1. **Selección de variables de entrada**:  
   El objetivo es predecir el mejor momento para sembrar cereales. Podrías usar variables como temperatura, precipitación, humedad, entre otras, para hacer la predicción. Crea una variable de objetivo (por ejemplo, una columna binaria llamada "sembrado" que indique si las condiciones son favorables para sembrar).

python

CopiarEditar

# Aquí puedes definir una regla basada en el clima, por ejemplo, si la precipitación es suficiente y la temperatura es adecuada  
df['sembrado'] = (df['precip'] >= 40) & (df['tmed'] >= 8) # Ejemplo de regla

1. **División en conjunto de entrenamiento y prueba**:

python

CopiarEditar

from sklearn.model\_selection import train\_test\_split  
  
X = df[['precip', 'tmed', 'tmin', 'tmax', 'humedad\_relativa']] # Ejemplo de características  
y = df['sembrado'] # Etiqueta de si es adecuado sembrar o no  
  
X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

### **2. Entrenamiento del modelo de machine learning**

Usaremos un modelo como un **Random Forest** o **Logistic Regression** para predecir si es adecuado sembrar o no. Aquí un ejemplo usando **Random Forest**:

python

CopiarEditar

from sklearn.ensemble import RandomForestClassifier  
from sklearn.metrics import accuracy\_score, classification\_report  
  
# Entrenamiento del modelo  
model = RandomForestClassifier(n\_estimators=100, random\_state=42)  
model.fit(X\_train, y\_train)  
  
# Predicción en conjunto de prueba  
y\_pred = model.predict(X\_test)  
  
# Evaluación del modelo  
print("Accuracy:", accuracy\_score(y\_test, y\_pred))  
print("Classification Report:\n", classification\_report(y\_test, y\_pred))

### **3. Desarrollo de la aplicación Flask**

Ahora que tienes el modelo entrenado, puedes crear una aplicación en Flask que prediga si es adecuado sembrar en una fecha futura basada en los valores meteorológicos introducidos por el usuario.

#### **Pasos para crear la app Flask:**

1. **Instala Flask**:

bash

CopiarEditar

pip install flask

1. **Estructura básica de la app Flask**:

python

CopiarEditar

from flask import Flask, render\_template, request  
import pickle # Para cargar el modelo preentrenado  
  
# Cargar el modelo entrenado  
with open('modelo\_rf.pkl', 'rb') as model\_file:  
 model = pickle.load(model\_file)  
  
app = Flask(\_\_name\_\_)  
  
@app.route('/')  
def index():  
 return render\_template('index.html')  
  
@app.route('/predecir', methods=['POST'])  
def predecir():  
 # Obtener datos del formulario  
 precip = float(request.form['precip'])  
 tmed = float(request.form['tmed'])  
 tmin = float(request.form['tmin'])  
 tmax = float(request.form['tmax'])  
 humedad\_relativa = float(request.form['humedad\_relativa'])  
  
 # Predecir si es adecuado sembrar  
 prediccion = model.predict([[precip, tmed, tmin, tmax, humedad\_relativa]])  
  
 # Convertir la predicción en mensaje  
 if prediccion == 1:  
 resultado = "¡Es un buen momento para sembrar!"  
 else:  
 resultado = "No es un buen momento para sembrar."  
  
 return render\_template('resultado.html', resultado=resultado)  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(debug=True)

1. **Plantillas HTML** (index.html y resultado.html) para recibir los datos del usuario y mostrar los resultados.

* **index.html**: Una forma básica para que el usuario ingrese los datos meteorológicos.

html

CopiarEditar

<!DOCTYPE html>  
<html lang="es">  
<head>  
 <meta charset="UTF-8">  
 <title>Predicción de Siembra</title>  
</head>  
<body>  
 <h1>Predicción de Siembra para Cereal</h1>  
 <form action="/predecir" method="POST">  
 <label for="precip">Precipitación (mm):</label><br>  
 <input type="number" name="precip" required><br>  
  
 <label for="tmed">Temperatura Media (°C):</label><br>  
 <input type="number" name="tmed" required><br>  
  
 <label for="tmin">Temperatura Mínima (°C):</label><br>  
 <input type="number" name="tmin" required><br>  
  
 <label for="tmax">Temperatura Máxima (°C):</label><br>  
 <input type="number" name="tmax" required><br>  
  
 <label for="humedad\_relativa">Humedad Relativa (%):</label><br>  
 <input type="number" name="humedad\_relativa" required><br>  
  
 <button type="submit">Predecir</button>  
 </form>  
</body>  
</html>

* **resultado.html**: Para mostrar si es un buen momento para sembrar o no.

html

CopiarEditar

<!DOCTYPE html>  
<html lang="es">  
<head>  
 <meta charset="UTF-8">  
 <title>Resultado de la Predicción</title>  
</head>  
<body>  
 <h1>{{ resultado }}</h1>  
 <a href="/">Volver</a>  
</body>  
</html>

### **4. Guardar el modelo**

Antes de poder usar el modelo en Flask, debes guardarlo en un archivo para cargarlo posteriormente:

python

CopiarEditar

import pickle  
  
# Guardar el modelo  
with open('modelo\_rf.pkl', 'wb') as model\_file:  
 pickle.dump(model, model\_file)

### **5. Ejecutar la aplicación**

Para ejecutar la app, solo debes correr el archivo de Flask:

bash

CopiarEditar

python app.py

Visita <http://127.0.0.1:5000> en tu navegador y podrás interactuar con la aplicación.

Este es un enfoque general. A medida que avances en el desarrollo de tu aplicación, podrías mejorar el modelo, las predicciones y la interfaz de usuario. Si necesitas más ayuda, no dudes en preguntar. ¡Buena suerte con tu proyecto! 🚜🌾