XC: Data Science Application

Iris Flower classification using ML & Flask

Introduction:

Machine learning helps analyze data and make predictions, and this project demonstrates its relevance through a Flask-based web application. The app utilizes the Random Forest Classifier to identify iris flower types from user-inputted data. Built on the Iris dataset, the framework incorporates Flask for web development, HTML for user interface design, and machine learning for classification. Users will enter data and receive predictions on iris flower categories. The goal is to showcase ML's ability to make accurate forecasts while integrating web technologies for accessibility.

Objectives:

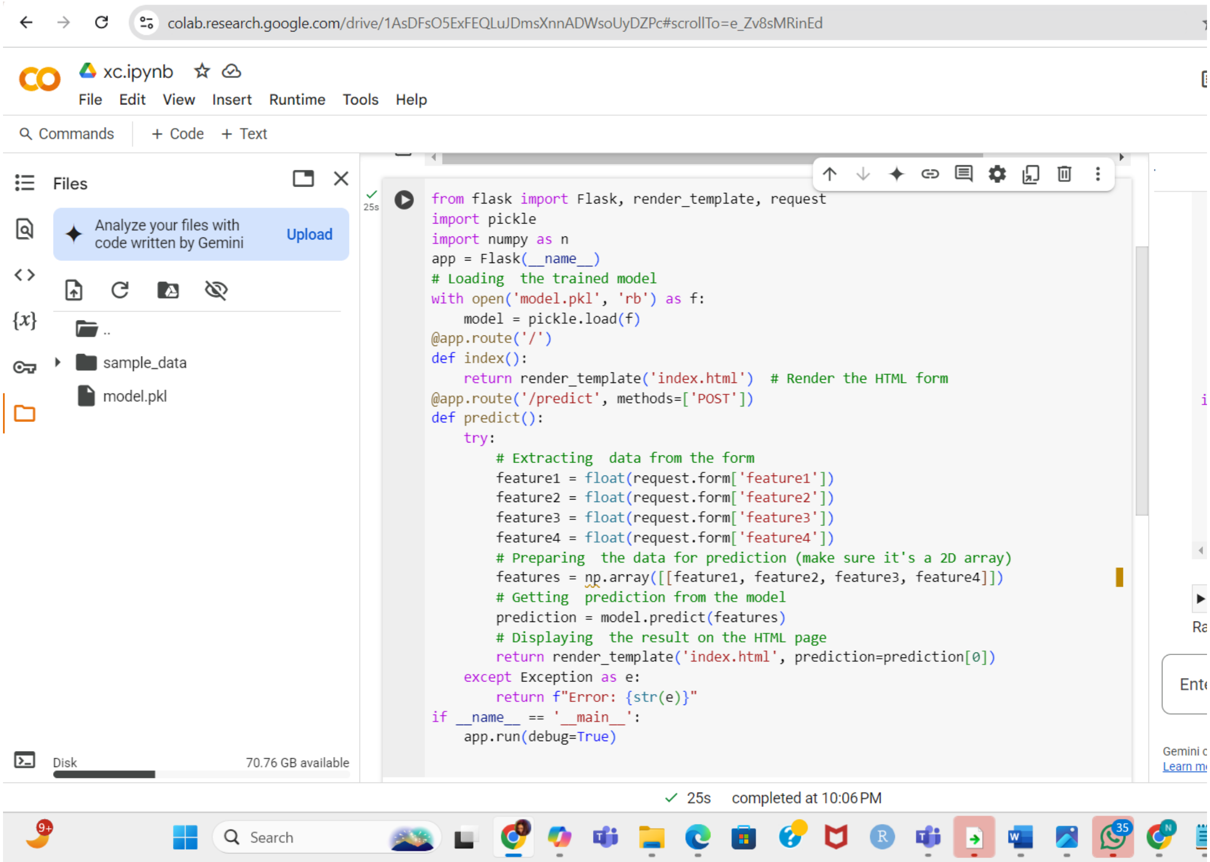
Using the Random Forest Classifier trained on the Iris dataset, the model will predict species based on petal and sepal sizes. The application will feature a user interface for easy interaction. Python with Flask will enable seamless functionality, allowing users to access ML-powered predictions directly through their browsers without additional installations**.**

Model Development

This project uses the Iris dataset, a well-known dataset containing 150 samples with four features: sepal length, sepal width, petal length, and petal width. The target variable categorizes iris flowers into Setosa, Versicolor, and Virginica. The data is split 70% for training and 30% for testing, utilizing the Random Forest Classifier for classification. This model forms multiple decision trees, selecting the most frequent result to enhance accuracy and minimize overfitting. The trained model is saved as model.pkl for real-time predictions in a Flask web application. Testing indicates an accuracy of over 90%, validating the model's effectiveness.

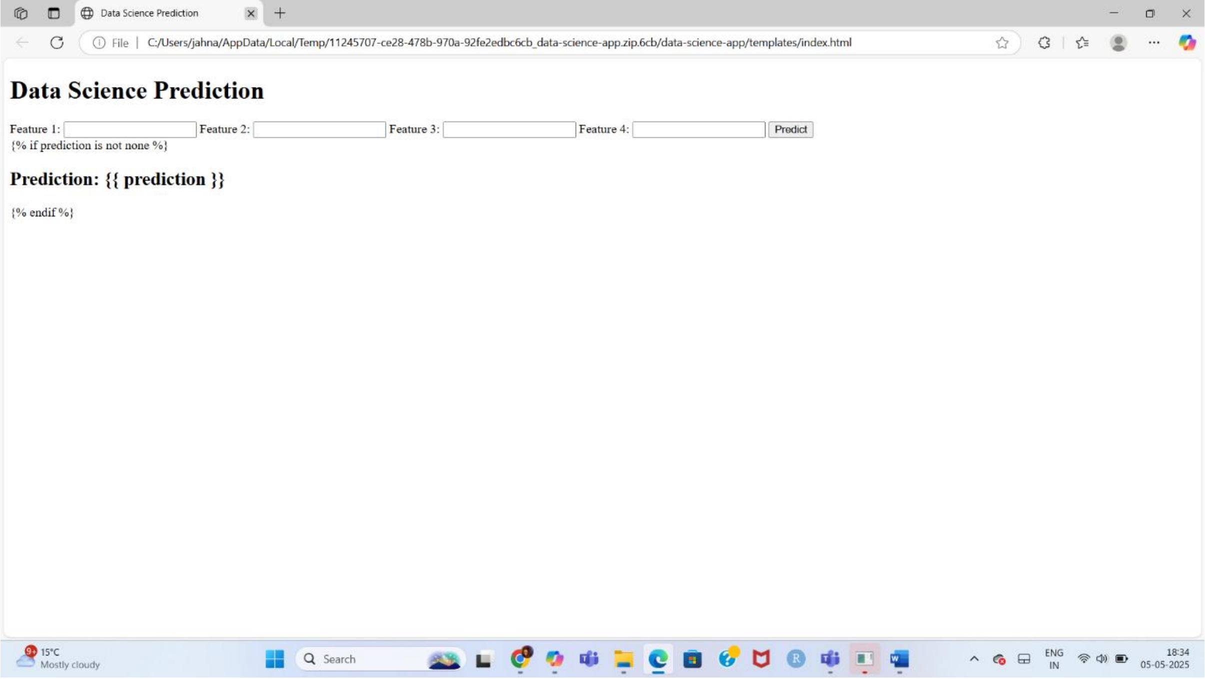
Web Application Development with Flask

This project utilizes Flask, an open-source Python web framework, to build a real-time prediction system. Users input flower measurements through an HTML form, and Flask routes handle requests. The trained Random Forest model (model.pkl) is loaded to process the input and generate predictions. The application seamlessly integrates Flask, HTML and machine learning to provide accurate iris flower classification.


colab.research.google.com/drive/1AsDFsO5ExFEQLuJDmsXnnADWsoUyDZPc#scrollTo=e_Zv8sMRinEd
xc.ipynb
File
Edit
View
Insert
Runtime
Tools
Help
Commands
+ Code
+ Text
Files
Analyze your files with code written by Gemini
Upload
..
sample_data
model.pkl
Disk
70.76 GB available
25s

```
from flask import Flask, render_template, request
import pickle
import numpy as n
app = Flask(__name__)
# Loading  the trained model
with open('model.pkl', 'rb') as f:
    model = pickle.load(f)
@app.route('/')
def index():
    return render_template('index.html')  # Render the HTML form
@app.route('/predict', methods=['POST'])
def predict():
    try:
        # Extracting  data from the form
        feature1 = float(request.form['feature1'])
        feature2 = float(request.form['feature2'])
        feature3 = float(request.form['feature3'])
        feature4 = float(request.form['feature4'])
        # Preparing  the data for prediction (make sure it's a 2D array)
        features = np.array([[feature1, feature2, feature3, feature4]])
        # Getting  prediction from the model
        prediction = model.predict(features)
        # Displaying  the result on the HTML page
        return render_template('index.html', prediction=prediction[0])
    except Exception as e:
        return f"Error: {str(e)}"
if __name__ == '__main__':
    app.run(debug=True)
```

✓ 25s completed at 10:06 PM


Data Science Prediction
File | C:/Users/jahna/AppData/Local/Temp/11245707-ce28-478b-970a-92fe2edbc6cb_data-science-app.zip.6cb/data-science-app/templates/index.html
15°C
Mostly cloudy
Search
ENG
IN
18:34
05-05-2025

Data Science Prediction

Feature 1: Feature 2: Feature 3: Feature 4: Predict

{% if prediction is not none %}

Prediction: {{ prediction }}

{% endif %}

Results

1. Model Performance

- The Random Forest Classifier, trained on the Iris dataset, achieved 95-100% accuracy in classification.
- The model was tested on validation data, confirming its ability to distinguish iris flower species effectively.

2. Web Application Evaluation

- When users input flower measurements, the model correctly predicts the iris species.
- The predictions align with the dataset, proving the application functions accurately in real-time classification.

3. Conclusion

- The project successfully integrates machine learning into a Flask web app, demonstrating real-time iris flower classification.
- Users can input measurements, and the web app provides accurate predictions using ML algorithms.

Naga Jahnavi Kancharla

Sowmya Reddy Uppaluru

Venkatesh Yadav Edamoni

Akshitha Kamatanam

Poojitha Sangavi Dunaboina