

Code

import streamlit as st

import pandas as pd

from sklearn.datasets import load\_iris

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

# Function to load data

def load\_data():

    data = load\_iris()

    df = pd.DataFrame(data.data, columns=data.feature\_names)

    df['target'] = data.target

    return df

# Function to train model and return predictions

def train\_model(df):

    X = df.drop('target', axis=1)

    y = df['target']

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

    scaler = StandardScaler()

    X\_train\_scaled = scaler.fit\_transform(X\_train)

    X\_test\_scaled = scaler.transform(X\_test)

    model = LogisticRegression()

    model.fit(X\_train\_scaled, y\_train)

    y\_pred = model.predict(X\_test\_scaled)

    accuracy = accuracy\_score(y\_test, y\_pred)

    return model, accuracy, y\_pred, y\_test

# Set up the Streamlit app

st.set\_page\_config(page\_title="Machine Learning Pipeline Demo", layout="wide")

# CSS to set background color

page\_bg\_color = '''

<style>

[data-testid="stAppViewContainer"] {

    background-color: #BFDDDF;  /\* Change this to your desired background color \*/

    color: black;

}

</style>

'''

st.markdown(page\_bg\_color, unsafe\_allow\_html=True)

# App title and description

st.markdown(

    """

    <div style="background-color: #70BDC2; padding: 5px;">

        <h1 style="text-align: center; color: white;">PROJECT DEMO</h1>

    </div>

    """,

    unsafe\_allow\_html=True,

)

# Display sample dataset

df = load\_data()

st.write("## Sample Dataset")

st.write(df.head())

# Train and display model accuracy on the sample dataset

model, accuracy, y\_pred, y\_test = train\_model(df)

st.write(f"## Model accuracy on sample dataset: {accuracy:.2f}")

# Map the numerical labels to species names

species\_mapping = {0: 'Setosa', 1: 'Versicolor', 2: 'Virginica'}

y\_pred\_names = [species\_mapping[label] for label in y\_pred]

y\_test\_names = [species\_mapping[label] for label in y\_test]

# Display some sample predictions

st.write("### Sample Predictions")

for i in range(5):

    st.write(f"Predicted: {y\_pred\_names[i]}, Actual: {y\_test\_names[i]}")

# File uploader for user dataset

st.write("## Upload Your Dataset")

uploaded\_file = st.file\_uploader("Choose a CSV file", type="csv")

if uploaded\_file is not None:

    user\_df = pd.read\_csv(uploaded\_file)

    st.write("### Uploaded Dataset")

    st.write(user\_df.head())

    # Check if the uploaded dataset has the necessary columns

    if set(df.columns).issubset(user\_df.columns):

        user\_model, user\_accuracy, user\_y\_pred, user\_y\_test = train\_model(user\_df)

        user\_y\_pred\_names = [species\_mapping[label] for label in user\_y\_pred]

        user\_y\_test\_names = [species\_mapping[label] for label in user\_y\_test]

        st.write(f"### Model accuracy on uploaded dataset: {user\_accuracy:.2f}")

        # Display some sample predictions on user data

        st.write("### Sample Predictions on Uploaded Data")

        for i in range(5):

            st.write(f"Predicted: {user\_y\_pred\_names[i]}, Actual: {user\_y\_test\_names[i]}")

    else:

        st.write("The uploaded dataset does not have the required columns.")

# Add a button to train the model on sample data

if st.button("Train model on sample data"):

    model, accuracy, y\_pred, y\_test = train\_model(df)

    st.write(f"Model accuracy on sample dataset: {accuracy:.2f}")

# Add a button to train the model on user data

if uploaded\_file is not None:

    if st.button("Train model on uploaded data"):

        user\_model, user\_accuracy, user\_y\_pred, user\_y\_test = train\_model(user\_df)

        user\_y\_pred\_names = [species\_mapping[label] for label in user\_y\_pred]

        user\_y\_test\_names = [species\_mapping[label] for label in user\_y\_test]

        st.write(f"Model accuracy on uploaded dataset: {user\_accuracy:.2f}")

        # Display some sample predictions on user data

        st.write("### Sample Predictions on Uploaded Data")

        for i in range(5):

            st.write(f"Predicted: {user\_y\_pred\_names[i]}, Actual: {user\_y\_test\_names[i]}")