# Introducción a la minería de datos

**Universidad Nacional de Colombia, sede Medellín.**

# Objetivos de la sesión

- Repaso por la métricas de desempeño.
- Regresión polinómica.
- Regularización L1 y L2.
- Ejemplo en Python.

# ¿Cómo saber si un modelo es bueno o no?

- Lo más importante es la **capacidad predictiva** del modelo.
- Hacer predicciones correctas sobre los datos de entrenamiento no es suficiente para determinar la capacidad predictiva.
- El modelo construido debe **generalizar**, es decir, debe ser capaz de realizar predicciones correctas en datos distintos a los datos de entrenamiento.
- Otros factores importantes: interpretabilidad, eficiencia.

# ¿Cómo saber si un modelo es bueno o no?

- Resumimos la **capacidad predictiva** de un modelo mediante métricas de desempeño.
- Las métricas se calculan contrastando los valores predichos versus los valores reales de la variable objetivo.
- Este se hace con datos no usados durante entrenamiento.
- Diseñamos experimentos en que comparamos las métricas de desempeño para varios modelos distintos y nos quedamos con el mejor.

# Métricas para Regresión

**Error cuadrático medio (MSE: Mean Squared Error)**

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

- Promedio de los errores cuadráticos.
- Siempre es no negativa.
- Valores cerca de 0 son *mejores.*
- El error no está en las mismas unidades que las predicciones.

# Métricas para Regresión

**Raíz cuadrada del Error cuadrático medio (RMSE: Root Mean Squared Error)**

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- Siempre es no negativa.
- Valores cerca de 0 son *mejores.*
- El error está en las mismas unidades que las predicciones, por lo que se interpreta más fácilmente.

# Métricas para Regresión

**Error absoluto medio (MAE: Mean Absolute Error)**

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y}_i|}{n}$$

- Siempre es no negativa.
- Valores cerca de 0 son *mejores.*
- Es más fácil de interpretar que RMSE, simplemente la distancia entre la predicción y el error.
- No es tan sensible a los *outliers,* porque no eleva al cuadrado.

# Métricas para Regresión

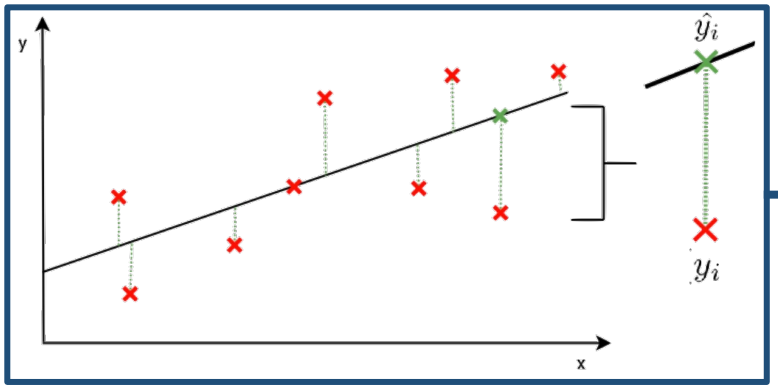
**MAPE-Mean Absolute Percentage Error**

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100$$

- Promedio de los errores absolutos en términos porcentuales. Es útil para entender el error en términos relativos a los valores reales.
- El MAPE se expresa en porcentaje, lo que facilita la interpretación del error en términos relativos. Un MAPE del 10% indica que, en promedio, las predicciones del modelo se desvían un 10% de los valores reales.
- A diferencia de métricas como el Error Cuadrático Medio (MSE) o el Error Absoluto Medio (MAE), el MAPE no está influenciado por la escala de los datos, ya que se basa en porcentajes.
- Sensibilidad a valores cero o cercanos a cero.

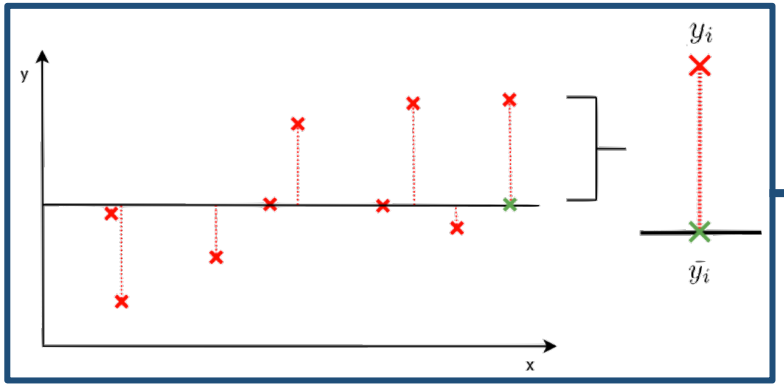
# Métricas para Regresión

**R Cuadrado o Coeficiente de determinación:**




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

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$




$$SS_{tot} = \sum_{i=1}^{n} (y_i - \bar{y}_i)^2$$

# Regresión Polinomial

La regresión polinómica es, en realidad, una regresión lineal. Ésta se compone de:

1. Agregar términos de orden superior.
2. Usar el algoritmo visto para encontrar los pesos W (regresión lineal).

Antes → $X = [x_0, x_1, x_2, x_3, ..., x_N]$

Ahora → $X_{polinómica} = [x_0, x_1, x_1^2, x_2, x_2^2, ..., x_N, x_N^2]$

- De forma general, se pueden aplicar otras transformaciones no lineales a $X$ para crear nuevas características.
- Se debe tener precaución con la expansión de características polinomiales en grados altos, ya que esto puede conducir a modelos que se sobreajustan (overfitting). La expansión de características polinomiales a menudo se combina con regularizaciones (ver ejercicio en Python : Sesión 5 Seminario ML_1).

# Regularización Lasso L1

La regularización consiste en añadir una penalización a la función de costo (o función objetivo), lo que lleva a tener modelos más simples que generalizan mejor.

$$RSS_{Lasso}(w) = \sum_{i=1}^{n} \left(y_i - (w_0 + wx_i)\right)^2 + \alpha \sum_{j=1}^{d} |W_j|$$

- Esta regularización tiene el efecto de establecer algunos pesos de los parámetros (w) en cero para las variables menos influyentes. A esto se le llama una solución dispersa: es una especie de selección de características. Esto puede ser útil para descubrir cuáles de los atributos de entrada son relevantes.
- $\alpha$ es un hiperparámetro que controla el monto de la regularización (por defecto es 1).
- Lasso funciona mejor cuando los atributos no están muy correlacionados entre ellos.
- Esto se puede aplicar a regresiones lineales, polinómicas, redes neuronales, máquinas de soporte vectorial, etc.
- LassoCV: para encontrar el mejor $\alpha$.

# Regularización L2: Regresión Ridge

La regularización consiste en añadir una penalización a la función de costo (o función objetivo) por variaciones grandes en los parámetros dados en w:

$$RSS_{Rigde}(w) = \sum_{i=1}^{n} \left(y_i - (w_0 + wx_i)\right)^2 + \alpha \sum_{j=1}^{d} W_j^2$$

- $\alpha$ es un hiperparámetro que controla la influencia de la regularización.
- Ridge nos va a servir de ayuda cuando sospechemos que varios de los atributos de entrada (features) estén correlacionados entre ellos. Ridge hace que los coeficientes acaben siendo más pequeños. Esta disminución de los coeficientes minimiza el efecto de la correlación entre los atributos de entrada y hace que el modelo generalice mejor.
- Ridge funciona mejor cuando la mayoría de los atributos son relevantes.
- RidgeCV: para encontrar el mejor $\alpha$.

# Regularización Elasticnet: L1 y L2

**Elastic Net** es un modelo de regresión lineal que normaliza el vector de coeficientes con las normas L1 y L2.

$$RSS_{elasticnet}(w) = \sum_{i=1}^{n} \left(y_i - (w_0 + wx_i)\right)^2 + \alpha \left( \lambda \sum_{j=1}^{d} |W_j| + (1-\lambda) \sum_{j=1}^{d} W_j^2 \right)$$

- La combinación de ambas penalizaciones suele dar lugar a buenos resultados. Una estrategia frecuentemente utilizada es asignarle casi todo el peso a la penalización *L1* ($\lambda$ muy próximo a 1) para conseguir seleccionar predictores y un poco a la *L2* para dar cierta estabilidad en el caso de que algunos predictores estén correlacionados.
- ElasticNetCV: para encontrar los mejores $\alpha$ y $\lambda$ (l1 ratio).

# Normalización de Características

- Es importante para algunos métodos de aprendizaje automático que todas las funciones estén en la misma escala (por ejemplo, una convergencia más rápida en el aprendizaje, una influencia más uniforme o 'justa' para todos los pesos). Ejemplo: regresiones regularizadas, máquinas de soporte vectorial, redes neuronales, etc.

- Un ejemplo de normalización, es transformar la escala con la función MinMax:

1. Para cada $x_i$ se calcula $x_i^{min}$ y $x_i^{max}$ para todas las variables en el conjunto de entrenamiento.
2. Para cada $x$, se calcula: $x_i' = (x_i - x_i^{MIN})/(x_i^{MAX} - x_i^{MIN})$

- Ajuste el escalador usando el conjunto de entrenamiento, luego aplique el mismo escalador para transformar el conjunto de prueba.
- No escale los conjuntos de entrenamiento y prueba utilizando diferentes escaladores: esto podría provocar una desviación aleatoria en los datos.

# Métricas para Clasificación

| Matriz de confusión | | Estimado por el modelo | | | |
|---|---|---|---|---|---|
| | | *Negativo (N)* | *Positivo (P)* | | |
| **Real** | *Negativo* | a: (TN) | b: (FP) | | |
| | *Positivo* | c: (FN) | d: (TP) | **Precisión** ***("precision")*** Porcentaje predicciones positivas correctas: | d/(b+d) |
| | | **Sensibilidad, exhaustividad** ***("Recall")*** Porcentaje casos positivos detectados | **Especifidad** ***(Specifity)*** Porcentaje casos negativos detectados | **Exactitud ("accuracy")** Porcentaje de predicciones correctas *(No sirve en datasets poco equilibrados)* | |
| | | d/(d+c) | a/(a+b) | (a+d)/(a+b+c+d) | |

# Métricas para Clasificación

$$accuracy(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} 1 \times (y_i = \hat{y}_i)$$

Promedio de los aciertos

$$TPR = \frac{TP}{P} = \frac{TP}{FN + TP}$$

Fracción de positivos que se lograron detectar

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN}$$

Fracción de negativos que se detectaron como positivos

# Métricas para Clasificación

$$Precision = \frac{TP}{FP + TP}$$

De los que predije positivos, cuáles en realidad lo eran

$$Recall = \frac{TP}{FN + TP}$$

De los que debía seleccionar como positivos, cuántos logré clasificar bien

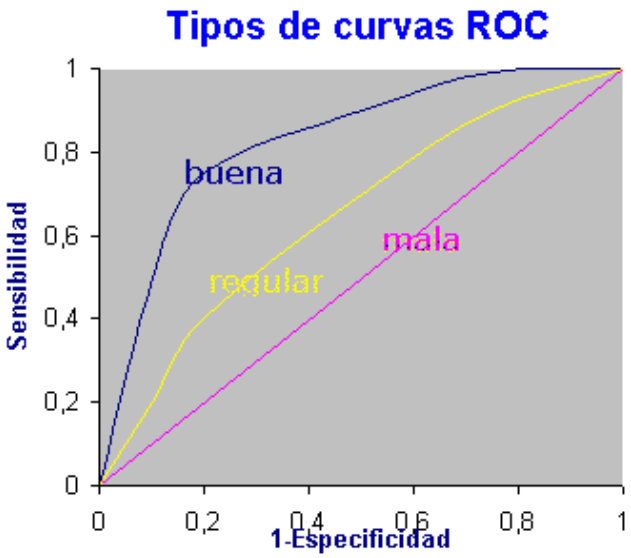
$$F1 = 2 \times \frac{precision \times recall}{precision + recall}$$

Es la media armónica de precisión y recall.

# Métricas para Clasificación

Un método para evaluar clasificadores alternativo a las métricas expuestas es la curva ROC (Receiver Operating Characteristic). La curva ROC es una representación gráfica que ilustra el rendimiento de un clasificador binario en todos los umbrales posibles. Se utiliza para evaluar y comparar modelos de clasificación en términos de su capacidad para distinguir entre las clases positiva y negativa.

El mejor método posible de predicción se situaría en un punto en la esquina superior izquierda, o coordenada (0,1) del espacio ROC, representando un 100% de sensibilidad (ningún falso negativo) y un 100% también de especificidad (ningún falso positivo). Una clasificación totalmente aleatoria daría un punto a lo largo de la línea diagonal, que se llama también línea de no-discriminación. En definitiva, se considera un modelo inútil, cuando la curva ROC recorre la diagonal positiva del gráfico. En tanto que en un test perfecto, la curva ROC recorre los bordes izquierdo y superior del gráfico. La curva ROC permite comparar modelos a través del área bajo su curva.



# Métricas para Clasificación

•El **AUC** (Area Under the Curve) es una medida del rendimiento del modelo. Representa la probabilidad de que el clasificador clasifique correctamente una instancia positiva sobre una instancia negativa de manera aleatoria.

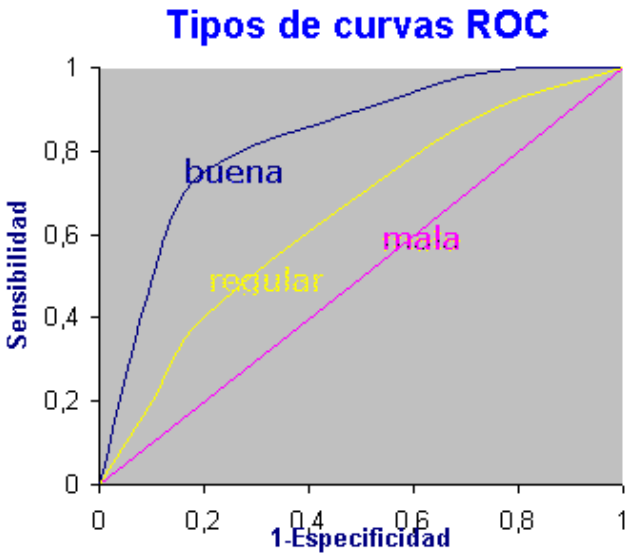
**Interpretación:**

•**AUC = 1:** Modelo perfecto; clasifica todas las instancias correctamente.

•**0.5 < AUC < 1:** Modelo con buen rendimiento; mejor que un clasificador aleatorio.

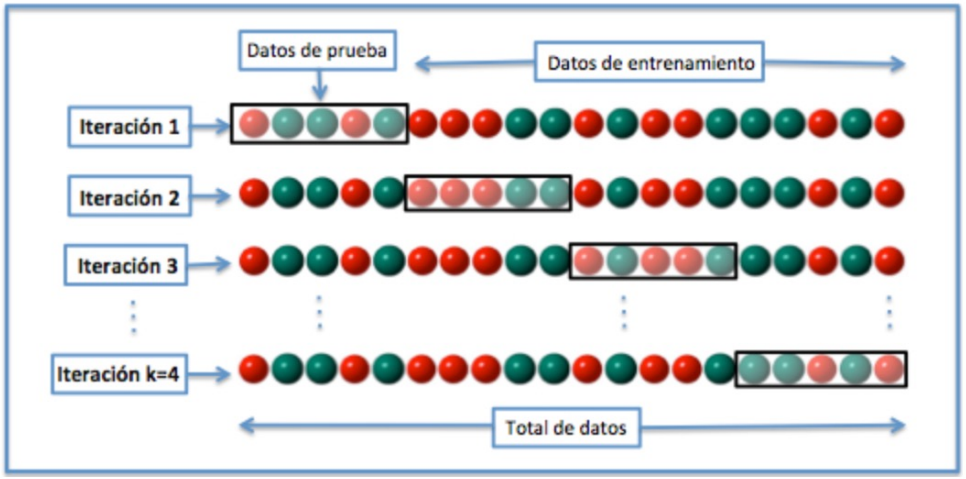
•**AUC = 0.5:** Modelo no mejor que una clasificación aleatoria.

•**AUC < 0.5:** Modelo peor que una clasificación aleatoria.



# Validación Cruzada

La validación cruzada o cross-validation es una técnica utilizada para evaluar los resultados de un análisis estadístico cuando el conjunto de datos se ha segmentado en una muestra de entrenamiento y otra de prueba, la validación cruzada comprueba si los resultados del análisis son independientes de la partición. Aunque la validación cruzada es una técnica diseñada para modelos de regresión y predicción, su uso se ha extendido a muchos otros ejercicios de machine learning.




Tomado de: https://bookdown.org/content/2274/metodos-de-clasificacion.html

# Bibliografía


•**“Machine Learning and Pattern Recognition”** de Christopher Bishop.

•**“Elements of Statistical Learning”** Hastie, Tibshirani and Friedman

•**“Machine Learning: probabilistic Perspective”,** Kevin Murphy

•“**Machine Learning Yearnin”,** Andrew Ng

Ñapa:

- Kutner, M., C. Nachtsheim, J. Neter, and W. Li. 2005. *Applied Linear Statistical Models*. Fifth. McGraw Hill/Irwin.
- Montgomery, E. & Vining, D. & Peck. 2006. *Introducción Al Análisis de Regresión Lineal*. 3ed ed. México: Cecsa.

•Link: https://fhernanb.github.io/libro_regresion/rls.html

# Gracias!!!