Regresión Lineal

(Las partes que salgan en este color, no se preguntarán en el examen, pero viene bien que estén en este documento)

Métodos para ajustar un modelo de regresión lineal.

Al ajustar un modelo de regresión lineal, existen varios métodos y enfoques que se pueden utilizar para estimar los parámetros. Algunos de los métodos más comunes son:

1. **Mínimos Cuadrados Ordinarios (MCO u OLS, por sus siglas en inglés)**:
    - **Descripción**: Este es el método más tradicional y consiste en minimizar la suma de los cuadrados de las diferencias entre los valores observados y los valores predichos por el modelo.
    - **Ventajas**: Es sencillo de implementar y, bajo el supuesto de que los errores tienen distribución normal, es el estimador de máxima verosimilitud.
    - **Aplicación**: Se utiliza cuando el número de observaciones es moderado y la matriz de diseño es manejable.
2. **Máxima Verosimilitud (MLE)**:
    - **Descripción**: Bajo el supuesto de que los errores se distribuyen normalmente, la estimación de máxima verosimilitud conduce a los mismos estimadores que OLS.
    - **Ventajas**: Permite extender el análisis a modelos más complejos y facilita la incorporación de inferencia estadística (por ejemplo, intervalos de confianza, pruebas de hipótesis).
    - **Aplicación**: Es útil cuando se requiere una interpretación probabilística del modelo.
3. **Métodos de Optimización Numérica (como el Gradiente Descendente)**:
    - **Descripción**: Algoritmos iterativos que ajustan los parámetros minimizando la función de error (por ejemplo, la suma de los cuadrados) mediante pasos en la dirección del gradiente negativo.
    - **Ventajas**: Son especialmente útiles cuando se trabaja con conjuntos de datos muy grandes o cuando la solución analítica (mediante la inversa de XTX) resulta computacionalmente costosa o inestable.
    - **Variantes**: Se pueden usar variantes como el gradiente descendente estocástico (SGD), mini-batch, entre otros.
4. **Métodos Basados en Descomposiciones Matriciales**:
    - **Descripción**: Técnicas como la descomposición QR o la descomposición en valores singulares (SVD) permiten resolver el sistema de ecuaciones normal de forma numéricamente estable.
    - **Ventajas**: Ayudan a evitar problemas de inestabilidad numérica, especialmente cuando XTX está cerca de ser singular o mal condicionado.
5. **Métodos Bayesianos**:
    - **Descripción**: En este enfoque se especifican distribuciones previas para los parámetros y se utiliza la inferencia bayesiana para obtener la distribución posterior.

- **Técnicas**: Se pueden usar métodos como el muestreo Monte Carlo por cadena de Markov (MCMC) o técnicas de inferencia variacional.
- **Ventajas**: Permiten incorporar información previa y gestionar la incertidumbre de manera explícita.

6. **Métodos de Regularización (por ejemplo, Ridge y Lasso)**:
    - **Descripción**: Aunque se parte de un modelo de regresión lineal, se añaden términos de penalización (normas L2 para Ridge y L1 para Lasso) a la función de pérdida para controlar la complejidad del modelo y evitar el sobreajuste.
    - **Ventajas**: Son muy útiles cuando se tienen muchas variables o se sospecha de colinealidad entre los predictores.
    - **Aplicación**: La elección entre Ridge y Lasso dependerá de si se prefiere una penalización que tiende a distribuir la penalización de manera uniforme (Ridge) o una que favorece la eliminación de variables irrelevantes (Lasso).

Consideraciones al elegir un método

- **Tamaño y complejidad del dataset**: En datasets muy grandes, métodos iterativos como el gradiente descendente pueden ser más eficientes que calcular una solución analítica.
- **Condición de la matriz de diseño**: Si XTX es mal condicionado, métodos basados en descomposiciones matriciales o la incorporación de regularización pueden mejorar la estabilidad del cálculo.
- **Supuestos sobre los errores**: Si se asume normalidad de los errores, OLS y MLE son equivalentes, pero en otros contextos puede convenir un enfoque bayesiano.
- **Necesidades de interpretación y predicción**: La elección del método también puede depender de si se busca una estimación interpretable o una predicción óptima en términos de error.

Cada uno de estos métodos tiene sus propias ventajas y limitaciones, por lo que la elección dependerá del contexto específico del problema, la calidad y cantidad de los datos, y los recursos computacionales disponibles.

Hipótesis que debe cumplir un modelo de regresión lineal básica

1. Esperanza matemática nula

Esto significa que el término de error (ϵ) tiene un valor esperado de **0** en cada observación.

Ejemplo:
Si intentamos predecir el salario de una persona en función de su experiencia, la regresión puede hacer buenas predicciones en promedio, pero habrá desviaciones aleatorias (errores). Lo importante es que estos errores no sean sistemáticos, es decir, que no tiendan a ser siempre positivos o siempre negativos, sino que se equilibren alrededor de **cero**.

2. Homocedasticidad (Varianza constante)

Los errores tienen **la misma varianza** en todas las observaciones.

Ejemplo:
Si modelamos el precio de casas en función de su tamaño, **homocedasticidad** significa que el error en las predicciones es el mismo tanto para casas pequeñas como para casas grandes.

Si la varianza cambia (heterocedasticidad), los errores en casas más grandes podrían ser más grandes que en casas pequeñas, lo que haría que el modelo sea menos confiable.

3. Incorrelación o independencia de los errores

Los errores de una observación no deben estar correlacionados con los errores de otras observaciones.

Ejemplo:
Si estamos prediciendo el consumo de electricidad en función del clima, los errores de predicción de un día no deberían estar relacionados con los errores del día anterior.

Si hay correlación (autocorrelación), significa que si cometimos un error en una observación, ese error puede afectar otras observaciones, lo que indica que el modelo no está capturando bien la estructura de los datos.

4. Regresores no estocásticos (Variables explicativas fijas o deterministas)

Las variables independientes (predictoras) no deben ser aleatorias. Esto significa que en nuestro análisis de regresión, **tratamos a las variables predictoras como valores fijos y conocidos, no como variables aleatorias** que cambian cada vez que tomamos una nueva muestra de datos.

Ejemplo:
Si estamos modelando la relación entre la **temperatura y la cantidad de helados vendidos**, la temperatura debe ser considerada como una variable fija en el análisis, no como una variable aleatoria.

Si las variables predictoras son estocásticas (aleatorias), los coeficientes estimados pueden ser sesgados e inconsistentes.

5. Independencia lineal (No multicolinealidad)

No debe haber una relación lineal exacta entre las variables explicativas.

Ejemplo:
Si queremos predecir el precio de un coche usando las variables **edad del coche** y **kilómetros recorridos**, y añadimos otra variable que sea simplemente el doble de los kilómetros recorridos, entonces hay dependencia lineal entre las variables.

Si hay multicolinealidad, el modelo tendrá problemas para estimar los coeficientes de manera precisa, y los valores pueden volverse inestables.

6. T > k + 1 (Número de observaciones mayor que el número de parámetros)

Necesitamos más observaciones que variables explicativas para poder estimar el modelo correctamente.

Ejemplo:
Si queremos ajustar una regresión lineal con **5 variables predictoras**, necesitamos al menos **6 observaciones** para poder estimar el modelo.

Si hay menos observaciones que variables, el modelo no se puede calcular correctamente.

7. Normalidad de los errores

Los errores deben seguir una **distribución normal**.

Ejemplo:
Si estamos prediciendo la estatura de una persona en función de la edad, los errores (las diferencias entre la predicción del modelo y la estatura real) deberían distribuirse de manera simétrica alrededor de cero, sin sesgos.

Si los errores no son normales, puede afectar pruebas de hipótesis y la inferencia estadística, aunque para grandes cantidades de datos, esto no suele ser un problema grave.

¿Cómo funciona `LinearRegression()` en Scikit-Learn?

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X, Y)
```

Scikit-Learn **resuelve la ecuación de regresión lineal analíticamente** usando la **fórmula cerrada de mínimos cuadrados**: $\beta=(X^TX)^{-1}X^TY$

Ventajas del método analítico (OLS)

- Es **rápido y exacto** cuando el número de datos no es muy grande.
- No necesita iteraciones como el descenso de gradiente.

¿Cuándo se usa descenso por gradiente en Scikit-Learn?

Si tienes un **dataset muy grande** (millones de datos o muchas variables), calcular $(X^TX)^{-1}$ puede ser muy **costoso computacionalmente**. En esos casos, se usa una **versión iterativa basada en descenso por gradiente**.

Para esto, **Scikit-Learn tiene `SGDRegressor()`**, que usa **descenso de gradiente estocástico (SGD)** en lugar de OLS:

```
from sklearn.linear_model import SGDRegressor
model = SGDRegressor(max_iter=1000, tol=1e-3)
model.fit(X, Y.ravel())  # Y.ravel() aplana el array si es necesario
```

¿Diferencias entre `LinearRegression()` y `SGDRegressor()`?

Método	Algoritmo	Cuándo usarlo
`LinearRegression()`	Mínimos cuadrados (OLS, solución exacta)	Datos pequeños o medianos
`SGDRegressor()`	Descenso por gradiente estocástico (SGD, iterativo)	Datos masivos (millones de filas)

¿Qué hiperparámetros tiene SGDRegressor()?

`SGDRegressor()` en **Scikit-Learn** tiene varios **hiperparámetros** que controlan su comportamiento y rendimiento. Aquí te explico los más importantes:

Parámetro	Descripción	Valor por defecto
`loss`	Función de pérdida a minimizar. Puede ser `'squared_error'` (MSE), `'huber'`, `'epsilon_insensitive'`, etc.	`'squared_error'`
`penalty`	Tipo de regularización. Puede ser `'l2'` (Ridge), `'l1'` (Lasso) o `'elasticnet'` (combinación).	`'l2'`
`alpha`	Tasa de regularización (λ). Controla la penalización de los coeficientes. Valores más altos reducen el sobreajuste.	`0.0001`
`learning_rate`	Estrategia de ajuste de la tasa de aprendizaje. Opciones: `'constant'`, `'optimal'`, `'invscaling'`, `'adaptive'`.	`'invscaling'`
`eta0`	Valor inicial de la tasa de aprendizaje (η), si `learning_rate='constant'` o `'invscaling'`.	`0.01`
`max_iter`	Número máximo de iteraciones para la convergencia.	`1000`
`tol`	Tolerancia para la convergencia (detiene el entrenamiento si la mejora es menor que este valor).	`1e-3`
`early_stopping`	Si es `True`, usa un conjunto de validación para detener el entrenamiento temprano si no hay mejora.	`False`
`n_iter_no_change`	Número de iteraciones sin mejora antes de detener el entrenamiento si `early_stopping=True`.	`5`
`shuffle`	Si es `True`, reorganiza los datos en cada iteración para mejorar la convergencia.	`True`
`epsilon`	Parámetro para las pérdidas `'huber'` y `'epsilon_insensitive'`. Define un umbral dentro del cual los errores no se penalizan.	`0.1`
`power_t`	Parámetro que controla la tasa de decrecimiento en `invscaling`.	`0.25`

Regularización

La regularización es una técnica clave en Machine Learning que nos ayuda a evitar el sobreajuste y mejorar la generalización de los modelos. Existen tres métodos principales de regularización para la regresión lineal:

- **Lasso (L1)**
- **Ridge (L2)**
- **Elastic Net (L1 + L2 combinadas)**

¿Qué es la regularización y por qué es importante?

En **regresión lineal**, buscamos encontrar los coeficientes β que minimicen el **error cuadrático medio (MSE)**:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

Sin embargo, cuando tenemos muchas variables predictoras (X), el modelo puede **ajustarse demasiado a los datos de entrenamiento** (sobreajuste), capturando incluso ruido irrelevante.

Solución: La regularización añade una **penalización** a los coeficientes del modelo, forzando a que algunos sean más pequeños o incluso **cero**.

Ridge Regression (Regularización L2)

En Ridge, **se agrega una penalización a la suma de los cuadrados de los coeficientes (ℓ_2)**:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2$$

Donde:

- **λ** → Controla la penalización. Si **λ** es alto, los coeficientes serán más pequeños.
- **$\sum \beta_j^2$** → La regularización hace que los coeficientes sean pequeños, pero **nunca cero**.

Ventajas de Ridge:

- Reduce el sobreajuste cuando hay **multicolinealidad** (variables altamente correlacionadas).
- Útil cuando todas las variables son importantes y queremos reducir su influencia sin eliminarlas.

Lasso Regression (Regularización L1)

Lasso aplica una penalización a la **suma del valor absoluto de los coeficientes (ℓ_1)**:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

Diferencia clave con Ridge:

- **Lasso puede hacer que algunos coeficientes sean exactamente cero**, eliminando variables irrelevantes.
- Esto significa que **también actúa como un método de selección de características**.

Ventajas de Lasso:

- **Elimina variables irrelevantes**, lo que simplifica el modelo.
- Útil cuando hay **muchas variables y queremos seleccionar las más importantes**.

Elastic Net (Combinación de L1 y L2)

Elastic Net combina **Ridge y Lasso**, usando una combinación de penalización L1 y L2:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} \beta_j^2$$

Ventajas de Elastic Net:

- **Elimina variables irrelevantes (como Lasso), pero mantiene estabilidad en los coeficientes (como Ridge).**
- Útil cuando hay muchas variables y algunas están altamente correlacionadas.
- Permite ajustar el peso entre L1 y L2 con un hiperparámetro α:
  - $\alpha \times L1 + (1-\alpha) \times L2$
    - **Si α=1** → Es **Lasso** puro.
    - **Si α=0** → Es **Ridge** puro.
    - **Si α está entre 0 y 1**, obtenemos un equilibrio entre ambos.

¿Cuándo la regresión lineal es mejor sin regularización?

La **regresión lineal sin regularización** es preferible en los siguientes casos:

1. Cuando el número de muestras (n) es mucho mayor que el número de características (m)

- **Si $n \gg m$, el modelo tiene suficiente información** para aprender sin sobreajustar.
- **Ejemplo:**
  - Si tienes **10,000 datos** y solo **5 variables predictoras**, el modelo puede generalizar bien sin regularización.

¿Por qué no usar regularización?

- No hay riesgo significativo de sobreajuste.
- Regularizar podría hacer que los coeficientes se reduzcan innecesariamente y **perder información útil**.

2. Cuando no hay colinealidad entre las variables predictoras

Si las variables son independientes, la regresión lineal estándar encuentra la mejor solución sin necesidad de ajustar coeficientes.

¿Por qué no usar regularización?

- Ridge y Elastic Net ayudan cuando las variables están correlacionadas, pero si no lo están, **no se necesita penalización**.

3. Cuando hay pocas variables irrelevantes o ruido bajo

Si todas las variables son importantes y no hay muchas características irrelevantes, **la regularización no es necesaria.**

¿Por qué no usar regularización?

- **Lasso eliminaría variables** importantes innecesariamente.
- **Ridge reduciría coeficientes útiles**, haciendo el modelo menos interpretable.

4. Cuando se necesita interpretar los coeficientes tal como son

- **La regularización modifica los coeficientes, lo que puede dificultar la interpretación.**
- **Ejemplo:**
  - Modelos en **economía, medicina o ciencias sociales**, donde **cada coeficiente tiene un significado real** (ej., cuánto influye el consumo de azúcar en el peso de una persona).

¿Por qué no usar regularización?

- Ridge/Lasso pueden hacer que los coeficientes sean más difíciles de interpretar.

Resumen

Caso	¿Se recomienda regularización?	Motivo
$n \gg m$ (muchas muestras, pocas variables)	No	El modelo generaliza bien sin regularización.
Variables independientes (sin colinealidad)	No	No hay problemas de coeficientes inflados.
Pocas variables irrelevantes o ruido bajo	No	Regularizar podría eliminar o reducir coeficientes útiles.
Necesidad de interpretar coeficientes exactos	No	Regularización puede alterar la interpretación.

¿Es necesario escalar los datos si usamos `LinearRegression()` sin regularización?

- **La respuesta corta: No es obligatorio, pero en algunos casos es recomendable**.
- **La respuesta larga:** Depende de si **quieres mejorar la estabilidad numérica del modelo y la interpretación de los coeficientes**.

Cuándo NO es necesario escalar los datos

Si solo usas **`LinearRegression()` sin regularización**, la escala de las variables **no afecta los coeficientes** en términos de predicción.
Esto se debe a que **la regresión lineal encuentra la mejor solución analítica sin depender de la escala**.

Ejemplo donde no hace falta escalar:

- Si todas las variables están en **la misma escala** (ej., todas en metros o en segundos).
- Si no te interesa interpretar los coeficientes en una escala comparable.

Cuándo SÍ es recomendable escalar los datos

En algunos casos, escalar **puede ser útil** en regresión lineal estándar:

1. Cuando hay grandes diferencias en la escala de las variables

- Si una variable representa **edad en años (1-100)** y otra representa **ingresos en dólares (miles a millones)**, los coeficientes tendrán escalas muy distintas.
- Esto **no afecta la precisión del modelo**, pero puede hacer que **los coeficientes sean difíciles de interpretar**.
- **Ejemplo:**
  Si $\mathbf{X_1}$ **= edad** (rango: 1-100) y $\mathbf{X_2}$ **= ingreso** (rango: 10,000-1,000,000):
    - $Y = 0.02X_1 + 0.0000003X_2$
  - Aquí, parece que X_1 (edad) tiene más impacto en Y, pero en realidad **la escala de X_2 es mucho mayor**, lo que reduce su coeficiente.
  - **Con escalado (`StandardScaler()`)**:
    - $Y = 0.8X_1 + 0.6X_2$
  - Ahora los coeficientes son **comparables** en magnitud.

2. Para mejorar estabilidad numérica

- Si los valores de las variables son muy grandes, puede generar **problemas de precisión en los cálculos matriciales**.
- Esto **no suele ser un problema en datasets pequeños**, pero en modelos grandes puede hacer que los coeficientes tengan **valores muy altos o fluctuaciones**.

Conclusión

Situación	¿Es necesario escalar?	Motivo
Regresión lineal (`LinearRegression()`) sin regularización	No obligatorio	La escala no afecta la solución.
Ridge, Lasso o Elastic Net	Sí	Son sensibles a la escala porque penalizan los coeficientes.
Diferencias grandes en la escala de las variables	Recomendable	Para hacer los coeficientes más comparables.

Situación	¿Es necesario escalar?	Motivo
Problemas de estabilidad numérica	Recomendable	Evita valores muy grandes o fluctuaciones en los coeficientes.

¿Es necesario escalar los datos cuando usamos `PolynomialFeatures()` con regularización?

Sí, es altamente recomendable escalar los datos si estás usando **`PolynomialFeatures()` junto con regularización (Ridge, Lasso o Elastic Net)**.

Cuando usas **`PolynomialFeatures()`**, los valores de las nuevas variables pueden crecer **muy rápido**. Esto causa:

- **Coeficientes grandes e inestables** → Especialmente en polinomios de alto grado.
- **Una penalización desbalanceada en Ridge y Lasso** → Las características con valores más grandes dominan la regularización.
- **Problemas numéricos en el entrenamiento** → Pérdida de precisión debido a la diferencia de escala.