

Repositorio en GitHub

## Intergrado por:

Geremi Armando Venegas Dueñas y Eder Elvis Luna Ochochoque

## ¿Qué es Optuna?

Optuna es un framework de optimización de hiperparámetros de código abierto diseñado para ser fácil de usar y eficiente. Es especialmente útil en el campo del aprendizaje automático, donde la optimización de hiperparámetros es una tarea común y crucial para mejorar el rendimiento de los modelos. Optuna permite definir un espacio de búsqueda para los hiperparámetros y utiliza algoritmos avanzados para encontrar la mejor combinación de estos parámetros.

## Tutorial

### Primer Paso: Creación del Proyecto

Antes de comenzar, asegúrese de tener instalado Python. Luego, cree una carpeta con un nombre relacionado con el tema (por ejemplo, **Optuna**), como se muestra en las imágenes a continuación. Esta carpeta se abrirá con Visual Studio Code:

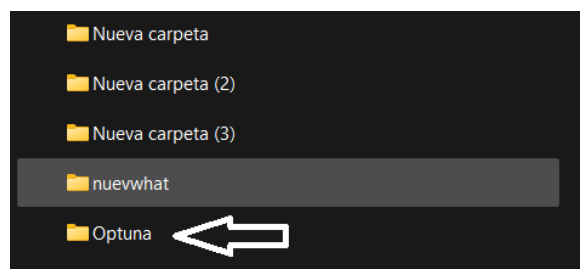


Figure 1: Carpeta del proyecto creada

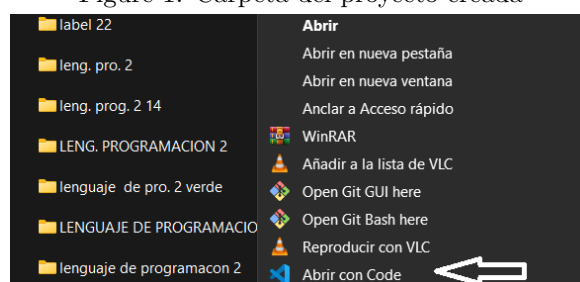
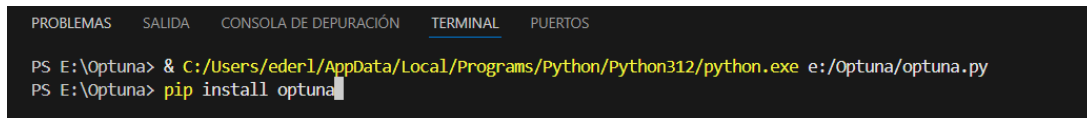


Figure 2: Carpeta abierta en Visual Studio Code

### Segundo Paso: Instalación de Optuna

Una vez dentro de Visual Studio Code, cree un nuevo archivo Python, por ejemplo: `optuna.py`. Luego, ejecute el archivo y se abrirá el terminal donde procederemos a instalar Optuna con el siguiente comando:

```
pip install optuna
```



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS

PS E:\Optuna> & C:/Users/eder1/AppData/Local/Programs/Python/Python312/python.exe e:/Optuna/optuna.py
PS E:\Optuna> pip install optuna
```

Figure 3: Instalación de Optuna en el terminal

## Tercer Paso: Pegar el código de optuna

Aquí tienes el código de optuna pero necesitas una base de datos:

```
1 import optuna
2
3 # Función objetivo para optimizar
4 def objective(trial):
5     # Definir los hiperparámetros a optimizar
6     n_estimators = trial.suggest_int("n_estimators", 2, 20)
7     max_depth = int(trial.suggest_float("max_depth", 1, 32, log=True))
8
9     # Crear y evaluar el modelo (aquí debes incluir tu modelo y datos)
10    clf = ... # Tu modelo (por ejemplo, RandomForestClassifier)
11    score = ... # Métrica a optimizar (por ejemplo, accuracy)
12    return score
13
14 # Crear un estudio de Optuna y optimizar
15 study = optuna.create_study(direction="maximize") # Maximizar la métrica
16 study.optimize(objective, n_trials=100) # Número de intentos
17
18 # Mostrar los mejores resultados
19 trial = study.best_trial
20 print("Mejor métrica:", trial.value)
21 print("Mejores hiperparámetros:", trial.params)
```

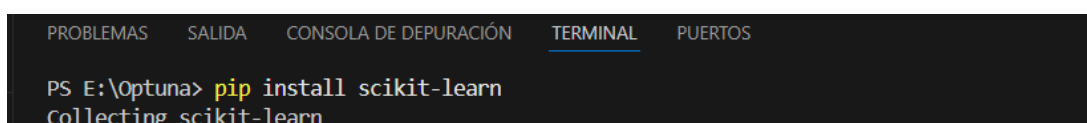
## Base de Datos de IRIS

El conjunto de datos Iris es un dataset clásico en machine learning que contiene información sobre 150 muestras de flores Iris, pertenecientes a tres especies: setosa, versicolor y virginica. Para cada muestra, se midieron cuatro características:

- Longitud del sépalo.
- Ancho del sépalo.
- Longitud del pétalo.
- Ancho del pétalo.

Para visualizar los datos de iris se debe instalar lo siguiente:

```
pip install scikit-learn
```



```
PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS

PS E:\Optuna> pip install scikit-learn
Collecting scikit-learn
```

Figure 4: Enter Caption

## Codigo para Visualizar Base de Datos de IRIS

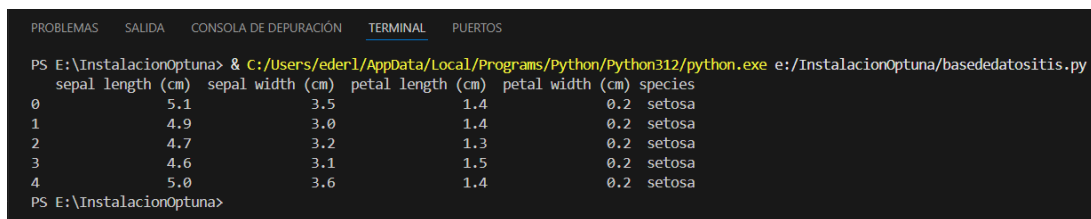
Crear un nuevo archivo con el nombre que desees y pega el siguiente codigo:

```

1 from sklearn.datasets import load_iris
2 import pandas as pd
3
4 # Cargar el dataset Iris
5 iris = load_iris()
6
7 # Convertir a un DataFrame de pandas para facilitar la visualizaci n
8 df = pd.DataFrame(iris.data, columns=iris.feature_names)
9
10 # Agregar la columna de especies (etiquetas)
11 df['species'] = iris.target
12
13 # Mapear los n meros de las clases a nombres de especies
14 df['species'] = df['species'].map({0: 'setosa', 1: 'versicolor', 2: 'virginica'})
15
16 # Mostrar las primeras filas del DataFrame
17 print(df.head())

```

Al ejecutar te mostrara los cinco primeros datos:



```

PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS
PS E:\InstalacionOptuna> & C:/Users/eder1/AppData/Local/Programs/Python/Python312/python.exe e:/InstalacionOptuna/basededatositis.py
  sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  species
0              5.1              3.5              1.4              0.2  setosa
1              4.9              3.0              1.4              0.2  setosa
2              4.7              3.2              1.3              0.2  setosa
3              4.6              3.1              1.5              0.2  setosa
4              5.0              3.6              1.4              0.2  setosa
PS E:\InstalacionOptuna>

```

Figure 5: Enter Caption

## Codigo de Optuna con Base de Datos de IRIS

Puedes reemplazar o crear otro archivo para ejecutar el siguiente codigo:

```

1 import optuna
2 import sklearn.datasets
3 import sklearn.ensemble
4 import sklearn.model_selection
5
6 def objective(trial):
7     iris = sklearn.datasets.load_iris()
8
9     n_estimators = trial.suggest_int("n_estimators", 2, 20)
10    max_depth = int(trial.suggest_float("max_depth", 1, 32, log=True))
11
12    clf = sklearn.ensemble.RandomForestClassifier(n_estimators=n_estimators, max_depth=
        max_depth)
13
14    return sklearn.model_selection.cross_val_score(
15        clf, iris.data, iris.target, n_jobs=-1, cv=3
16    ).mean()
17
18 study = optuna.create_study(direction="maximize")
19 study.optimize(objective, n_trials=100)
20
21 trial = study.best_trial

```

```

22
23 print("Accuracy:{}".format(trial.value))
24 print("Best hyperparameters:{}".format(trial.params))

```

la impresion que te mostrara lo siguiente:

```

PS E:\InstalacionOptuna> & C:/Users/eder1/AppData/Local/Programs/Python/Python312/python.exe e:/InstalacionOptuna/optuna_example.py
[I 2025-02-09 11:02:18,973] A new study created in memory with name: no-name-ffdc81bd-ba3d-4eb6-9a1d-eb4e0a577498
[I 2025-02-09 11:02:23,516] Trial 0 finished with value: 0.9533333333333333 and parameters: {'n_estimators': 5, 'max_depth': 3.9595439848785516}. Best is trial 0 with value: 0.9533333333333333.
[I 2025-02-09 11:02:25,491] Trial 1 finished with value: 0.9666666666666667 and parameters: {'n_estimators': 7, 'max_depth': 20.16316141999073}. Best is trial 1 with value: 0.9666666666666667.
[I 2025-02-09 11:02:27,345] Trial 2 finished with value: 0.94 and parameters: {'n_estimators': 6, 'max_depth': 21.96605339881194}. Best is trial 1 with value: 0.9666666666666667.
[I 2025-02-09 11:02:29,256] Trial 3 finished with value: 0.9666666666666667 and parameters: {'n_estimators': 15, 'max_depth': 19.238087883980242}. Best is trial 1 with value: 0.9666666666666667.
[I 2025-02-09 11:02:29,300] Trial 4 finished with value: 0.9533333333333333 and parameters: {'n_estimators': 10, 'max_depth': 24.53465325665927}. Best is trial 1 with value: 0.9666666666666667.
[I 2025-02-09 11:02:29,346] Trial 5 finished with value: 0.96 and parameters: {'n_estimators': 8, 'max_depth': 11.29549749002239}. Best is trial 1 with value: 0.9666666666666667.

```

Figure 6: Enter Caption

### Interpretacion de los resultados mostrados:

La ejecución de un estudio de optimización de hiperparámetros utilizando Optuna en un modelo de Random Forest para el conjunto de datos Iris.

[I 2025-02-09 11:02:18,973] A new study created in memory  
with name: no-name-ffdc81bd-ba3d-4eb6-9a1d-eb4e0a577498

**no-name-ffdc81bd-ba3d-4eb6-9a1d-eb4e0a577498**

(es un identificador único generado automáticamente).

### Resultados de cada trial

Cada línea que comienza con [I representa un trial (prueba) en el que Optuna prueba una combinación específica de hiperparámetros. Por ejemplo:

```

1 [I 2025-02-09 11:02:23,516] Trial 0 finished with value: 0.9533333333333333
2 and parameters: {'n_estimators': 5, 'max_depth': 3.9595439848785516}.
3 Best is trial 0 with value: 0.9533333333333333.

```

- **Trial 0:** Es la primera prueba que realiza Optuna.
- **value: 0.9533333333333333:** Es la precisión (accuracy) obtenida por el modelo con los hiperparámetros probados en este trial.
- **parameters:** Los valores de los hiperparámetros probados en este trial:
  - **n\_estimators:** Número de árboles en el Random Forest (en este caso, 5).
  - **max\_depth:** Profundidad máxima de los árboles (en este caso, ~3.96).
- **Best is trial 0:** Indica que, hasta este momento, este es el mejor resultado encontrado.

### Mejor resultado final

Al final de la ejecución, Optuna muestra el mejor resultado encontrado después de todos los trials:

```

1 Accuracy: 0.9733333333333333
2 Best hyperparameters: {'n_estimators': 3, 'max_depth': 10.021672710092872}

```

- **Accuracy:** La precisión máxima alcanzada por el modelo (en este caso, ~97.33%).
- **Best hyperparameters:** Los valores óptimos de los hiperparámetros que dieron lugar a la mejor precisión:

- **n\_estimators**: 3 (número de árboles).
- **max\_depth**: ~10.02 (profundidad máxima de los árboles).

## Interpretación de los resultados

1. **Precisión (Accuracy)** La precisión del 97.33% significa que el modelo clasifica correctamente el 97.33% de las muestras en el conjunto de datos Iris.

Esto es un muy buen resultado, ya que el dataset Iris es relativamente simple y las clases están bien separadas.

2. **Hiperparámetros óptimos**

- **n\_estimators = 3**: El modelo funciona mejor con solo 3 árboles en el Random Forest. Esto puede deberse a que el dataset Iris es pequeño y no requiere un modelo muy complejo.
- **max\_depth = 10.02**: La profundidad máxima de los árboles es de aproximadamente 10. Esto indica que los árboles no necesitan ser muy profundos para clasificar correctamente las muestras.

3. **Evolución de los trials** Durante los 100 trials, Optuna probó diferentes combinaciones de **n\_estimators** y **max\_depth**.

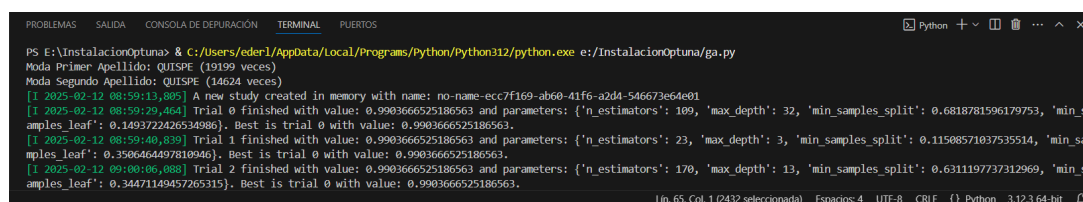
Algunos trials tuvieron una precisión más baja (por ejemplo, 0.7533 en el Trial 61), lo que indica que esas combinaciones de hiperparámetros no eran óptimas.

El mejor resultado se encontró en el Trial 15, con una precisión del 97.33%.

## Código de Optuna con base de datos en formato csv

Para aplicar Optuna en la optimización de hiperparámetros con la base de datos de "Pensión 65", primero necesitamos definir un problema de machine learning. Dado que los datos que has mostrado son principalmente categóricos y de identificación (como DNI, nombres, ubicaciones, etc.), no está claro cuál sería la variable objetivo o qué tipo de problema de machine learning podríamos resolver (clasificación, regresión, etc.).

## Vizualización de la base de datos



```

PS E:\InstalacionOptuna> & C:/Users/eder1/AppData/Local/Programs/Python/Python312/python.exe e:/InstalacionOptuna/ga.py
Moda Primer Apellido: QUISPE (19199 veces)
Moda Segundo Apellido: QUISPE (14624 veces)
[I 2025-02-12 08:59:13.805] A new study created in memory with name: no-name-ecc7f169-ab60-41f6-a2d4-546673e64e01
[I 2025-02-12 08:59:29.464] Trial 0 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 109, 'max_depth': 32, 'min_samples_split': 0.6818781596179753, 'min_s
amples_leaf': 0.1493722426534986}. Best is trial 0 with value: 0.9903666525186563.
[I 2025-02-12 08:59:40.839] Trial 1 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 23, 'max_depth': 3, 'min_samples_split': 0.11508571037535514, 'min_s
amples_leaf': 0.3506464497810946}. Best is trial 0 with value: 0.9903666525186563.
[I 2025-02-12 09:00:06.088] Trial 2 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 170, 'max_depth': 13, 'min_samples_split': 0.6311197737312969, 'min_s
amples_leaf': 0.34471149457265315}. Best is trial 0 with value: 0.9903666525186563.
  
```

Figure 7: Enter Caption

La base de datos se encuentra en [datosabiertos.gob.pe](https://datosabiertos.gob.pe) puedes buscar y descargarlo

Sin embargo, supongamos que queremos predecir la columna **TIPO.USUARIO** (que parece ser categórica) en función de las otras columnas. Esto sería un problema de clasificación.

A continuación, te muestro cómo podrías adaptar el código de Optuna para trabajar con estos datos. Pasos a seguir:

1. **Cargar los datos**: Leer el archivo CSV y preparar los datos para el modelo.
2. **Preprocesamiento**: Convertir las variables categóricas en numéricas (usando, por ejemplo, One-HotEncoding o LabelEncoding).

3. **Definir el objetivo:** Crear una función objetivo que Optuna optimizará.
4. **Optimización:** Usar Optuna para encontrar los mejores hiperparámetros.

## Código de Optuna con Base de Datos de [datosabiertos.gob.pe](https://datosabiertos.gob.pe)

```
1 import optuna
2 import pandas as pd
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.model_selection import cross_val_score
5 from sklearn.preprocessing import LabelEncoder
6 from sklearn.model_selection import train_test_split
7
8 # Cargar los datos
9 data = pd.read_csv("trama_U0_usuarios_202410.csv", encoding="latin-1")
10 # Preprocesamiento
11 # Convertir variables categóricas a numéricas
12 label_encoders = {}
13 for column in data.select_dtypes(include=['object']).columns:
14     le = LabelEncoder()
15     data[column] = le.fit_transform(data[column])
16     label_encoders[column] = le
17
18 # Separar características (X) y variable objetivo (y)
19 X = data.drop(columns=["TIPO_USUARIO"])
20 y = data["TIPO_USUARIO"]
21
22 # Dividir los datos en entrenamiento y prueba (opcional, pero recomendado)
23 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
24
25 # Función objetivo para Optuna
26 def objective(trial):
27     n_estimators = trial.suggest_int("n_estimators", 2, 200)
28     max_depth = trial.suggest_int("max_depth", 2, 32, log=True)
29     min_samples_split = trial.suggest_float("min_samples_split", 0.1, 1.0)
30     min_samples_leaf = trial.suggest_float("min_samples_leaf", 0.1, 0.5)
31
32     clf = RandomForestClassifier(
33         n_estimators=n_estimators,
34         max_depth=max_depth,
35         min_samples_split=min_samples_split,
36         min_samples_leaf=min_samples_leaf,
37         random_state=42
38     )
39
40     # Usar validación cruzada para evaluar el modelo
41     score = cross_val_score(clf, X_train, y_train, cv=3, n_jobs=-1).mean()
42     return score
43
44 # Crear un estudio de Optuna y optimizar
45 study = optuna.create_study(direction="maximize")
46 study.optimize(objective, n_trials=100)
47
48 # Obtener los mejores hiperparámetros
49 trial = study.best_trial
50 print("Best accuracy: {}".format(trial.value))
51 print("Best hyperparameters: {}".format(trial.params))
```

la impresion que te mostrara lo siguiente:

```

PS E:\InstalacionOptuna> & C:/Users/eder1/AppData/Local/Programs/Python/Python312/python.exe e:/InstalacionOptuna/ga.py
Moda Primer Apellido: QUISPE (19199 veces)
Moda Segundo Apellido: QUISPE (14624 veces)
[I 2025-02-12 08:59:13.895] A new study created in memory with name: no-name-ec07f169-ab60-41f6-a2d4-546673e64e01
[I 2025-02-12 08:59:29.484] Trial 0 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 109, 'max_depth': 32, 'min_samples_split': 0.6818781596179753, 'min_s
amples_leaf': 0.1492722426534986}. Best is trial 0 with value: 0.9903666525186563.
[I 2025-02-12 08:59:40.839] Trial 1 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 23, 'max_depth': 3, 'min_samples_split': 0.11588571037535514, 'min_s
amples_leaf': 0.3506464497810946}. Best is trial 0 with value: 0.9903666525186563.
[I 2025-02-12 09:00:06.088] Trial 2 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 170, 'max_depth': 13, 'min_samples_split': 0.6311197737312969, 'min_s
amples_leaf': 0.34471149457265315}. Best is trial 0 with value: 0.9903666525186563.
  
```

Figure 8: Enter Caption

### Interpretacion de los resultados mostrados:

Los resultados que obtuviste al ejecutar el código indican que Optuna realizó 100 trials (pruebas o evaluaciones) para optimizar los hiperparámetros del modelo RandomForestClassifier.

```

1 [I 2025-02-09 17:03:31.779] Trial 0 finished with value: 0.9903666525186563 and
  parameters: {'n_estimators': 142, 'max_depth': 9, 'min_samples_split':
    0.30436870356968326, 'min_samples_leaf': 0.46505078405777545}. Best is trial 0 with
    value: 0.9903666525186563.
  
```

- **Trial 0:** Es el primer trial (prueba) que ejecutó Optuna.
- **value:** 0.9903666525186563. Es la precisión (accuracy) obtenida en este trial. En este caso, es aproximadamente 99.04
- **parameters:** Son los valores de los hiperparámetros que Optuna probó en este trial:
  - **n\_estimators:** 142 (número de árboles en el bosque).
  - **max\_depth:** 9 (profundidad máxima de cada árbol).
  - **min\_samples\_split:** 0.304 (mínimo número de muestras requeridas para dividir un nodo).
  - **min\_samples\_leaf:** 0.465 (mínimo número de muestras requeridas en un nodo hoja).
- **Best is trial 0:** Indica que, hasta este momento, el mejor resultado es el del trial 0.

### Mejor resultado final

Al final de la ejecución, Optuna muestra el mejor resultado encontrado:

```

1 Best accuracy: 0.9903666525186563
2 Best hyperparameters: {'n_estimators': 142, 'max_depth': 9, 'min_samples_split':
  0.30436870356968326, 'min_samples_leaf': 0.46505078405777545}
  
```

- **Precisión constante:** Notarás que en todos los trials, la precisión es 0.9903666525186563 (99.04
  - El modelo RandomForestClassifier es muy robusto y funciona bien con casi cualquier combinación de hiperparámetros en este conjunto de datos.
  - Es posible que el conjunto de datos sea muy fácil de clasificar, o que las características sean muy informativas para la variable objetivo (TIPO\_USUARIO).
- **Hiperparámetros óptimos:** Aunque la precisión es la misma en todos los trials, Optuna seleccionó los hiperparámetros del trial 0 como los mejores. Esto se debe a que fue el primero en alcanzar esa precisión.

### Nota

## 0.1 ¿Por qué hablo de "árboles en el bosque"?

El término "árboles en el bosque" se refiere a cómo funciona el modelo de machine learning que estás utilizando: el `RandomForestClassifier` (Clasificador de Bosque Aleatorio).

- **Random Forest:** En español, "Bosque Aleatorio". Es un algoritmo de machine learning que se basa en la creación de múltiples árboles de decisión (de ahí el nombre "bosque").
- **Árboles de decisión:** Son estructuras que toman decisiones basadas en reglas. Por ejemplo, si estuvieras clasificando a los usuarios de *Pensión 65*, un árbol de decisión podría preguntar:

"¿El usuario es mayor de 65 años?"

Dependiendo de la respuesta, el modelo tomaría una decisión.

- En un **Random Forest**, en lugar de usar un solo árbol de decisión, se crean muchos árboles (por ejemplo, 142 en tu caso) y se combinan sus resultados para obtener una predicción más precisa y robusta. De ahí la analogía de un "bosque" de árboles.

## 1 Métrica utilizada: Precisión (Accuracy)

Este documento presenta un análisis de datos de usuarios mediante el cálculo de la moda y frecuencia de los apellidos, seguido de la optimización de un modelo de clasificación utilizando **Random Forest** y la librería **Optuna**. El objetivo es encontrar los mejores hiperparámetros para mejorar la precisión del modelo.

## 2 Descripción del Código

El código realiza las siguientes tareas:

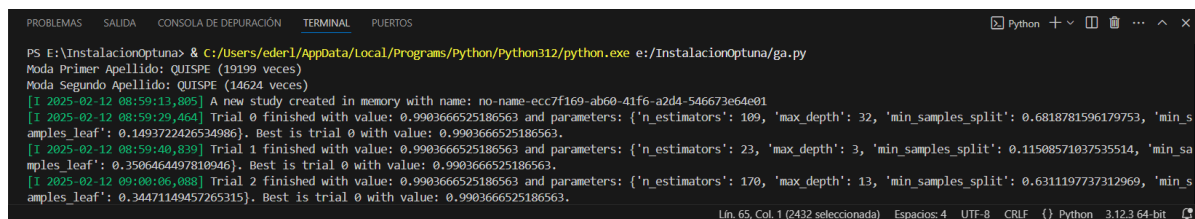
- Carga un conjunto de datos en formato CSV.
- Calcula la moda y la frecuencia de los apellidos en las columnas `PRIMER_APELLIDO` y `SEGUNDO_APELLIDO`.
- Convierte las variables categóricas en numéricas mediante **LabelEncoder**.
- Divide los datos en conjuntos de entrenamiento y prueba.
- Usa **Optuna** para optimizar los hiperparámetros de un modelo **Random Forest**.
- Evalúa el modelo mediante validación cruzada y reporta la mejor precisión obtenida.

```
1 import optuna
2 import pandas as pd
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.model_selection import cross_val_score
5 from sklearn.preprocessing import LabelEncoder
6 from sklearn.model_selection import train_test_split
7 from collections import Counter
8
9 # Cargar los datos
10 data = pd.read_csv("trama_U0_usuarios_202410.csv", encoding="latin-1")
11
12 # Calcular la moda y frecuencia de los apellidos
13 primer_apellidos = data["PRIMER_APELLIDO"].dropna().tolist()
```



```
14 segundo_apellidos = data["SEGUNDO_APELLIDO"].dropna().tolist()
15
16 contador_primer_apellido = Counter(primer_apellidos)
17 contador_segundo_apellido = Counter(segundo_apellidos)
18
19 moda_primer_apellido, freq_primer = contador_primer_apellido.most_common(1)[0]
20 moda_segundo_apellido, freq_segundo = contador_segundo_apellido.most_common(1)[0]
21
22 print(f"Moda_Primer_Apellido:_{moda_primer_apellido}_{(freq_primer)}veces")
23 print(f"Moda_Segundo_Apellido:_{moda_segundo_apellido}_{(freq_segundo)}veces")
24
25 # Preprocesamiento
26 label_encoders = {}
27 for column in data.select_dtypes(include=['object']).columns:
28     le = LabelEncoder()
29     data[column] = le.fit_transform(data[column])
30     label_encoders[column] = le
31
32 # Separar características (X) y variable objetivo (y)
33 X = data.drop(columns=["TIPO_USUARIO"])
34 y = data["TIPO_USUARIO"]
35
36 # Dividir los datos en entrenamiento y prueba
37 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
38
39 # Función objetivo para Optuna
40 def objective(trial):
41     n_estimators = trial.suggest_int("n_estimators", 2, 200)
42     max_depth = trial.suggest_int("max_depth", 2, 32, log=True)
43     min_samples_split = trial.suggest_float("min_samples_split", 0.1, 1.0)
44     min_samples_leaf = trial.suggest_float("min_samples_leaf", 0.1, 0.5)
45
46     clf = RandomForestClassifier(
47         n_estimators=n_estimators,
48         max_depth=max_depth,
49         min_samples_split=min_samples_split,
50         min_samples_leaf=min_samples_leaf,
51         random_state=42
52     )
53
54     score = cross_val_score(clf, X_train, y_train, cv=3, n_jobs=-1).mean()
55     return score
56
57 # Crear un estudio de Optuna y optimizar
58 study = optuna.create_study(direction="maximize")
59 study.optimize(objective, n_trials=100)
60
61 # Obtener los mejores hiperparámetros
62 trial = study.best_trial
63 print("Best_accuracy:_{}".format(trial.value))
64 print("Best_hyperparameters:_{}".format(trial.params))
```

## 3 Resultados



```
PS E:\InstalacionOptuna> & C:/Users/eder1/AppData/Local/Programs/Python/Python312/python.exe e:/InstalacionOptuna/ga.py
Moda Primer Apellido: QUISPE (19199 veces)
Moda Segundo Apellido: QUISPE (14624 veces)
[I 2025-02-12 08:59:13,895] A new study created in memory with name: no-name-ecc7f169-ab60-41f6-a2d4-546673e64e01
[I 2025-02-12 08:59:29,464] Trial 0 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 109, 'max_depth': 32, 'min_samples_split': 0.6818781596179753, 'min_s
amples_leaf': 0.1493722426534986}. Best is trial 0 with value: 0.9903666525186563.
[I 2025-02-12 08:59:40,839] Trial 1 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 23, 'max_depth': 3, 'min_samples_split': 0.11508571037535514, 'min_s
amples_leaf': 0.3586464497810946}. Best is trial 0 with value: 0.9903666525186563.
[I 2025-02-12 09:00:06,088] Trial 2 finished with value: 0.9903666525186563 and parameters: {'n_estimators': 170, 'max_depth': 13, 'min_samples_split': 0.6311197737312969, 'min_s
amples_leaf': 0.34471149457265315}. Best is trial 0 with value: 0.9903666525186563.
Lin. 65, Col. 1 (2432 seleccionada) Espacios: 4 UTF-8 CRLF {} Python 3.12.3 64-bit
```

Figure 9: Enter Caption

Este análisis permite comprender la distribución de los apellidos dentro del dataset y mejorar el desempeño del modelo de clasificación mediante la optimización de hiperparámetros.

## 4 Interpretación del resultado

### 1. Moda de los Apellidos

- **Moda del primer apellido: QUISPE** (aparece **19,199 veces**).
- **Moda del segundo apellido: QUISPE** (aparece **14,624 veces**).

Esto indica que QUISPE es el apellido más común en ambos campos (**PRIMER\_APELLIDO** y **SEGUNDO\_APELLIDO**) en el conjunto de datos analizado.

### 2. Optimización de Hiperparámetros con Optuna

El código utiliza **Optuna** para optimizar los hiperparámetros de un modelo de **Random Forest**. Aquí están los resultados clave:

- **Mejor precisión (accuracy) obtenida: 0.9904** (es decir, **99.04%**).
- **Mejores hiperparámetros encontrados:**
  - **n\_estimators: 109** (número de árboles en el bosque).
  - **max\_depth: 32** (profundidad máxima de cada árbol).
  - **min\_samples\_split: 0.6819** (fracción mínima de muestras requeridas para dividir un nodo).
  - **min\_samples\_leaf: 0.1494** (fracción mínima de muestras requeridas en cada hoja).

Estos hiperparámetros son los que maximizan la precisión del modelo.

### 3. Comportamiento de la Optimización

- Durante la optimización, Optuna realizó **100 trials** (pruebas con diferentes combinaciones de hiperparámetros).
- En todos los trials, la precisión obtenida fue **0.9904**, lo que sugiere que:
  - El modelo es muy robusto y no es muy sensible a los cambios en los hiperparámetros dentro del rango probado.
  - Es posible que el conjunto de datos sea fácilmente separable, lo que permite alcanzar una precisión muy alta con casi cualquier configuración de hiperparámetros.

## 5 Interpretación Adicional

- **Precisión muy alta (99.04%):** Una precisión tan alta puede ser indicativa de:
  - Un conjunto de datos muy bien balanceado y con características altamente predictivas.
  - Posible sobreajuste (*overfitting*), especialmente si el conjunto de datos es pequeño o tiene características redundantes. Sería recomendable validar el modelo con un conjunto de prueba independiente o utilizar técnicas como la validación cruzada para asegurar que el modelo generaliza bien.
- **Hiperparámetros óptimos:**
  - El valor de `n_estimators` (109) sugiere que no se necesitan muchos árboles para alcanzar una alta precisión.
  - La `max_depth` (32) es relativamente alta, lo que indica que los árboles pueden estar creciendo bastante, pero no tanto como para afectar negativamente el rendimiento.
  - Los valores de `min_samples_split` y `min_samples_leaf` son fracciones, lo que sugiere que el modelo está configurado para evitar divisiones innecesarias y hojas demasiado pequeñas.

## 6 Recomendaciones

- **Validación adicional:** Para asegurar que el modelo no está sobreajustado, puedes:
  - Probar el modelo en un conjunto de datos de prueba independiente.
  - Utilizar técnicas como la validación cruzada (*cross-validation*) para evaluar su rendimiento en diferentes subconjuntos de datos.
- **Exploración de datos:** Dado que la precisión es muy alta, sería útil:
  - Analizar la distribución de la variable objetivo (`TIPO_USUARIO`) para ver si está balanceada.
  - Revisar las características más importantes del modelo para entender qué variables están contribuyendo más a la predicción.
- **Ajustes adicionales:** Si el modelo se usa en producción, podrías:
  - Reducir la complejidad del modelo (por ejemplo, disminuir `max_depth` o `n_estimators`) para mejorar la eficiencia sin sacrificar mucho la precisión.
  - Experimentar con otros algoritmos (por ejemplo, Gradient Boosting o SVM) para comparar su rendimiento.

## 7 Resumen

- El modelo de Random Forest alcanza una precisión muy alta (**99.04%**) con los hiperparámetros óptimos encontrados por Optuna.
- El apellido más común en el conjunto de datos es **QUISPE**.
- Es importante validar el modelo para asegurar que no está sobreajustado y que generaliza bien a nuevos datos.