Clase 067 — Clasificación multiclase, multilabel, multioutput

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 3. Duración estimada: 60 min.

Clase 067 — Clasificación multiclase, multilabel, multioutput

> *Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 3. Duración estimada: 60 min.*

Objetivo

Que el alumno distinga los tres escenarios de clasificación más allá del binario — multiclase (una salida con K>2 clases), multilabel (varias etiquetas por muestra) y multioutput (varias salidas, cada una con su propio rango de valores) — y sepa qué estrategia de sklearn (OneVsRest, OneVsOne, MultiOutputClassifier) usar en cada caso, eligiendo además la métrica correcta (accuracy global, hamming loss, macro/micro F1).

Resultados de aprendizaje

Al finalizar la clase, el alumno podrá:

1. Diferenciar multiclase, multilabel y multioutput con un ejemplo concreto de cada uno.
2. Elegir entre OvR y OvO según el costo computacional del clasificador base y el tamaño del dataset.
3. Entrenar un clasificador multilabel con KNeighborsClassifier y evaluarlo con f1_score(average='macro') y hamming_loss.
4. Envolver un clasificador binario en MultiOutputClassifier para resolver un problema multioutput.
5. Interpretar las salidas de predict_proba en cada escenario (lista de arrays vs array 2D).

Temas

#	Tema	Por qué importa
1	Multiclase: definición y clasificadores na	Decision Tree, Random Forest, Naive Bayes,
2	OneVsRestClassifier (OvR)	K modelos binarios — escala lineal en K, d
3	OneVsOneClassifier (OvO)	K·(K-1)/2 modelos — caro en clases, barato
4	Multilabel: y es matriz binaria K-dim	Tags de noticias, géneros de película, mul
5	Métricas multilabel: hamming loss, macro/m	Accuracy global castiga demasiado; necesit
6	Multioutput: MultiOutputClassifier / Multi	Cada salida es independiente, con sus prop

Definiciones y características

Clasificación multiclase

: Una sola etiqueta por muestra pero con K>2 valores posibles (ej.: dígitos 0–9). y es vector 1D de enteros. Algunos clasificadores la soportan nativamente (RandomForest, GaussianNB, SGDClassifier); otros (SVM, Logistic puro) son binarios y sklearn los envuelve automáticamente con OvR u OvO.

One-vs-Rest (OvR / OvA)

: Estrategia que entrena K clasificadores binarios, cada uno "esta clase vs todas las demás". Predice la clase del clasificador con mayor score. Default para la mayoría — lineal en K, ideal cuando entrenar es barato pero el clasificador base no escala con N.

One-vs-One (OvO)

: Entrena K·(K-1)/2 clasificadores binarios, uno por cada par de clases. Cada uno ve solo las muestras de sus dos clases (subsets chicos). Conviene cuando el clasificador base escala mal con N (ej.: SVM kernelizado, $O(N^2)$); sklearn lo usa por default para SVC.

Clasificación multilabel

: Cada muestra puede pertenecer a varias clases simultáneamente. y es matriz binaria (n_samples, n_labels) con 0/1 por etiqueta. Ejemplo: una foto puede tener ['perro', 'aire libre', 'soleado'] a la vez. KNeighborsClassifier, RandomForestClassifier y DecisionTreeClassifier lo soportan nativamente.

Clasificación multioutput

: Generalización: cada muestra tiene varias salidas y cada salida es a su vez multiclase (no binaria como en multilabel). Ejemplo típico de Géron: denoising de imagen — cada píxel es una "salida" con 256 valores posibles. Se resuelve con MultiOutputClassifier(base_estimator).

Hamming loss

: Fracción promedio de etiquetas mal predichas sobre el total (n_samples × n_labels). Métrica natural para multilabel — un error en una de 5 etiquetas pesa 0.2, no 1.0 como subset accuracy. Cuanto más bajo, mejor.

Macro vs micro averaging

: Macro = promedio simple de la métrica calculada por clase (cada clase pesa igual; bueno con desbalance si querés tratar todas las clases por igual). Micro = suma global de TP/FP/FN y luego cálculo (cada muestra pesa igual; bueno cuando importa el volumen total).

MultiOutputClassifier

: Wrapper de sklearn que entrena un clasificador independiente por columna de salida. No modela correlaciones entre salidas (para eso existe ClassifierChain). Trivial: MultiOutputClassifier(RandomForestClassifier()).fit(X, Y).

Dataset / recursos

MNIST (multiclase 10 clases) vía fetch_openml('mnist_784', as_frame=False). Para multilabel/multioutput se construye Y sintético sobre MNIST: etiquetas [es_grande (>=7), es_impar] → multilabel; o X_noisy → X_clean → multioutput.

Ejercicios

1. Multiclase nativo vs OvR. Entrená SGDClassifier en MNIST (10 clases) y comparalo con OneVsRestClassifier(SGDClassifier()). ¿Cuántos clasificadores entrena cada uno? Mirá .estimators_.

2. OvO con SVM. Entrená SVC() sobre un subset de 5k muestras de MNIST. Verificá que clf.decision_function(X[:1]) devuelve 45 scores = 10·9/2. Forzá luego OneVsRestClassifier(SVC()) y compará tiempos.

3. Multilabel con KNN. Construí Y_multilabel = np.c_[y >= 7, y % 2 == 1]. Entrená KNeighborsClassifier() con ese Y. Reportá f1_score(Y_test, Y_pred, average='macro') y hamming_loss(Y_test, Y_pred).

4. Macro vs micro F1. Con el modelo del ejercicio 3, calculá F1 con average='macro' y average='micro'. Si una de las etiquetas estuviera muy desbalanceada (ej.: solo 5% de positivos), ¿cuál cambiaría más y por qué?

5. Multioutput denoising. Generá X_train_noisy = X_train + np.random.randint(0, 100, X_train.shape). Entrená

KNeighborsClassifier() con X_noisy como features y X_clean como target (cada píxel es una salida multiclase 0–255). Predecí y visualizá una imagen denoised.

Homework verificable

Notebook sobre MNIST con: (a) clasificador multiclase con RandomForestClassifier y matriz de confusión 10×10; (b) versión multilabel con etiquetas [es_grande, es_impar] usando KNeighborsClassifier y reporte de hamming_loss + f1_score(average='macro'); (c) ejemplo multioutput de denoising sobre 100 imágenes con ruido uniforme, mostrando 3 pares (ruidosa, denoised, original).

Criterio de aceptación: F1 macro multilabel > 0.95 sobre test. Hamming loss < 0.05. Denoising visualmente reconocible (no hace falta métrica numérica, sí imagen comparativa lado a lado).

Errores comunes

Síntoma / mensaje	Causa y cómo arreglar
ValueError: y should be a 1d array, got ..	El clasificador no soporta multilabel nati
Accuracy multilabel sale carísima (1.0 es	accuracy_score en multilabel exige que tod
SVC tarda eternidades en MNIST completo	SVC con kernel es O(N²)–O(N³). Sklearn ya
predict_proba de un multilabel devuelve li	Es lo esperado: una matriz (n_samples, n_c
Macro F1 da muy distinto al micro F1	Indica desbalance fuerte entre clases/etiq

Preguntas frecuentes

¿Cuándo OvR y cuándo OvO?

Default: OvR (lineal en K, simple). Usá OvO solo si el clasificador base escala mal con N (SVM kernelizado es el caso típico) y K es chico. Sklearn ya elige bien por default — rara vez hace falta override manual.

Multilabel vs multioutput, ¿en qué se diferencian exactamente?

Multilabel = caso particular de multioutput donde cada salida es binaria (0/1 = "tiene/no tiene esa etiqueta"). Multioutput general = cada salida puede ser multiclase (ej.: 256 niveles de gris por píxel). Sklearn los trata casi igual; la diferencia es solo el rango de valores que puede tomar cada columna de Y.

¿MultiOutputClassifier modela correlaciones entre las salidas?

No — entrena un modelo independiente por columna. Si las etiquetas están correlacionadas (ej.: "es perro" y "tiene cola"), ClassifierChain aprende esa estructura: pasa la predicción de la primera etiqueta como feature extra al modelo de la siguiente.

¿Por qué cross_val_score con SGDClassifier en MNIST da accuracy alta pero la matriz de confusión muestra confusión entre 3 y 5?

Accuracy global enmascara errores localizados entre pocas clases. La matriz de confusión normalizada por filas es indispensable en multiclase — la clase 060 (la próxima) se dedica exactamente a eso.

¿Puedo combinar class_weight / SMOTE con multilabel?

class_weight='balanced' sí, etiqueta por etiqueta. SMOTE multilabel es más delicado (existe MLSMOTE en imbalanced-learn, no en sklearn core). Para el curso: usá class_weight y métricas macro.

Referencias

- Géron, cap. 3 § Multiclass / Multilabel / Multioutput Classification.
- sklearn — Multiclass and multioutput algorithms
- sklearn — MultiOutputClassifier
- sklearn — Métricas de clasificación (hamming_loss, f1_score)

Siguiente clase

Clase 068 — Análisis de errores

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb