Clase 083 — Voting classifiers: hard y soft

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 7. Duración estimada: 45 min.

Clase 083 — Voting classifiers: hard y soft

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 7. Duración estimada: 45 min.

Objetivo

Combinar varios modelos heterogéneos en un ensemble por votación y entender cuándo conviene hard voting (voto por mayoría) versus soft voting (promedio de probabilidades), apoyándonos en el principio de wisdom of the crowd.

Resultados de aprendizaje

Al finalizar la clase, el estudiante podrá:

- Explicar por qué un ensemble de clasificadores diversos suele superar al mejor modelo individual.
- Implementar un VotingClassifier de scikit-learn con voting="hard" y voting="soft".
- Decidir entre hard y soft voting según si los modelos base estiman probabilidades calibradas.
- Comparar la accuracy del ensemble contra la de cada modelo base en un dataset de clasificación.
- Identificar errores frecuentes al armar el ensemble (modelos correlacionados, probabilidades mal calibradas, pesos sin justificar).

Temas

- Wisdom of the crowd: por qué combinar predicciones reduce error.
- Hard voting: predicción final = clase con más votos entre los modelos base.
- Soft voting: predicción final = clase con mayor promedio de probabilidades estimadas.
- Requisitos para soft voting: que cada modelo exponga predict_proba y esté bien calibrado.
- Diversidad entre modelos base (algoritmos distintos, no sólo hiperparámetros distintos).
- VotingClassifier(estimators=[...], voting=..., weights=...).
- Limitaciones: si los modelos se equivocan en los mismos ejemplos, el ensemble no ayuda.

Definiciones y características

- Hard voting: cada clasificador del ensemble emite una predicción discreta y se elige la clase con más votos (moda). No requiere predict_proba.
- Soft voting: se promedian las probabilidades estimadas por cada modelo (opcionalmente ponderadas) y se elige la clase con mayor probabilidad promedio. Suele rendir mejor que hard voting cuando las probabilidades son confiables.
- VotingClassifier (sklearn.ensemble): meta-estimador que envuelve una lista de (nombre, estimador) y expone fit, predict, predict_proba (sólo si voting="soft").
- Weak learner / strong learner: un weak learner apenas supera el azar; al combinar muchos weak learners diversos se obtiene un strong learner. Ley de los grandes números aplicada a clasificadores independientes.
- Wisdom of the crowd: si los errores de los modelos son independientes, la probabilidad de que la mayoría se equivoque cae exponencialmente con el número de modelos.
- Diversidad: condición clave para que el ensemble funcione. Se logra usando algoritmos distintos

(logística, SVM, árbol, kNN, etc.), no copias del mismo modelo con seeds distintos.
- Calibración: un modelo está calibrado si predict_proba ≈ frecuencia empírica. SVM con probability=True y árboles puros suelen estar mal calibrados, lo que degrada el soft voting.

Dataset / recursos

- Dataset sugerido: make_moons(n_samples=500, noise=0.30, random_state=42) (mismo que usa Géron en el cap. 7).
- Alternativa: load_breast_cancer() de sklearn.datasets para un caso real.
- Imports clave: LogisticRegression, RandomForestClassifier, SVC, VotingClassifier, train_test_split, accuracy_score.

Ejercicios

1. Cargar make_moons y partir en train/test (80/20, random_state=42). Entrenar por separado LogisticRegression, RandomForestClassifier y SVC(probability=True). Reportar accuracy de cada uno.
2. Armar un VotingClassifier con esos tres modelos y voting="hard". Comparar accuracy contra cada modelo base.
3. Repetir con voting="soft" (recordá SVC(probability=True)). ¿Mejora? ¿Por qué?
4. Probar weights=[1, 2, 1] favoreciendo al Random Forest. ¿Cómo cambia la performance? Justificar.
5. Reemplazar uno de los modelos por una copia casi idéntica de otro (ej.: dos regresiones logísticas con C muy parecido). Observar y explicar por qué el ensemble deja de ganar.

Homework verificable

Entregar un script voting.py que:

1. Cargue make_moons(n_samples=500, noise=0.30, random_state=42).
2. Entrene tres modelos base distintos y un VotingClassifier en modo hard y otro en modo soft.
3. Imprima la accuracy en test de los cinco (3 base + 2 ensembles).

Criterio de aceptación: el VotingClassifier en modo soft debe alcanzar accuracy ≥ que el mejor modelo base individual sobre el test split. Si no lo logra, justificar en un comentario qué modelo está rompiendo la diversidad o la calibración.

Errores comunes

- Usar voting="soft" con modelos no calibrados (ej.: SVC sin probability=True, o árboles muy profundos). Las probabilidades estimadas son ruido y el promedio empeora la predicción.
- Olvidar probability=True en SVC: sin eso, SVC no expone predict_proba y voting="soft" falla con AttributeError.
- Modelos base correlacionados (tres árboles, o tres regresiones logísticas con hiperparámetros parecidos): no hay diversidad, el ensemble no aporta nada.
- Pesos arbitrarios en weights= sin validación cruzada que los respalde. Mejor empezar con pesos iguales y ajustar con evidencia.
- No fijar random_state en los modelos estocásticos: los resultados no son reproducibles y dificultan la comparación.

Preguntas frecuentes

- ¿Soft voting es siempre mejor que hard? No. Es mejor si los modelos base están bien calibrados. Con modelos mal calibrados, hard voting puede ganarle.
- ¿Cuántos modelos conviene meter? Géron muestra que con 3-5 modelos diversos ya hay ganancia notoria. Más allá, los rendimientos decrecen.
- ¿Puedo mezclar modelos lineales y no lineales? Sí, y es lo recomendado: la diversidad de sesgos inductivos es justo lo que hace funcionar al ensemble.
- ¿Diferencia con bagging? Bagging usa el mismo algoritmo entrenado en distintos bootstraps. Voting usa algoritmos distintos sobre el mismo dataset. Se cubre en la clase 075.
- ¿Y si un modelo base es mucho peor que los demás? Suele convenir sacarlo o bajarle el peso; arrastra al ensemble, sobre todo en hard voting.

Referencias

- Géron, A. Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow (3ª ed.), cap. 7 — sección "Voting Classifiers".
- scikit-learn — VotingClassifier.
- scikit-learn — Ensemble methods: voting classifier.

Siguiente clase

Clase 084 — Bagging y pasting

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb