

✓ Recomendación tipo de cultivo

Generaremos un modelo de máquina de soporte vectorial SVM para recomendar un tipo de cultivo dependiendo de la información del suelo y ambiental. El modelo luego se pondrá en producción usando google cloud y streamlit.

Dataset:

El dataset usado se denomina "Crop Recommendation Dataset" y se encuentra público en kaggle. Puedes encontrarlo en el siguiente enlace:

[Dataset Kaggle](#)

Variables independientes:

- Nitrógeno
- Fósforo
- Potasio
- Temperatura
- Humedad
- pH
- Lluvia

```
import sys
print(sys.version)
```

```
3.11.12 (main, Apr 9 2025, 08:55:54) [GCC 11.4.0]
```

✓ 1. Instalación scikit-learn

```
!pip install scikit-learn==0.24.1
```

```
Collecting scikit-learn==0.24.1
  Downloading scikit-learn-0.24.1.tar.gz (7.4 MB)
    7.4/7.4 MB 20.9 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: numpy>=1.13.3 in /usr/local/lib/python3.11/dist-packages (from scikit-learn==0.24.1) (2.0.2)
Requirement already satisfied: scipy>=0.19.1 in /usr/local/lib/python3.11/dist-packages (from scikit-learn==0.24.1) (1.15.2)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.11/dist-packages (from scikit-learn==0.24.1) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.11/dist-packages (from scikit-learn==0.24.1) (3.6.0)
Building wheels for collected packages: scikit-learn
  error: subprocess-exited-with-error

  × Building wheel for scikit-learn (pyproject.toml) did not run successfully.
    | exit code: 1
    | See above for output.

  note: This error originates from a subprocess, and is likely not a problem with pip.
Building wheel for scikit-learn (pyproject.toml) ... error
ERROR: Failed building wheel for scikit-learn
Failed to build scikit-learn
ERROR: ERROR: Failed to build installable wheels for some pyproject.toml based projects (scikit-learn)
```

✓ 2. Importamos librerías

```
import numpy as np
from sklearn import svm
import pandas as pd
import pickle
from sklearn.model_selection import train_test_split
import sklearn
sklearn.__version__
```

```
'1.6.1'
```

```
!python3 --version
```

```
Python 3.11.12
```

3. Cargamos dataset

```
df= pd.read_csv("/content/Crop_recommendation.csv")
df.head()
```

	N	P	K	temperature	humidity	ph	rainfall	label
0	90	42	43	20.879744	82.002744	6.502985	202.935536	rice
1	85	58	41	21.770462	80.319644	7.038096	226.655537	rice
2	60	55	44	23.004459	82.320763	7.840207	263.964248	rice
3	74	35	40	26.491096	80.158363	6.980401	242.864034	rice
4	78	42	42	20.130175	81.604873	7.628473	262.717340	rice

```
df.describe()
```

	N	P	K	temperature	humidity	ph	rainfall
count	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000	2200.000000
mean	50.551818	53.362727	48.149091	25.616244	71.481779	6.469480	103.463655
std	36.917334	32.985883	50.647931	5.063749	22.263812	0.773938	54.958389
min	0.000000	5.000000	5.000000	8.825675	14.258040	3.504752	20.211267
25%	21.000000	28.000000	20.000000	22.769375	60.261953	5.971693	64.551686
50%	37.000000	51.000000	32.000000	25.598693	80.473146	6.425045	94.867624
75%	84.250000	68.000000	49.000000	28.561654	89.948771	6.923643	124.267508
max	140.000000	145.000000	205.000000	43.675493	99.981876	9.935091	298.560117

```
# Obtenemos variables independientes
X = df.drop(["label"],axis = 1)
X.head()
```

	N	P	K	temperature	humidity	ph	rainfall
0	90	42	43	20.879744	82.002744	6.502985	202.935536
1	85	58	41	21.770462	80.319644	7.038096	226.655537
2	60	55	44	23.004459	82.320763	7.840207	263.964248
3	74	35	40	26.491096	80.158363	6.980401	242.864034
4	78	42	42	20.130175	81.604873	7.628473	262.717340

```
# Obtenemos variable dependiente
Y = df.pop('label')
Y
```

	label
0	rice
1	rice
2	rice
3	rice
4	rice
...	...
2195	coffee
2196	coffee
2197	coffee
2198	coffee
2199	coffee

2200 rows × 1 columns

dtype: object

```
# Separamos datos para ajuste y prueba
X_train, X_test, y_train, y_test = train_test_split(X, Y, train_size=0.8, test_size=0.2, random_state=100)
```

✓ 4. Ajuste del modelo SVM


```
# Creamos el modelo SVM para clasificacion con kernel lineal/rbf y entrenamos el modelo
model = svm.SVC(kernel='linear', C=100).fit(X_train, y_train)
```

```
# Grabamos el modelo en el directorio
pkl_filename = "pickle_model.pkl"
with open(pkl_filename, 'wb') as file:
    pickle.dump(model, file)
```

```
# Cargamos el modelo
pkl_filename = "pickle_model.pkl"
with open(pkl_filename, 'rb') as file:
    model = pickle.load(file)
```


✓ 5. Desempeño del modelo

```
# Encontramos el accuracy promedio usando datos de test
score = model.score(X_test, y_test)
print(score)
```

 0.9931818181818182

✓ 6. Probamos con una muestra nueva

```
x_in = np.asarray([90, 42, 50, 20.8, 80, 6, 200]).reshape(1, -1)
predicts = model.predict(x_in)
predicts[0]
```

 'rice'