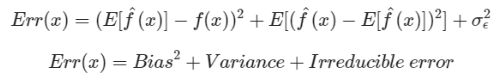
**Regularización**

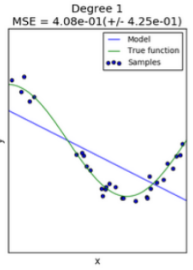
**Sesgo – Varianza:** Tenemos un modelo , para el cual épsilon es un término aleatorio con distribución: , podemos obtener una estimación de f para hacer predicciones sobre Y: 

La **esperanza del error de predicción al cuadrado** será 

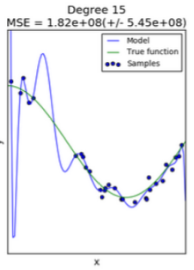
La podemos descomponer de la siguiente manera:



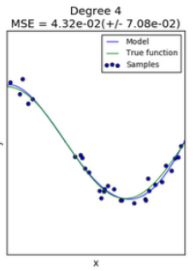
Con un **modelo demasiado simple** (**pocos grados de libertad**)independientemente del tamaño de la muestra, tendremos **sesgo o error sistémico**:



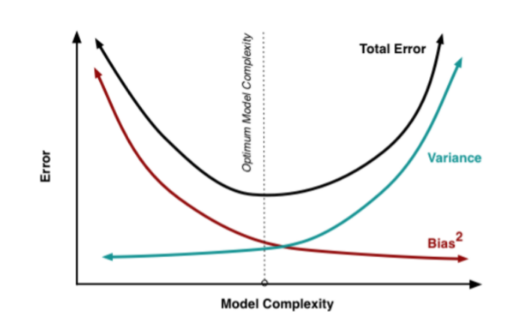
Mientras que con un **modelo demasiado complejo** (**demasiados grados de libertad**),tendremosun problema de **sobreajuste**:



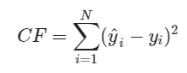
El ideal es un punto intermedio entre lo muy simple y lo muy complejo:



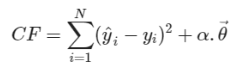
Como vemos, aumentar la complejidad del modelo reduce el error de bias, pero aumenta la varianza del error. El error total entre lo que aporta el bias y lo que aporta la varianza se encuentra en un punto intermedio:



**Regularización:**

Dada una función de pérdida de una regresión lineal: 

Con las **técnicas de regularización** lo que se hace es agregar una **penalidad** a esta función de costo, siendo θ (theta) el vector con los parámetros del modelo (betas, en el caso de una regresión lineal) y α un parámetro que “regula” la fuerza de la penalización. A mayor α, mayor penalización:



Dos Técnicas de Regularización:

1. **Regresión Ridge**
2. **Regresión Lasso**

Lo que proponen estas técnicas es cambiar ligeramente el problema de optimización de mínimos cuadrados, para tratar de achicar (**shrink**) el valor absoluto de los estimadores de los β. Este método lo que hace es **introducir un sesgo** **a cambio** de una **reducción en la varianza**.

**Norma L0:** Cantidad de elementos distintos de cero en el vector.

**Norma L1:** Suma de valores absolutos de los elementos del vector:



**Norma L2:** Raíz cuadrada de la suma de los cuadrados de los elementos del vector:



**Normalización L1:** Hacer que la suma del valor absoluto de uno.

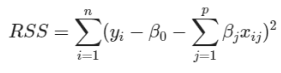


**Normalización L2:** Hacer que la suma del valor absoluto al cuadrado de uno.

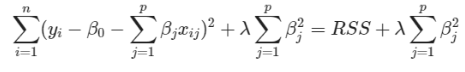


**Regresión Ridge:**

Esta es la función en la que se minimiza en la estimación de mínimos cuadrados:



La función que se minimiza con la Regresión Ridge es:

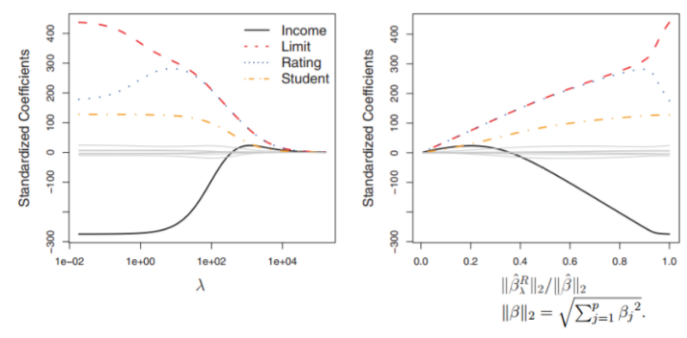


Se agrega un término nuevo, con un **hiperparámetros λ** que penaliza el valor de los coeficientes al cuadrado. Por lo tanto, esto va a hacer que busquemos minimizar el cuadrado de los errores, de forma tal que ningún βj2 sea demasiado grande.

Como con **MCO**, se busca achicar la función de pérdida; hay un **término de penalización** que se achica en la medida en la que los betas se aproximen a cero (independientemente de si son positivos o negativos); el hiperparámetro λ maneja la ponderación de cada término. La elección del mejor valor para λ se hace mediante **CROSS VALIDATION**.

En los siguientes gráficos, la línea negra representa la estimación de la regresión de Ridge para los distintos coeficientes en función de λ: A un λ cercano a 0, los coeficientes tienden a tener el valor de una regresión lineal sin regularización. A medida que λ va aumentando, para contrarrestar este aumento en la Función de Costo, los coeficientes van reduciéndose hasta tender a 0.

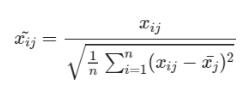
El gráfico de la derecha es otra forma de interpretar lo mismo. En el eje X está la relación de los coeficientes aplicando regularización sobre los coeficientes sin aplicar la regularización. Cuando esta relación es 0, los coeficientes con regularización tienden a cero, y esto ocurre cuando el λ es muy grande; por otro lado, si esta relación es 1, entonces los coeficientes con regularización coinciden con los coeficientes sin regularizar. Y esto tiende a ocurrir cuando λ tiende a cero. O Sea, ambos gráficos son como espejos.



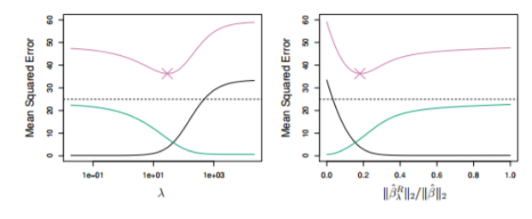
En el **extremo izquierdo**, **λ es prácticamente 0** y las **estimaciones** de los coeficientes de **Ridge** terminan **coincidiendo** prácticamente con las estimaciones de **Mínimos Cuadrados**. Yendo al **extremo derecho**, aumentando λ, todas las **estimaciones** del Coeficiente de **Ridge** son **cero** (modelo nulo que no contiene predictores).

Mirando el otro gráfico, se puede interpretar el eje x como la cantidad en que se redujo el tamaño de las estimaciones del coeficiente de regresión de Ridge.

Es importante tener en cuenta que en la **Regresión Ridge,** tanto la **estimación de los coeficientes** como la **predicción** **son sensibles a la escala**. Esto quiere decir que si una variable se encuentra en una escala tal que le da un valor absoluto mayor, esto **va a afectar** el **cálculo** de la **suma de cuadrados** del **vector de coeficientes**. Por esto, es **importante estandarizar** (**dividir por el desvío estándar**)todos los **regresores antes** **de ejecutar** una **Regresión Ridge**. De forma tal que no estén en unidades físicas si no en unidades de su propio desvío estándar.



**Trade-off Bias-Variance:** Para n = 50 simulaciones con p = 45 predictores, todos con coeficientes no nulos, podemos ver el **sesgo cuadrado**, la **varianza** y el **MSE** del test para una **Regresión Ridge** en los datos simulados, como una función de λ y de .



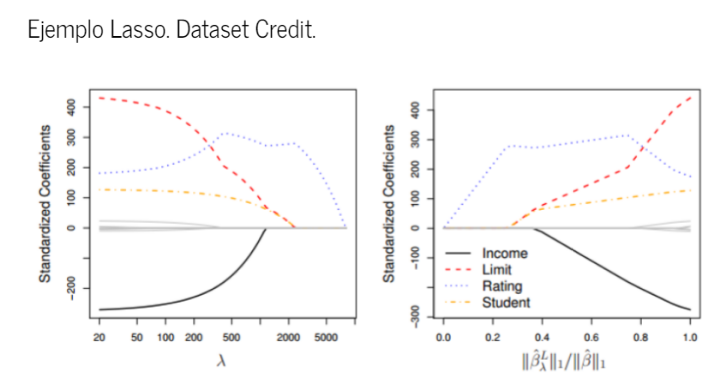
En el gráfico de la izquierda la línea negra representa la evolución del sesgo en la medida en que va aumentando λ (empieza en un modelo que overfitea muchísimo y se va haciendo cada vez más simple hasta que ya no nos sirve); la línea verde representa la evolución de la varianza en la medida que va aumentando λ (arranca alta y a medida que vamos simplificando el modelo, va bajando). El ideal es aquél valor de λ para el cual el efecto de sesgo + varianza es el menor posible. El gráfico de la derecha es el espejo de este gráfico, graficando en función del ratio entre los coeficientes regularizados y los coeficientes sin regularizar.

La **desventaja** de la **Regresión Ridge** es que **incluye a todos los predictores p** en el modelo final; en lugar de elegir un subconjunto de predictores.

**Regresión Lasso:** Introduce coeficientes  que minimizan el número de predictores, corrigiendo la gran desventaja de la Regresión Ridge. Mientras Ridge usa la norma L2 en la penalización, **Lasso utiliza la norma L1**.

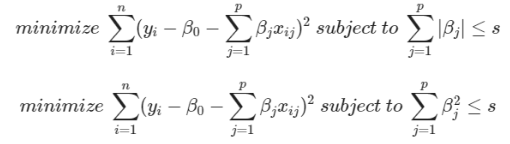
La norma de L1 de un vector de coeficientes β está dada por .

Al igual que con la Regresión Ridge, la Regresión Lasso **achica los coeficientes** estimados. Si λ es lo suficientemente grande, la regularización L1 fuerza los coeficientes a valer exactamente cero. Lasso genera **modelos dispersos** (modelos con una selección de variables). Una gran diferencia contra Ridge es que los coeficientes van convergiendo a cero a destiempo; con Ridge tienden a cero todos juntos para un mismo λ. Análogamente a como con la Regresión Ridge, con la Regresión Lasso es **crítico elegir un buen valor de λ**. Y esto se logra también con **cross-validation**.

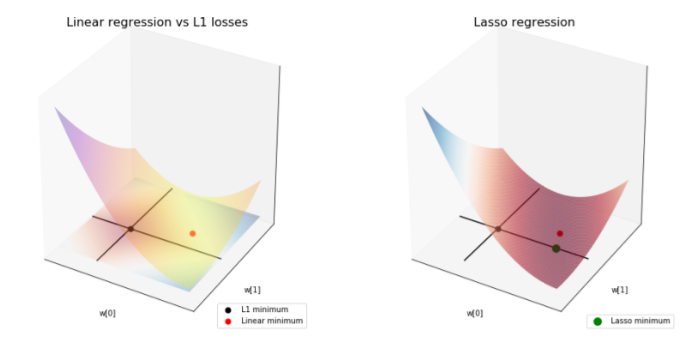


**Comparación entre Lasso y Ridge:**

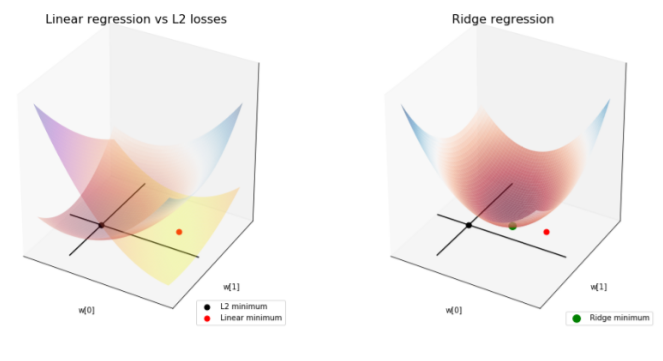
La estimación de coeficientes de la Regresión Lasso y Ridge resuelve estos problemas, respectivamente:

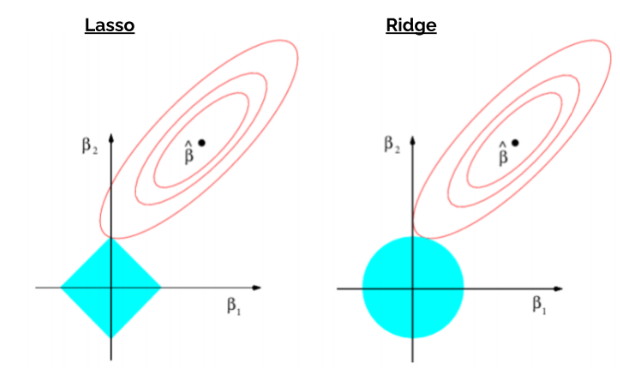


Lasso Gráficamente:

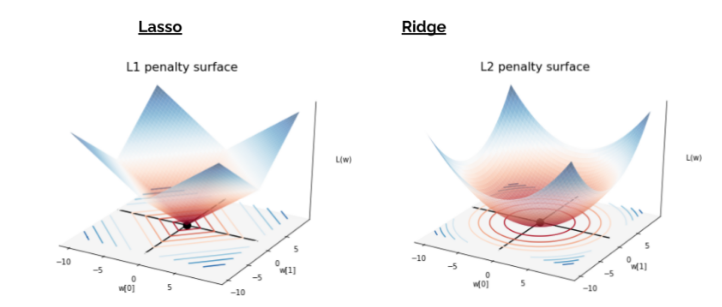


Ridge Gráficamente:





Las curvas del gráfico de arriba son curvas con la misma cantidad de error, dados dos coeficientes.



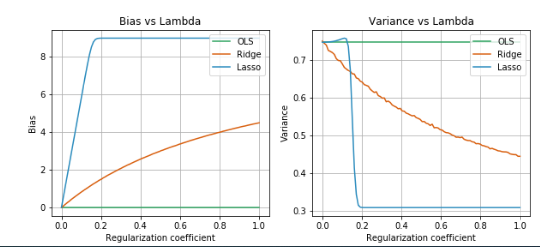
El **sesgo (bias)** se calcula como la distancia desde la predicción promedio y el valor verdadero; o sea, es el valor verdadero menos la media de las predicciones:



La **Varianza** es el desvío promedio de la predicción promedio; o sea, el valor medio de la predicción menos la media de las predicciones:

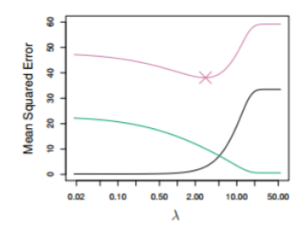


Así evolucionan el sesgo y la varianza con λ, podemos ver que Lasso tiene cambios fuertes sobre todo al principio, mientras que los cambios en Ridge son más paulatinos:

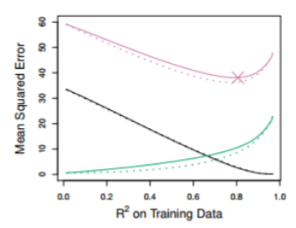


Estos datos se generaron **haciendo que todos los coeficientes fueran diferentes a cero**. Cuando esto ocurre, **los dos modelos tienden a performar prácticamente igual**. Ridge tiene menor varianza y por eso parece mejorar con respecto a Lasso.

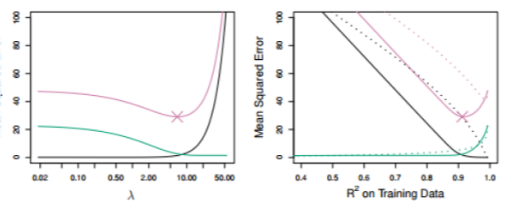
Acá vemos la evolución del sesgo, varianza y MSE para Lasso:



Acá podemos ver la comparación entre ambas regresiones. Las líneas continuas son de Lasso y las líneas punteadas son de Ridge.



Ahora bien, empezando a permitir que sólo dos coeficientes sean diferentes a cero, la situación cambia notablemente, en favor de Lasso (menor varianza y menor MSE):



**ElasticNet:** Lo que hace es combinar linealmente lo mejor de ambas regresiones. Se regula la complejidad del modelo con el parámetro λ, mientras que con el parámetro α se regula la importancia relativa de Lasso vs Ridge. De esta forma, podemos obtener soluciones parsimoniosas y bien condicionadas. Pero a expensas de que ahora hay que calibrar dos hiperparámetros.

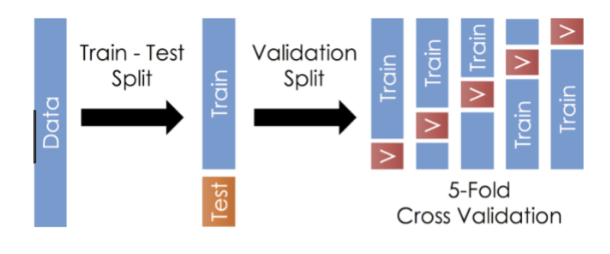


**Cross Validation:**

**Selección de Hiperparámetros para Ridge y Lasso**: Necesitamos un método para poder seleccionar el **mejor valor para el hiperparámetro λ.**

Una forma simple de abordar este problema es con **Cross Validation,** se elige un rango de valores que puede tomar el hiperparámetro y se computan los errores que devuelve cross-validation para cada valor del rango. Luego, se elige el hiperparámetro que tenga el menor error computado. Por último, se reentrenará el modelo con el valor elegido para el hiperparámetro.

1. Se hace el **split train/validación** y **test.**
2. Se **divide el dataset de train/validación en k grupos** (generalmente 5 o 10) del **mismo tamaño**.
3. En la i-ésima iteración, el i-ésimo grupo generado funciona como un **conjunto de validación**, mientras que el resto de los grupos es el **conjunto de training.**
4. Se **entrena un modelo** sobre el **conjunto de training** recién definido.
5. Se hacen las **predicciones** sobre el **conjunto de validación** y **se calcula el error** sobre este conjunto.
6. Se **repiten los pasos 3 a 5 k veces**; **variando** los **conjuntos de entrenamiento** y de **validación** en cada iteración.
7. Se **promedian** los **k errores obtenidos** (un valor de error por cada una de las iteraciones)



Ejemplo: Se busca predecir el valor de la variable **Balance** en el dataset Credit, usando como variables predictoras **Income**, **Limit**, **Rating**, **Age** y la **Regularización Ridge**. También se va a usar como variable predictora **Rating\_2**, resultado de 2 \* **Rating** para ejemplificar el efecto de la regularización sobre un predictor colineal. Queremos encontrar el mejor valor para λ y α, y conocer el valor de R2 en el test.

IE: **En Python**

from sklearn import linear\_model

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

data = pd.read\_csv(‘../Data/Credit.csv’)

data[‘Rating\_2’] = 2 \* data.Rating

X = data[[‘Income’, ‘Limit’, ‘Rating’, ‘Age’, ‘Rating\_2’]]

scaler = StandardScaler()

X\_std = scaler.fit\_transform(X)

y = data[‘Balance’]

X\_train, X\_test, y\_train, y\_test, = train\_test\_split(X\_std, y, test\_size = 0.3, random\_state = 117)

model\_ridge\_cv = linear\_model.RidgeCV(alphas=[0.3, 0.5, 1.0, 1.1, 1.15, 1.17, 1.18, 1.19, 1.2, 1.21, 1.22, 1.3, 1.4, 1.5, 10.0], fit\_intercept=True, normalize = False, cv=10)

model\_fit\_ridge\_cv = model\_ridge\_cv.fit(X\_train, y\_train)

print(model\_fit\_ridge\_cv.alpha\_)

print(model\_fit\_ridge\_cv.best\_score\_)



best\_alpha = model\_fit\_ridge\_cv.alpha\_

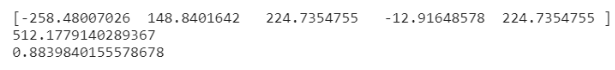
model\_ridge = linear\_model.Ridge(alpha = best\_alpha, fit\_intercept = True, normalize = False)

model\_fit\_ridge = model\_ridge.fit(X\_train, y\_train)

print(model\_fit\_ridge.coef)

print(model\_fit\_ridge.intercept\_)

print(model\_fit\_ridge.score(X\_train, y\_train))

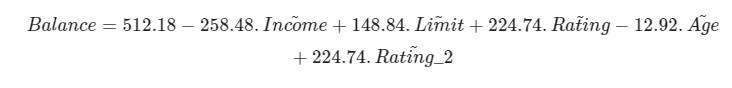


model\_fit\_ridge.score(X\_test, y\_test)



Entonces, el major valor de λ es 1.18; El valor de R2 es 0.86.

La ecuación que predice el valor de Balance es:



**Conclusión:**

Mediante la **regularización** podemos **evitar el sobreajuste** limitando la complejidad del modelo. Esto se traduce matemáticamente en una **penalización** de la **complejidad** dentro de la función de costo. Usar **modelos con regularización** les suele dar **más poder de generalización**. Podemos **determinar los hiperparámetros** usando **validación cruzada**.