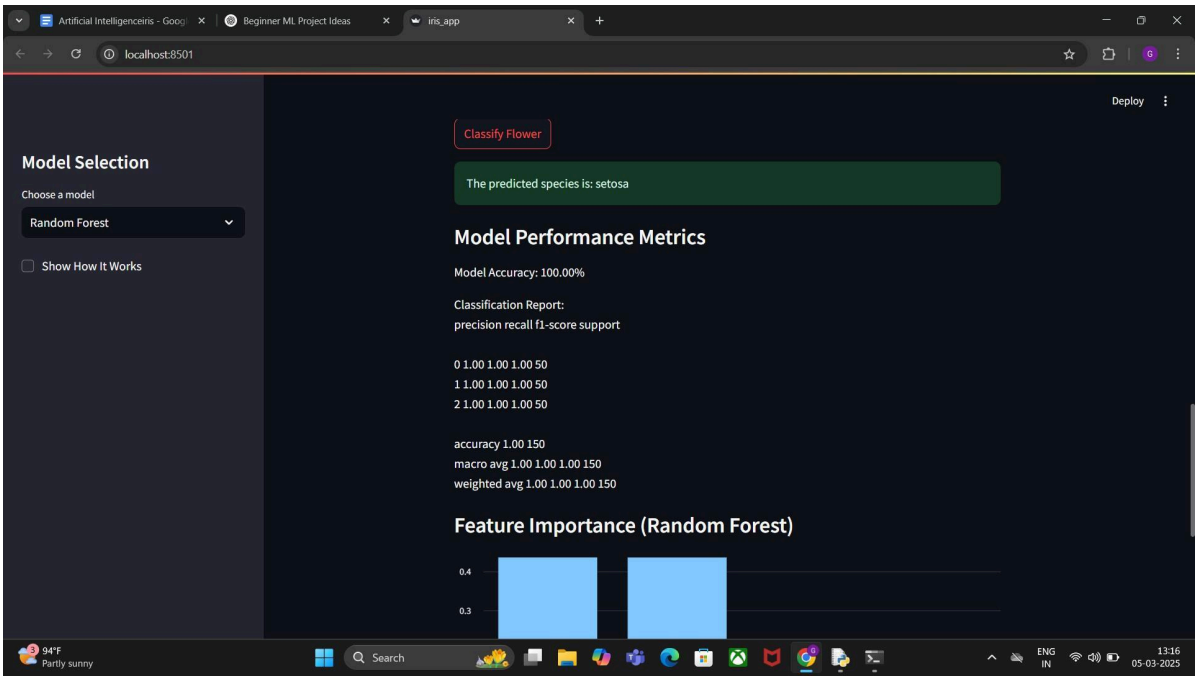

localhost:8501
Deploy
Model Selection
Choose a model
Random Forest
Show How It Works
sepal length (cm)
sepal width (cm)
petal length (cm)
petal width (cm)
Interactive Data Visualization
Iris Dataset Visualization
sepal width (cm)
sepal length (cm)
species
setosa
versicolor
virginica

For Random Forest

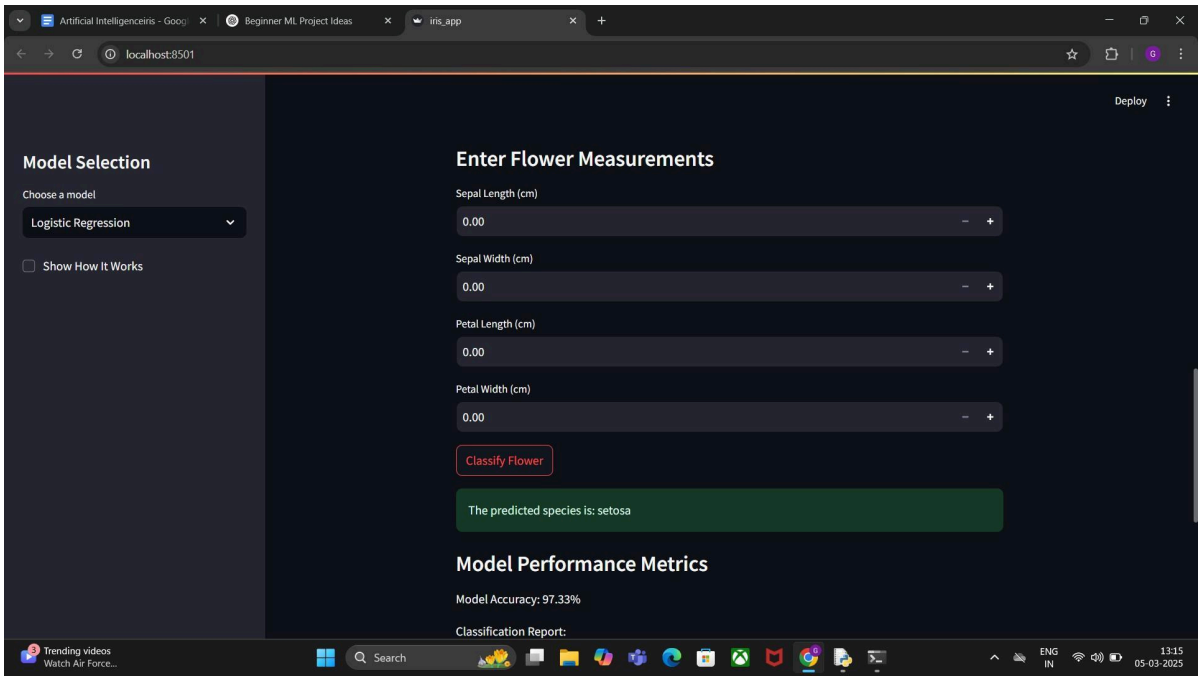


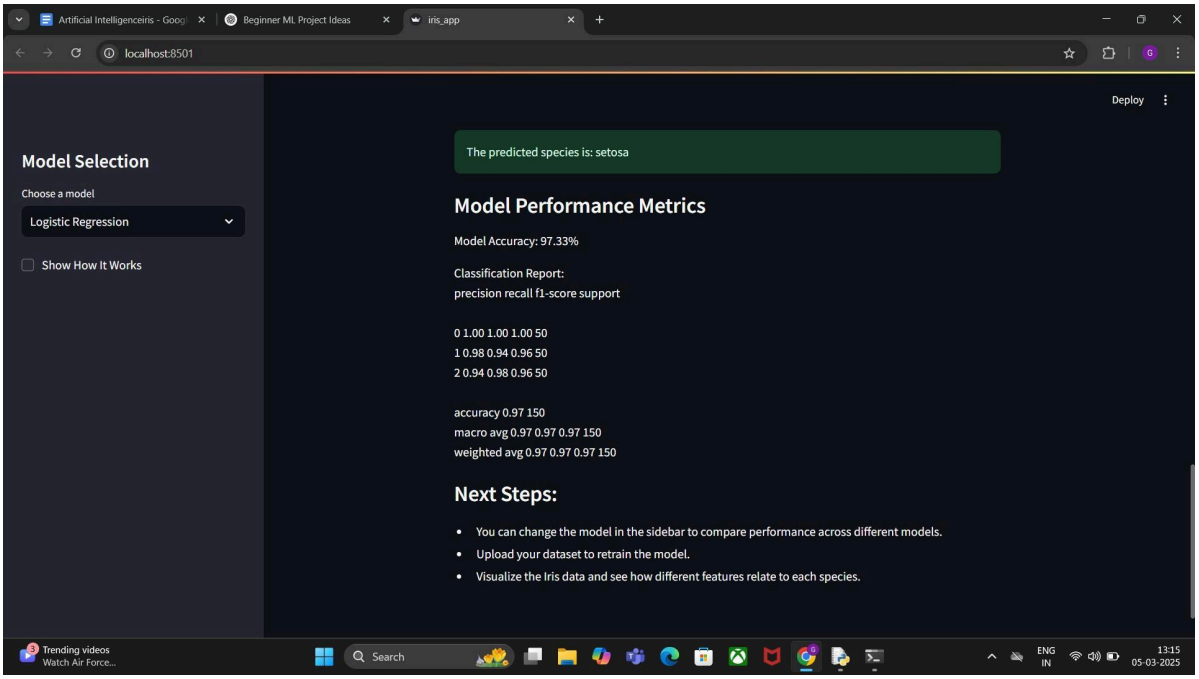



Enter Flower Measurements
Sepal Length (cm)
0.90
Sepal Width (cm)
1.40
Petal Length (cm)
1.30
Petal Width (cm)
1.40
Classify Flower
The predicted species is: setosa

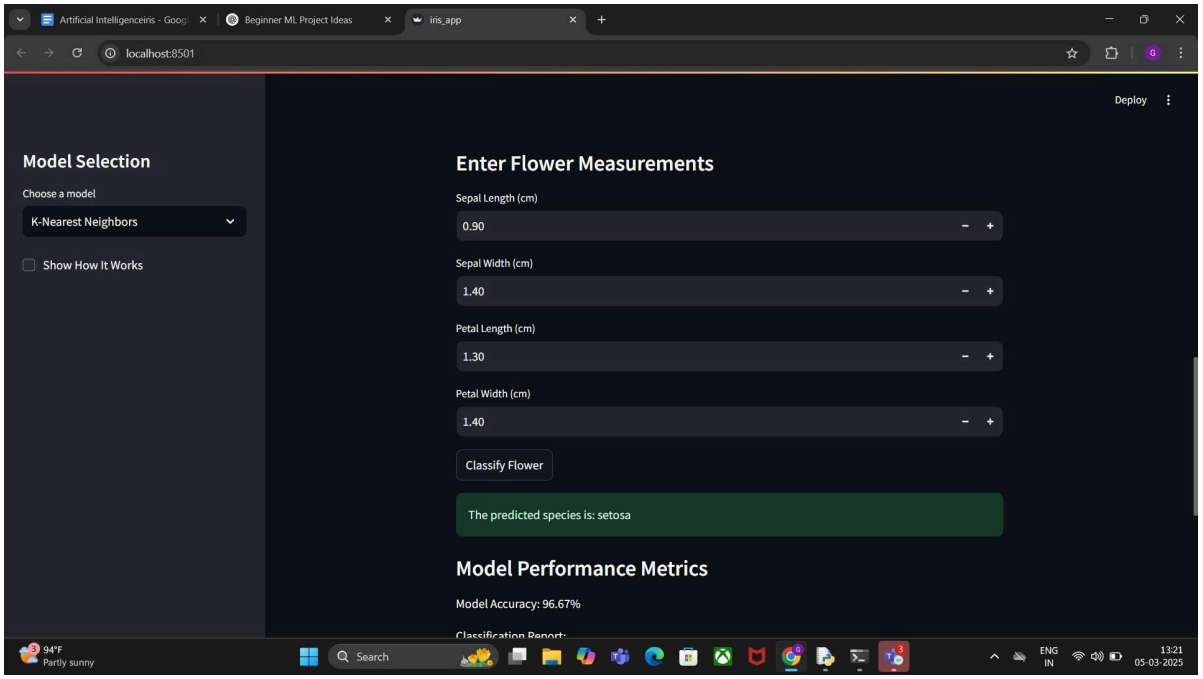

localhost:8501
Deploy
Model Selection
Choose a model
Random Forest
Show How It Works
Classify Flower
The predicted species is: setosa
Model Performance Metrics
Model Accuracy: 100.00%
Classification Report:
precision recall f1-score support
0 1.00 1.00 1.00 50
1 1.00 1.00 1.00 50
2 1.00 1.00 1.00 50
accuracy 1.00 150
macro avg 1.00 1.00 1.00 150
weighted avg 1.00 1.00 1.00 150
Feature Importance (Random Forest)
0.4
0.3
94°F
Partly sunny
Search
ENG
IN
13:16
05-03-2025

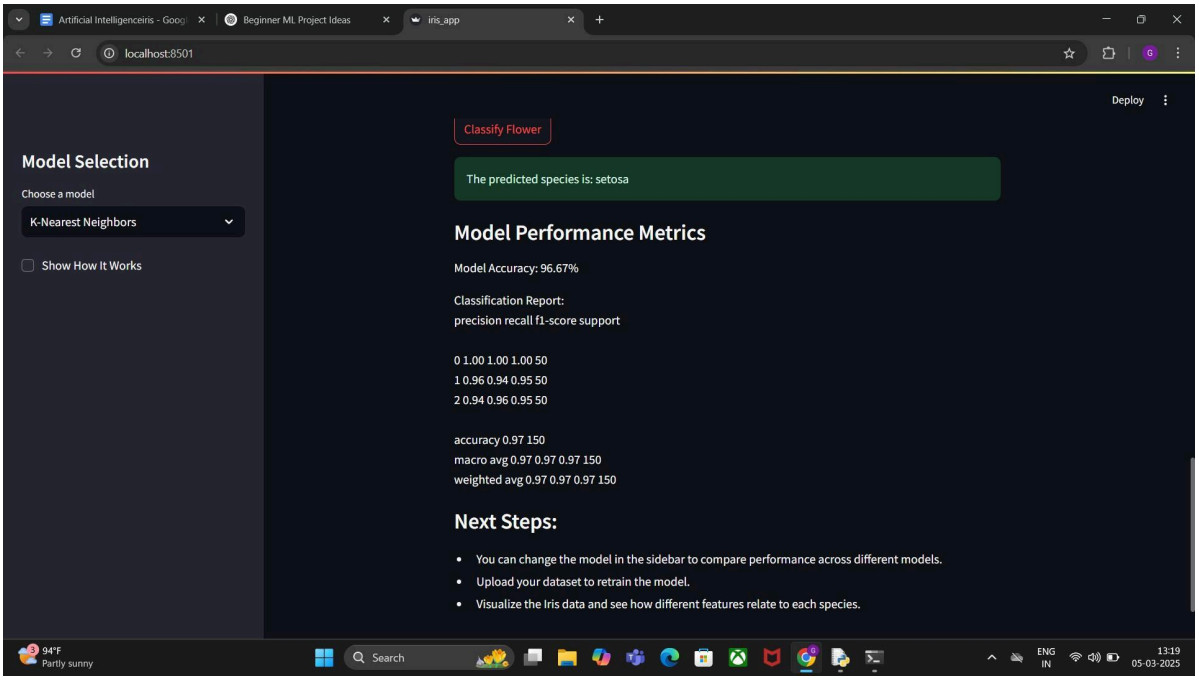
For Logistic Regression





For K-Nearest Neighbour





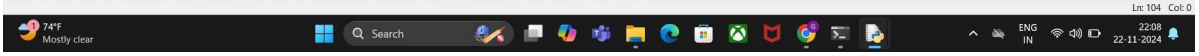
Code

iris_app.py - C:/Users/lenovo/OneDrive/Desktop/iris_app.py (3.11.9)

File Edit Format Run Options Window Help

```
import streamlit as st
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.datasets import load_iris
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load the Iris dataset
iris = load_iris()
X = iris.data
y = iris.target
df = pd.DataFrame(X, columns=iris.feature_names)
df['species'] = iris.target_names[y]

# Streamlit app title
st.title("Iris Flower Classification")

# Welcome message
st.write("This app classifies iris flowers into three species based on sepal and petal dimensions.")
st.write("You can also experiment with different models and explore the dataset.")

# Sidebar for model selection
st.sidebar.title("Model Selection")
model_option = st.sidebar.selectbox("Choose a model", ["Random Forest", "Logistic Regression", "K-Nearest Neighbors"])

# Select the model based on user input
if model_option == "Random Forest":
    model = RandomForestClassifier(random_state=42)
elif model_option == "Logistic Regression":
    model = LogisticRegression(max_iter=200)
else:
    model = KNeighborsClassifier()

# Train the model
model.fit(X, y)

# Show "How it works" section
if st.sidebar.checkbox('Show How It Works'):
    st.subheader("How It Works")
    st.write("""
    The model is trained using a well-known dataset called the Iris dataset, which contains measurements
    of flowers and their corresponding species. We use these features (sepal and petal length/width) to train
```

Ln: 104 Col: 0

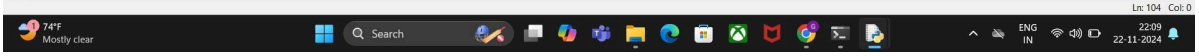


iris_app.py - C:/Users/lenovo/OneDrive/Desktop/iris_app.py (3.11.9)

File Edit Format Run Options Window Help

```
# Display dataset distribution (pairplot)
st.subheader("Iris Dataset Distribution")
sns.pairplot(df, hue='species')
st.pyplot(plt)

# Interactive Scatter Plot using Plotly
st.subheader("Interactive Data Visualization")
fig = px.scatter(df, x="sepal length (cm)", y="sepal width (cm)", color="species", title="Iris Dataset Visualization")
st.plotly_chart(fig)

# Input form for flower dimensions
st.subheader("Enter Flower Measurements")
sepal_length = st.number_input("Sepal Length (cm)", min_value=0.0, max_value=10.0, step=0.1)
sepal_width = st.number_input("Sepal Width (cm)", min_value=0.0, max_value=10.0, step=0.1)
petal_length = st.number_input("Petal Length (cm)", min_value=0.0, max_value=10.0, step=0.1)
petal_width = st.number_input("Petal Width (cm)", min_value=0.0, max_value=10.0, step=0.1)

# Predict button
if st.button("Classify Flower"):
    # Prepare input data
    input_data = np.array([[sepal_length, sepal_width, petal_length, petal_width]])

    # Make prediction
    prediction = model.predict(input_data)
    species = iris.target_names[prediction[0]]

    # Display the result
    st.success(f"The predicted species is: {species}")

    # Display model performance metrics
    st.subheader("Model Performance Metrics")
    y_pred = model.predict(X)
    accuracy = accuracy_score(y, y_pred)
    st.text(f"Model Accuracy: {accuracy*100:.2f}%")
    st.text(f"Classification Report: \n{classification_report(y, y_pred)}")

# Show Feature Importance Bar Chart for Random Forest
if model_option == "Random Forest":
    st.subheader("Feature Importance (Random Forest)")
    feature_importance = model.feature_importances_
    importance_df = pd.DataFrame({
        'Feature': iris.feature_names,
        'Importance': feature_importance
    })
    st.bar_chart(importance_df.set_index('Feature'))

# Footer
```

Ln: 104 Col: 0



Conclusion

This Iris Flower Classification app allows users to explore machine learning in a hands-on and interactive way. By allowing model selection, data exploration, and performance evaluation, it offers a complete tool for understanding the relationships between flower dimensions and species. Users can experiment with different models, visualize the Iris dataset, and evaluate model performance.

The app serves as an excellent educational tool for those learning about machine learning models, classification tasks, and data visualization. Additionally, it’s an easy-to-use platform for anyone looking to classify flowers based on their measurements.