Manual de Optimización de Hiperparámetros con Optuna

Fernando jose Mamani Machaca
Eddy Kennedy Mamani Hallasi

10 de febrero de 2025

Índice

Haz clic aquí para acceder al notebook

Original file is located at - COLAB

1. Introducción

Este manual documenta la implementación de un proceso de optimización de hiperparámetros utilizando Optuna para modelos de clasificación. El código está diseñado para trabajar con el conjunto de datos de cáncer de mama de scikit-learn y optimiza dos clasificadores diferentes: Random Forest y SVC.

2. Requisitos Previos

Para ejecutar el código es necesario instalar las siguientes bibliotecas:

```
!pip install --quiet optuna
import optuna
import sklearn.datasets
import sklearn.ensemble
import sklearn.model_selection
import sklearn.svm
import optuna.visualization
import random
import numpy as np
```

3. Implementación del Código

3.1. Configuración Inicial

```
random.seed(42)
np.random.seed(42)
breast_cancer = sklearn.datasets.load_breast_cancer()
```

3.2. Función Objetivo

```
def objective(trial):
    classifier = trial.suggest_categorical("classifier",
                                           ["RandomForest", "SVC
    "])

    if classifier == "RandomForest":
        n_estimators = trial.suggest_int("n_estimators", 2,
    20)
        max_depth = int(trial.suggest_float("max_depth",
                                            1, 32, log=True))
        clf = sklearn.ensemble.RandomForestClassifier(
            n_estimators=n_estimators,
            max_depth=max_depth
        )
    else:
        c = trial.suggest_float("svc_c", 1e-10, 1e10, log=
    True)
        clf = sklearn.svm.SVC(C=c, gamma="auto")

    return sklearn.model_selection.cross_val_score(
        clf,
        breast_cancer.data,
        breast_cancer.target,
        n_jobs=-1,
        cv=3
    ).mean()
```

3.3. Ejecución del Estudio de Optuna

A continuación, se muestra cómo crear y ejecutar el estudio de Optuna para encontrar los mejores hiperparámetros:

```
study = optuna.create_study(direction="maximize")
study.optimize(objective, n_trials=50)

print("Mejores hiperpar metros encontrados:")
print(study.best_params)
print(f"Precisi n alcanzada: {study.best_value:.4f}")
```

3.4. Explicación Técnica

Optuna utiliza un enfoque de optimización bayesiana para explorar el espacio de hiperparámetros. En este caso:

- **Para Random Forest**:

- Se utiliza una distribución **logarítmica** para el hiperparámetro `max_depth`. Esto permite explorar valores en un rango amplio (de 1 a 32) de manera eficiente, priorizando valores más pequeños al principio y aumentando gradualmente la exploración hacia valores más grandes.
- La distribución logarítmica es útil porque la profundidad de un árbol (`max_depth`) tiene un impacto no lineal en el rendimiento del modelo. Valores pequeños pueden subajustar (underfitting), mientras que valores grandes pueden sobreajustar (overfitting).

- **Para SVC (Support Vector Classifier)**:
  - El parámetro `C` (penalización de errores) se explora en una escala **logarítmica**, con un rango de 10^{-10} a 10^{10}. Esto se debe a que `C` puede variar en órdenes de magnitud, y una escala lineal no sería eficiente para explorar este rango tan amplio.
  - La escala logarítmica permite explorar valores muy pequeños (para modelos más simples) y valores muy grandes (para modelos más complejos) de manera equilibrada.

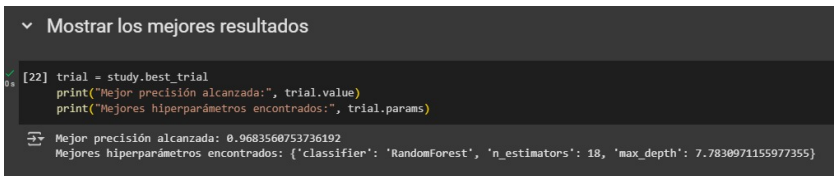
Beneficios de este enfoque:

- La optimización bayesiana es eficiente porque construye un modelo probabilístico (surrogate model) del espacio de búsqueda y lo utiliza para guiar la exploración hacia regiones prometedoras.
- Las distribuciones logarítmicas son ideales para hiperparámetros que tienen un impacto no lineal en el rendimiento del modelo, como `max_depth` y `C`.

3.5. Ejemplo de Salida

A continuación, se muestra un ejemplo de la salida generada por Optuna después de ejecutar el estudio:

```
trial = study.best_trial
print("Mejor precisi n alcanzada:", trial.value)
print("Mejores hiperpar metros encontrados:", trial.params)
```



```
Mejores hiperpar metros encontrados:
{'classifier': 'RandomForest', 'n_estimators': 18, 'max_depth
    ': 7.7830}
Precisi n alcanzada: 0.9684
```

Los resultados muestran un rendimiento destacado del modelo RandomForest con una precisión del 96.84 % en la clasificación del conjunto de datos de cáncer de mama. La optimización seleccionó una configuración eficiente con 18 árboles y una profundidad máxima de aproximadamente 8 niveles, lo que sugiere un buen balance entre complejidad y capacidad predictiva. Esta configuración demuestra que el modelo puede alcanzar una alta precisión mientras mantiene una estructura relativamente simple y computacionalmente eficiente.

4. Visualizaciones

4.1. Gráfico de Historia de Optimización

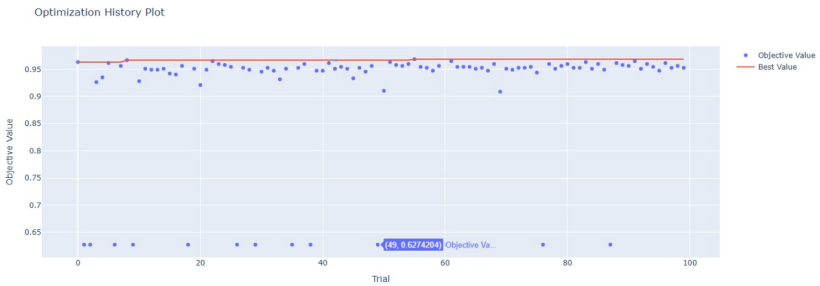


Figura 1: Historia de optimización mostrando la evolución del valor objetivo a lo largo de las iteraciones

4.2. Gráfico de Corte (Slice Plot)

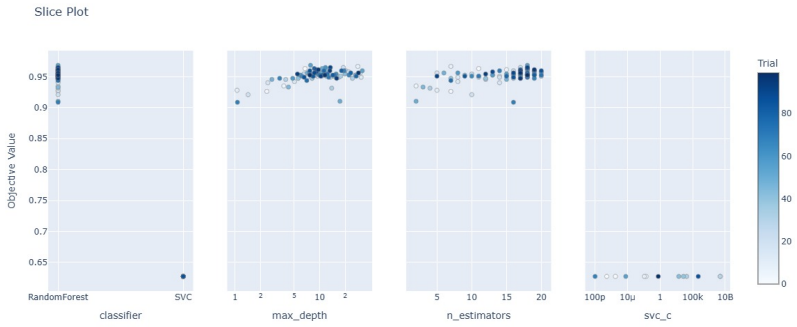


Figura 2: Visualización de la importancia de cada hiperparámetro

4.3. Gráfico de Contorno

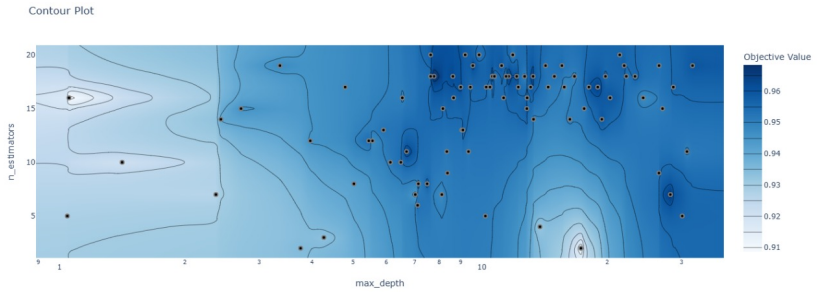


Figura 3: Visualización de la relación entre n_estimators y max_depth

5. Conclusiones

La optimización de hiperparámetros utilizando Optuna ha permitido encontrar la mejor configuración para el modelo de clasificación. Las visualizaciones proporcionan información valiosa sobre el proceso de optimización y la importancia relativa de cada hiperparámetro.

6. Referencias

- Documentación oficial de Optuna: `https://optuna.readthedocs.io/`
- Documentación oficial de scikit-learn: `https://scikit-learn.org/`

- Guía de uso de Random Forest en scikit-learn: `https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html`
- Guía de uso de SVC en scikit-learn: `https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html`