Clase 079 — SVM para regresión (SVR)

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 5. Duración estimada: 50 min.

Clase 079 — SVM para regresión (SVR)

Parte: 1 — Machine Learning Clásico · Fuente: Géron, cap. 5. Duración estimada: 50 min.

Objetivo

Aplicar Support Vector Machines al problema de regresión: entender el truco del epsilon-insensitive loss (ajustar dentro de un tubo de tolerancia en vez de minimizar el error puntual), entrenar LinearSVR y SVR con kernel, y elegir cuándo conviene SVR sobre regresión lineal clásica.

Resultados de aprendizaje

Al finalizar la clase, vas a poder:

- Explicar la lógica de SVR: maximizar el ancho del tubo ε mientras se contienen la mayoría de los puntos dentro.
- Entrenar un LinearSVR y un SVR(kernel="rbf") de scikit-learn con sus hiperparámetros (epsilon, C, gamma).
- Interpretar el efecto de epsilon (ancho del tubo) y C (penalización por puntos fuera del tubo) en el bias-variance trade-off.
- Comparar SVR vs LinearRegression / Ridge en un dataset con outliers.
- Justificar cuándo SVR escala mal (SVR es O(m²–m³)) y conviene LinearSVR u otro modelo.

Temas

1. De SVM clasificación a SVM regresión: invertir el objetivo.
2. Epsilon-insensitive loss: errores menores a ε no se penalizan.
3. LinearSVR: caso lineal, escala bien (O(m)).
4. SVR con kernel RBF: regresión no lineal, costo cuadrático.
5. Hiperparámetros clave: epsilon, C, gamma, kernel.
6. Robustez frente a outliers comparada con OLS.

Definiciones y características

- SVR (Support Vector Regression): algoritmo de regresión que busca una función que se desvíe a lo sumo ε de cada target, manteniéndola lo más "plana" posible (margen máximo).
- Epsilon-tube (tubo ε): banda de ancho 2ε alrededor de la predicción donde los errores no cuentan. Solo los puntos fuera del tubo aportan a la pérdida.
- LinearSVR: implementación lineal de SVR basada en liblinear. Escala linealmente con el tamaño del dataset; no soporta kernels.
- Kernel SVR: SVR de scikit-learn usa libsvm; soporta kernels (rbf, poly, sigmoid) para capturar no linealidad, pero su complejidad es entre O(m²) y O(m³).
- Hiperparámetro C: controla la penalización por puntos fuera del tubo. C chico → modelo más regularizado (tubo "flexible"); C grande → ajuste más estricto.
- Hiperparámetro epsilon: define el ancho del tubo de insensibilidad. ε grande → menos vectores de soporte, modelo más simple; ε chico → ajuste más fino, más vectores.

- Robust regression: SVR es menos sensible que OLS a outliers porque la pérdida es lineal fuera del tubo (no cuadrática).

Dataset / recursos

- California Housing (sklearn.datasets.fetch_california_housing) para regresión realista.
- Dataset sintético con outliers (make_regression + ruido pesado) para mostrar robustez.
- Géron, Hands-On ML (3ª ed.), cap. 5, sección "SVM Regression".

Ejercicios

1. Tubo ε en 2D. Generá y = 0.5x + ruido, entrená LinearSVR(epsilon=0.5) y graficá la recta junto al tubo ±ε. Marcá los vectores de soporte (puntos fuera del tubo).
2. Efecto de epsilon. Repetí el ejercicio 1 con ε {0.1, 0.5, 1.5}. ¿Cómo cambia la cantidad de vectores de soporte y el MSE en test?
3. Kernel RBF. En un dataset no lineal (y = sin(x) + ruido), compará LinearSVR vs SVR(kernel="rbf", gamma="scale"). Reportá MAE y graficá ambas curvas.
4. Grid search. Sobre California Housing, hacé GridSearchCV con SVR variando C {0.1, 1, 10} y gamma {"scale", 0.01, 0.1}. No uses más de 5 000 muestras (cuidado con el costo cuadrático).
5. Robustez vs OLS. Inyectá 5% de outliers en un dataset lineal y compará LinearRegression, Ridge y LinearSVR. ¿Cuál degrada menos?

Homework verificable

Entregar un script svr_california.py que:

1. Cargue California Housing y aplique StandardScaler (¡SVR exige escalado!).
2. Entrene LinearSVR(epsilon=0.5, C=1.0, random_state=42) y un SVR(kernel="rbf", C=10, gamma="scale") sobre un subset de 5 000 muestras.
3. Reporte RMSE en test para ambos modelos.

Criterio de aceptación: LinearSVR RMSE < 0.85 y SVR-RBF RMSE < 0.65 sobre el split de test (train_test_split(random_state=42, test_size=0.2)).

Errores comunes

- No escalar los features. SVR es extremadamente sensible a la escala: sin StandardScaler, los hiperparámetros pierden sentido y el entrenamiento no converge.
- Usar SVR con kernel sobre datasets grandes. Para más de ~10 000 muestras, SVR se vuelve impracticable. Usá LinearSVR o SGDRegressor(loss="epsilon_insensitive").
- Confundir epsilon de SVR con epsilon de optimizadores. Acá ε es el ancho del tubo de tolerancia, no una tolerancia numérica de convergencia.
- Dejar C en el default sin tunear. El default C=1.0 rara vez es óptimo; siempre validá con CV.
- Comparar tiempos sin dual="auto". En LinearSVR, el parámetro dual cambia drásticamente el tiempo según n_samples vs n_features.

Preguntas frecuentes

- ¿SVR vs Linear Regression cuándo conviene cuál? OLS minimiza el error cuadrático medio (sensible a outliers, óptimo si el ruido es gaussiano). SVR ignora errores chicos (dentro del tubo ε) y penaliza linealmente los grandes: más robusto a outliers y suele generalizar mejor con pocos datos ruidosos. Si tu dataset es limpio y grande, OLS/Ridge suele ser más simple y rápido.
- ¿Cómo elijo epsilon? Empezá con una fracción del desvío estándar del target (p. ej. $\varepsilon \approx 0.1 \cdot \sigma(y)$) y ajustá con CV. Un ε muy chico convierte SVR en algo cercano a regresión cuadrática; uno muy grande genera un modelo casi constante.
- ¿SVR devuelve probabilidades o intervalos? No. SVR predice un valor puntual. Para intervalos de predicción usá quantile regression, conformal prediction o un modelo bayesiano.
- ¿Por qué SVR con kernel es tan lento? Calcula la matriz kernel (m × m) y resuelve un QP. La complejidad va de $O(m^2)$ a $O(m^3)$. Para datasets grandes, LinearSVR o Nystroem + LinearSVR (kernel approximation) son alternativas viables.
- ¿Conviene SVR para deep learning / problemas con millones de muestras? No. Para esas escalas, gradient boosting (XGBoost, LightGBM) o redes neuronales superan a SVR tanto en performance como en costo.

Referencias

- Géron, A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3ª ed.), O'Reilly, cap. 5 — "SVM Regression".
- Smola, A. & Schölkopf, B. A Tutorial on Support Vector Regression (2004).
- scikit-learn docs: sklearn.svm.SVR y sklearn.svm.LinearSVR.
- scikit-learn user guide: Support Vector Machines — Regression.

Siguiente clase

Clase 080 — Árboles de decisión: entrenamiento, visualización, CART

Apéndice: notebook (primer bloque)

Primera celda ejecutable del notebook de la clase.

```
# Imports y configuración inicial
```

Archivos complementarios

- notebook.ipynb