Clase 087 — SHAP en profundidad: TreeExplainer, KernelExplainer, DeepExplainer

Parte: 1 — Machine Learning Clásico · Fuente: Lundberg & Lee (2017) + shap docs.
Duración estimada: 90 min.

Clase 087 — SHAP en profundidad: TreeExplainer, KernelExplainer, DeepExplainer

> *Parte: 1 — Machine Learning Clásico · Fuente: Lundberg & Lee (2017) + shap docs. Duración estimada: 90 min.*

Objetivo

Dominar SHAP (SHapley Additive exPlanations) en profundidad: teoría de Shapley values (teoría de juegos cooperativos), TreeExplainer (rápido y exacto para árboles), KernelExplainer (model-agnostic, lento), DeepExplainer (para NN), y los plots clave: summary_plot, waterfall_plot, force_plot, dependence_plot, decision_plot.

Resultados de aprendizaje

Al finalizar, el estudiante podrá:

- Explicar Shapley value intuitivamente: contribución marginal promedio sobre todos los órdenes de inclusión.
- Aplicar TreeExplainer en XGBoost/LightGBM/RF (segundos para millones de samples).
- Generar e interpretar los 5 plots SHAP principales.
- Diferenciar explicación global (summary_plot beeswarm) de local (waterfall de una predicción).
- Reconocer las limitaciones: SHAP asume cierta forma de "feature attribution" pero no es causal.

Temas

- Shapley value: φ_i = Σ |S|!(M-|S|-1)! / M! · [v(S {i}) - v(S)].
- 4 propiedades únicas: efficiency, symmetry, dummy, additivity.
- TreeExplainer: exacto para tree-based, O(TLD²).
- KernelExplainer: LIME-style con kernel especial → SHAP values aproximados.
- DeepExplainer: para Keras/PyTorch.
- Permutation explainer: alternativa moderna sin necesidad de árbol/red.

Definiciones y características

- Shapley value: única atribución que satisface las 4 propiedades.
- shap_values: matriz (n_samples, n_features) con contribución de cada feature a cada predicción.
- expected_value: predicción promedio del modelo (baseline). Σ shap_values[i] + expected_value ≈ predicción[i].
- summary_plot beeswarm: para cada feature, distribución de SHAP values; color = valor de la feature.
- waterfall_plot: para una predicción específica, contribución acumulada feature por feature.
- dependence_plot: feature en x, SHAP value en y; revela no-linearidades e interacciones.

Dataset / recursos

- fetch_california_housing o load_breast_cancer.
- Librerías: shap (pip install shap), xgboost, matplotlib.

Ejercicios

1. TreeExplainer: XGBoost en California Housing → explainer = shap.TreeExplainer(model); shap_values = explainer(X_test).
2. Summary plot: shap.summary_plot(shap_values, X_test). Identificar las 3 features más importantes y su dirección.
3. Waterfall: elegir 1 muestra concreta → shap.waterfall_plot(shap_values[0]). Sumar contribuciones y verificar que reconstruye la predicción.
4. Dependence plot: shap.dependence_plot('MedInc', shap_values.values, X_test). Detectar non-linearity.
5. Interaction values: shap_interaction = explainer.shap_interaction_values(X_test). Identificar par de features con mayor interacción.

Homework verificable

XGBoost sobre California Housing + análisis SHAP completo:

1. Modelo entrenado.
2. SHAP values con TreeExplainer.
3. Summary plot + force_plot de 3 predicciones (alta, media, baja).
4. Dependence plots para top-3 features.
5. Reporte de 1 página: insights de qué mueve el precio (en lenguaje no técnico).

Criterio de aceptación: el reporte identifica correctamente las features dominantes; las explicaciones individuales suman al valor del modelo (±0.01).

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
KernelExplainer sobre tree → muy lento	Usa el equivocado. Fix: TreeExplainer.
Asumir SHAP es causal	No lo es — atribución es estadística, no c
Interpretar SHAP sobre train data	Sesgo por overfit. Fix: explicar sobre val
force_plot no renderiza en Jupyter	Sin shap.initjs(). Fix: ejecutar al inicio
Features muy correlacionadas → SHAP distri	Limitación inherente. Fix: agrupar correla

Preguntas frecuentes

SHAP vs LIME?

SHAP: fundamento teórico, exacto en tree-based, determinista. LIME: model-agnostic, rápido pero inestable. SHAP gana hoy.

¿explainer(X) o explainer.shap_values(X)?

API moderna (≥ 0.40): explainer(X) devuelve Explanation object. Compatible con todos los plots. Recomendado.

¿SHAP para clasificación multiclase?

shap_values es lista/tensor con valores por clase. Plot por clase con shap.summary_plot(shap_values[class_idx]).

¿SHAP en LLM?

Existe pero costoso por la combinatoria. Para LLMs se usan otras técnicas (attention rollout, integrated gradients).

¿En producción reporto SHAP por predicción?

Sí — para decisiones high-stake (crédito, medicina), el SHAP por predicción ayuda al human-in-the-loop a entender qué pasó.

Referencias

- Lundberg & Lee (2017), A Unified Approach to Interpreting Model Predictions, NeurIPS.
- Lundberg et al. (2020), From local explanations to global understanding with explainable AI for trees, Nature Machine Intelligence.
- SHAP docs.
- Molnar, C. Interpretable Machine Learning — cap. SHAP.

Siguiente clase

Clase 088 — Boosting: AdaBoost y Gradient Boosting

Apéndice: notebook (primer bloque)

TreeExplainer sobre GBM (rápido y exacto) + DeepExplainer/KernelExplainer sobre MLP. Instalar: pip install shap.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.datasets import make_classification
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split

try:
    import shap
    SHAP_OK = True
except ImportError:
    print('instalar: pip install shap')
    SHAP_OK = False

np.random.seed(42)
```

Archivos complementarios

- notebook.ipynb