Clase 077 — SVM lineal

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 5. Duración estimada: 60 min.

Clase 077 — SVM lineal

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 5. Duración estimada: 60 min.

Objetivo

Entender el principio de maximización del margen que define a las Support Vector Machines lineales, distinguir entre hard margin y soft margin, y entrenar un clasificador con LinearSVC controlando el trade-off bias/varianza mediante el hiperparámetro C.

Resultados de aprendizaje

Al finalizar la clase, podrás:

- Explicar qué es el margen y por qué SVM busca el hiperplano que lo maximiza.
- Diferenciar hard margin (datos linealmente separables, sin tolerancia) de soft margin (admite violaciones).
- Interpretar el hiperparámetro C y su efecto sobre el ancho del margen y las violaciones permitidas.
- Entrenar un LinearSVC en scikit-learn con un Pipeline que incluya StandardScaler.
- Identificar los vectores soporte y entender por qué son los únicos puntos que definen la frontera.

Temas

- Intuición geométrica: hiperplano separador y margen.
- Hard margin: condiciones y limitaciones (sensibilidad a outliers, exige separabilidad).
- Soft margin: introducción de variables de holgura (slack).
- Hiperparámetro C: regularización, trade-off margen ancho vs. violaciones.
- Función de pérdida hinge loss.
- API de scikit-learn: LinearSVC, loss, C, dual, max_iter.
- Importancia crítica del escalado de features en SVM.

Definiciones y características

- Margen (margin): distancia perpendicular entre el hiperplano separador y los puntos más cercanos de cada clase. SVM busca maximizarlo.
- Vectores soporte (support vectors): instancias que tocan o violan el margen. Son los únicos puntos que determinan la frontera; el resto del dataset no influye.
- Hard margin classification: exige que todas las instancias estén del lado correcto del margen. Solo funciona si los datos son linealmente separables y es muy sensible a outliers.
- Soft margin classification: permite violaciones del margen (instancias dentro del margen o del lado equivocado) a cambio de un margen más ancho y robusto.
- Hiperparámetro C: controla cuánto se penalizan las violaciones del margen. C alto → poco tolerante, margen estrecho, riesgo de overfitting. C bajo → más tolerante, margen ancho, más regularización.
- LinearSVC: implementación eficiente de SVM lineal en scikit-learn, basada en liblinear. Escala mejor que SVC(kernel="linear") en datasets grandes.
- Hinge loss: función de pérdida que SVM minimiza, definida como max(0, 1 - t·y) donde t es la etiqueta

(±1) e y la salida del modelo. Es cero cuando la predicción está del lado correcto y fuera del margen.
- Escalado: SVM es sensible a la escala de los features; sin StandardScaler el margen queda dominado por las variables de mayor rango.

Dataset / recursos

- Iris (sklearn.datasets.load_iris) — filtrado a dos clases (Iris-Virginica vs. resto) usando los features petal length y petal width. Es el ejemplo canónico del capítulo 5 de Géron.
- Opcional: dataset sintético con make_classification o make_blobs para visualizar el efecto de C y de outliers.

Ejercicios

1. Cargá Iris, quedate con dos features (petal length, petal width) y la clase Virginica como problema binario. Entrená un Pipeline([StandardScaler, LinearSVC(C=1, loss="hinge")]) y reportá accuracy sobre un split train/test.
2. Repetí el entrenamiento con C=0.1, C=1, C=100. Compará accuracy, número de vectores soporte estimados y ancho del margen. ¿Qué pasa en los extremos?
3. Graficá la frontera de decisión y el margen para los tres valores de C del ejercicio anterior (scatter de los datos + línea + márgenes punteados).
4. Agregá un outlier artificial a la clase minoritaria y volvé a entrenar con C alto y C bajo. Mostrá cómo C bajo absorbe mejor el outlier.
5. Compará tiempo de entrenamiento de LinearSVC vs. SVC(kernel="linear") sobre un dataset de ~10.000 muestras (make_classification). Confirmá que LinearSVC es más rápido.

Homework verificable

Entregá un script tarea_068.py que:

1. Cargue Iris binarizado (Virginica vs. resto) con petal length y petal width.
2. Construya un Pipeline con StandardScaler + LinearSVC(C=1, loss="hinge", random_state=42).
3. Haga train_test_split(test_size=0.2, random_state=42, stratify=y) y entrene.
4. Imprima accuracy sobre test y los coeficientes del clasificador (coef_, intercept_).

Criterio de aceptación: accuracy >= 0.93 sobre el test set y los coeficientes deben provenir del modelo escalado (no del crudo).

Errores comunes

1. No escalar features. SVM es sensible a la escala; sin StandardScaler el margen lo domina la variable de mayor rango y el modelo rinde mal.
2. Confundir la dirección de C. C alto = menos regularización (margen estrecho, más overfitting). C bajo = más regularización. Es inversa a alpha de Ridge/Lasso.
3. Usar SVC(kernel="linear") en datasets grandes. Es $O(m^2)$ a $O(m^3)$; preferí LinearSVC (basado en liblinear) que escala mejor.
4. Esperar predict_proba de LinearSVC. No lo soporta directamente. Usá CalibratedClassifierCV o decision_function si necesitás scores.
5. Olvidar loss="hinge". El default de LinearSVC es "squared_hinge", que no es la pérdida SVM canónica del libro.

Preguntas frecuentes

1. ¿C alto o bajo? Depende del problema. Empezá con C=1 y ajustá con GridSearchCV. Si hay overfitting, bajá C; si hay underfitting o el modelo es demasiado tolerante, subí C.
2. ¿Por qué se llaman "vectores soporte"? Porque son los únicos puntos que "sostienen" la frontera: si los movés, el hiperplano cambia. Los demás puntos no afectan al modelo.
3. ¿SVM lineal sirve si los datos no son linealmente separables? Con soft margin sí, tolera violaciones. Si la separación claramente no es lineal, pasá a kernels (clase 069).
4. ¿LinearSVC vs. LogisticRegression? Ambos producen fronteras lineales. SVM optimiza margen (hinge loss), LogReg optimiza log-likelihood. En la práctica suelen dar resultados similares; SVM tiende a ser más robusto a outliers cuando C es moderado.
5. ¿Funciona para multiclase? Sí, vía one-vs-rest automáticamente en LinearSVC.

Referencias

- Géron, A. Hands-On Machine Learning, 3ra ed., cap. 5 — "Support Vector Machines", sección "Linear SVM Classification".
- scikit-learn user guide: 1.4. Support Vector Machines.
- API: sklearn.svm.LinearSVC.

Siguiente clase

Clase 078 — SVM no lineal: kernel polinomial y RBF

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb