Análisis de datos

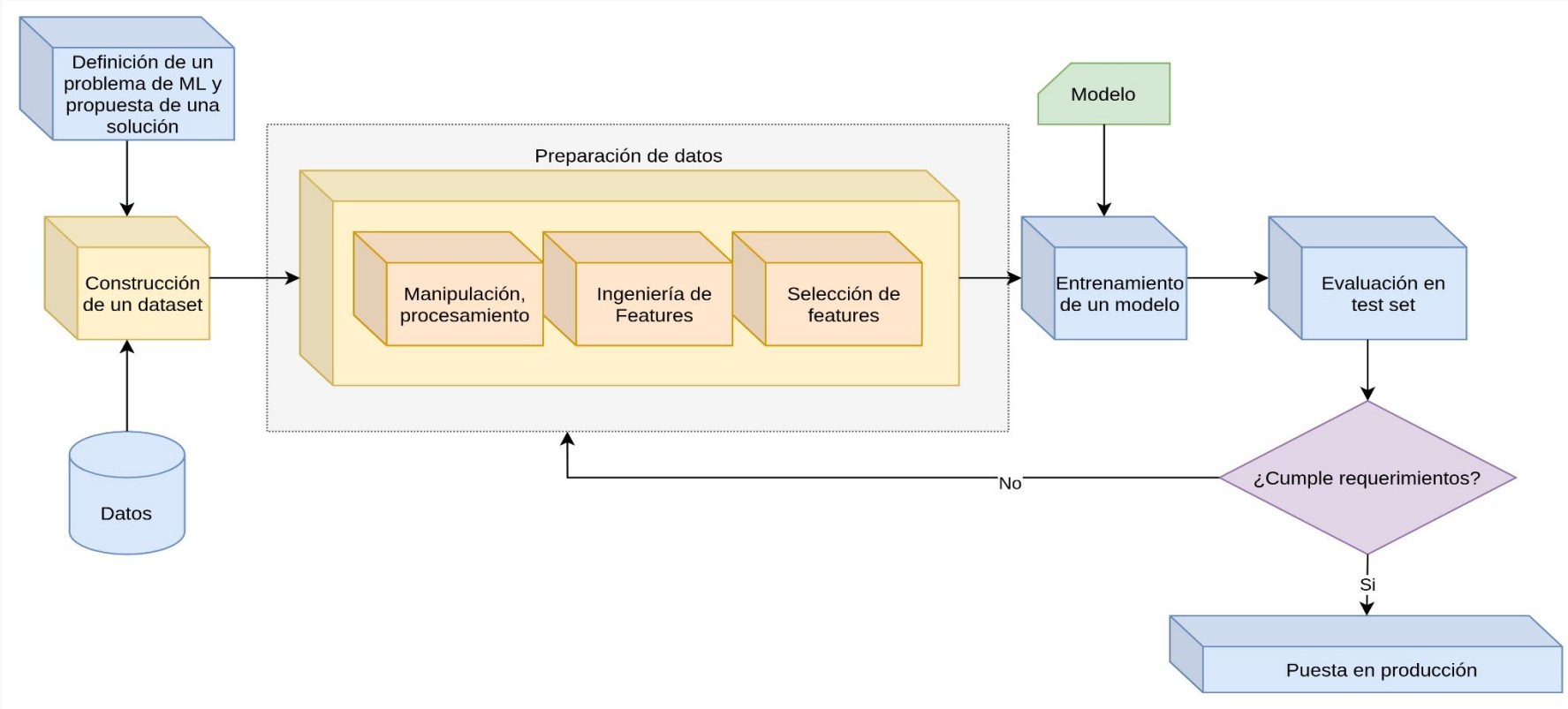
Clase 4. Taller de preparación de datos.

Temario

1. Introducción a la preparación de datos.
2. Caracterización de las variables.
3. Imputación de datos faltantes.
4. Codificación de variables categóricas.
5. Transformación de variables.
6. Discretización.
7. Tratamiento de valores extremos (outliers).
8. Feature scaling.

1. Introducción a la preparación de datos

Flujo de trabajo de un problema de ML. Tareas de preparación de datos.



Motivación

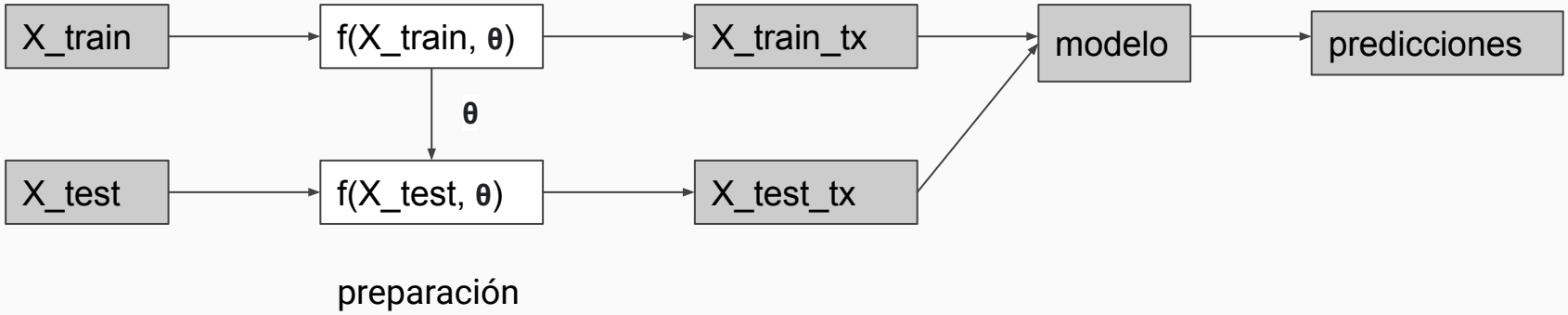
- El primer objetivo de la preparación de datos es generar un dataset apto para entrenar un modelo de aprendizaje automático.
- El segundo objetivo es optimizar su representación para mejorar el desempeño de un modelo.

Tareas de preparación de datos

- Generalmente, implica realizar una o más de las siguientes tareas:
  - **Limpieza de datos**: identificar y corregir errores en los datos.
  - **Transformación de datos**: modificar la escala o distribución de los valores.
  - **Ingeniería de features**: utilizar información del dominio del problema para seleccionar o generar variables relevantes.
  - **Selección de features**: identificar las variables de entrada de mayor relevancia para la tarea.
  - **Reducción de dimensiones**: crear proyecciones compactas de los datos.

Reproducibilidad de las tareas de preparación

- Los procesos de transformación que se apliquen durante el entrenamiento (train set) deben ser reproducibles para datos no vistos (test set).

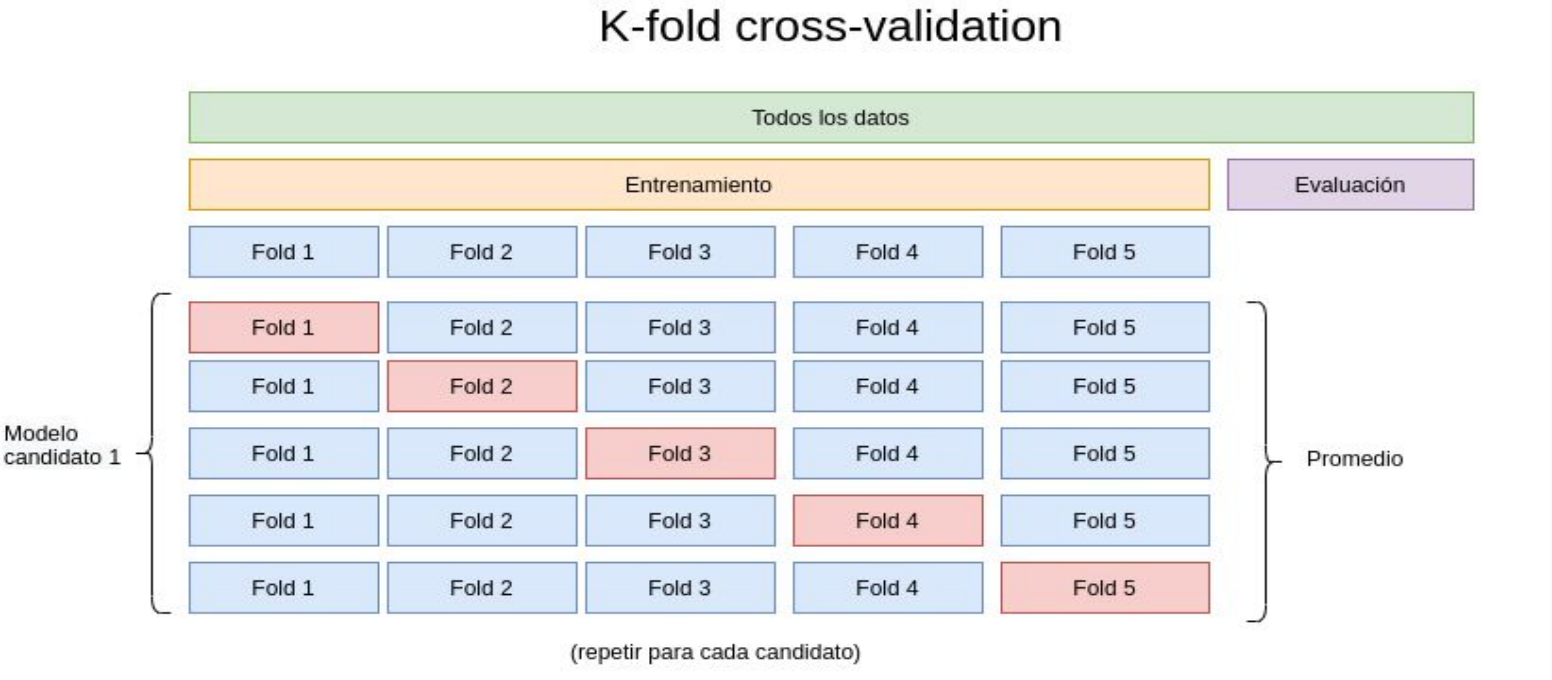


Preparación de datos y validación de resultados

- Al preparar datos para el entrenamiento de modelos es importante **calcular los parámetros de las transformaciones solo sobre el set de entrenamiento** y **nunca** sobre los datos de evaluación.
- En SKLearn típicamente utilizaremos el siguiente flujo de trabajo:
  - Durante el entrenamiento/validación cruzada:
    - Calcular parámetros óptimos en train set con *fit()/fit_transform()*.
    - Exportar estos parámetros (por ejemplo, con pickle).
  - Durante la evaluación/producción (datos no vistos):
    - Importar los parámetros y aplicarlos antes de realizar inferencias.
    - En algunos casos, aplicar una transformación inversa con *inverse_transform()* sobre el resultado de la inferencia.

Preparación de datos y validación de resultados. Validación cruzada.

Preparación de datos y validación de resultados. K-fold cross validation.



Ejemplos en jupyter

Clase 4.1 - Preparación de datos - Esquemas de validación.ipynb

2. Caracterización de las variables

Dataset (para un problema de aprendizaje supervisado)

- Llamamos ***dataset*** a un conjunto de ***m*** observaciones, cada una de ellas conformada por ***n*** variables de entrada y, en el caso de un problema de aprendizaje supervisado, ***r variables objetivo*** (target). Para los ejemplos de esta clase ***r=1***.

Variables de entrada

Variable(s) objetivo

Observaciones

$$\begin{bmatrix} X_0^{(0)} & X_1^{(0)} & X_2^{(0)} & \dots & X_{n\text{-}1}^{(0)} & y_0 \\ X_0^{(1)} & X_1^{(1)} & X_2^{(1)} & \dots & X_{n\text{-}1}^{(1)} & y_1 \\ \dots & & & & & \\ X_0^{(m-1)} & X_1^{(m-1)} & X_2^{(m-1)} & \dots & X_{n\text{-}1}^{(m-1)} & y_{m-1} \end{bmatrix}$$

¿Qué es una variable?

Una **variable** es una característica que puede fluctuar y cuya variación es susceptible a adoptar diferentes valores, los cuales pueden medirse u observarse.

Ejemplos de variables

- Edad (18, 23,70,...)
- Género (masculino, femenino)
- Ingreso ($40.000, $50.0000, …)
- País de nacimiento (Argentina, Perú, México)
- Color de ojos (marrón, verde)
- Vehículo (Ford, Volkswagen)

Tipos de variables

- Numéricas
  - Discretas
  - Continuas
- Categóricas
  - Nominales
  - Ordinales
- Fecha/hora
- Compuestas

¿Qué aspectos de las variables es importante considerar para un modelo de aprendizaje supervisado?

1. Datos faltantes.
2. Variables categóricas. Cardinalidad y etiquetas raras.
3. Cumplimiento de los supuestos de linealidad.
4. Distribución de los valores de las independientes.
5. Valores extremos (outliers).
6. Magnitud/escala.

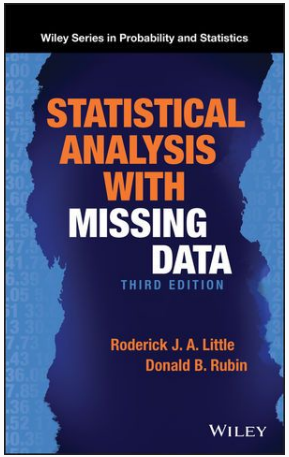
2.1 Datos faltantes. Definición.

- Los datos o valores faltantes ocurren cuando una observación está incompleta.
- Tienen un efecto en las conclusiones que puedan establecerse a partir de esos datos.

ID	Color	Weight	Broken	Class
1	Black	80	Yes	1
2	Yellow	100	No	2
3	Yellow	120	Yes	2
4	Blue	90	No	2
5	Blue	85	No	2
6	**?**	60	No	1
7	Yellow	100	**?**	2
8	**?**	40	**?**	1

2.1 Datos faltantes. Clasificación.

- Entender el motivo de ausencia de los datos permite seleccionar una estrategia adecuada para su tratamiento.
- Clasificación de Rubin de valores faltantes (1976):



  - **Missing Completely at Random (MCAR)**: significa que la razón por la cuál el dato no está es completamente aleatoria, y que probablemente no podamos predecir el valor a partir de otro valor en los datos.
  - **Missing at Random (MAR)**: los datos faltantes pueden ser explicados por valores en las otras columnas, pero no por valores de esa columna. Por ejemplo, si cada columna representa una elección excluyente.
  - **Missing not at Random (MNAR)**: significa que es probable que la falta de ese dato no sea al azar. En este caso tenemos que investigar la causa por la que falta ese valor.

2.1.1 Missing Data Completely at Random (MCAR)

- La probabilidad de valores faltantes es la misma para todas las observaciones.
- No existe ninguna relación entre los datos faltantes y otros valores observados o faltantes en el dataset.
- Omitir estas observaciones no implicaría un sesgo en las inferencias.

2.1.2 Missing Data at Random (MAR)

- La probabilidad de datos faltantes en una observación no está relacionada con los datos faltantes, pero puede depender de la información que sí está disponible. Es decir, el azar no es la única causa por la que faltan esos datos.
- Ejemplo:
    - En un estudio para un nuevo tratamiento, algunos sujetos se retiran cuando empiezan a tener efectos secundarios.

2.1.3 Missing Data not at Random (MNAR)

- Existe una explicación por la cual hay valores faltantes en un dataset.
- Ejemplos:
    - En una encuesta sobre salarios, los participantes con menor nivel educativo se ven menos inclinados a reportar sus ingresos.
    - En un estudio sobre depresión, los pacientes con depresión son menos propensos a responder algunas preguntas.

2.2 Cardinalidad. Definición.

- Los valores que puede tomar una **variable categórica** forman un grupo de **categorías** (también llamadas etiquetas).
- Se denomina **cardinalidad** al número de categorías existentes.

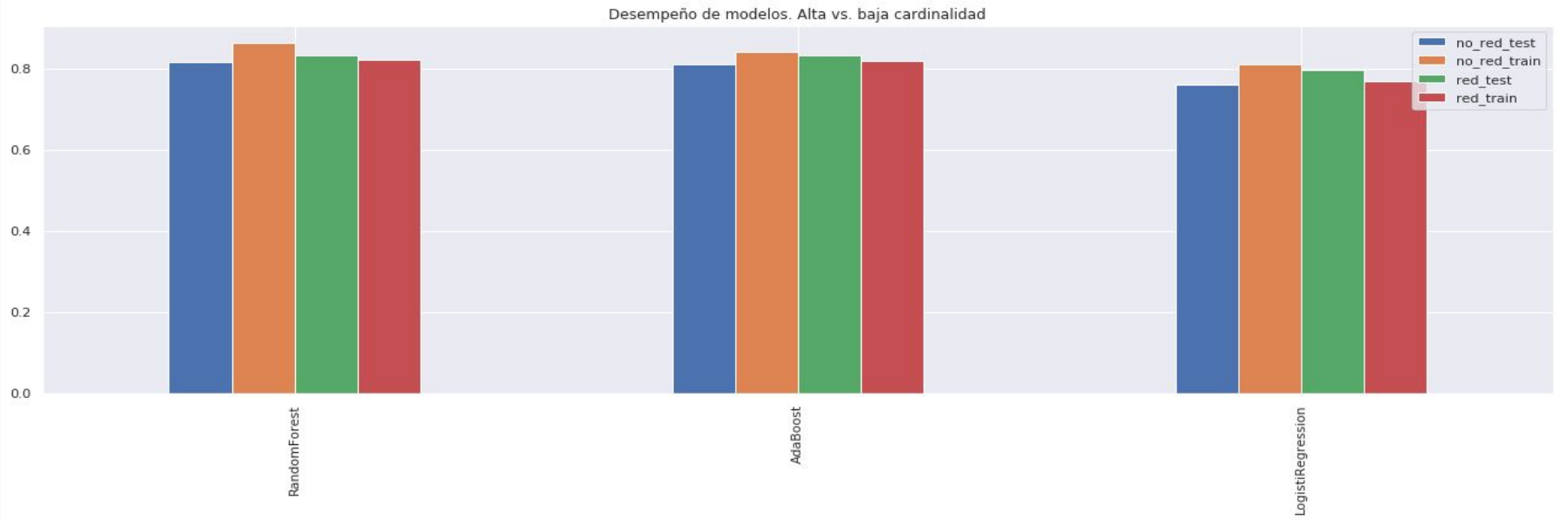
2.2 Cardinalidad. Problemas.

- La mayoría de los modelos de aprendizaje automático no aceptan cadenas de texto como entradas, por lo tanto, ***las categorías deben ser codificadas numéricamente***.
- Las técnicas de codificación pueden tener un efecto secundario indeseado, como aumentar la dimensión del espacio de las variables de entrada.
- Pueden ocurrir errores en la partición de entrenamiento y validación:
  - Valores que sólo estén disponibles en train set → overfitting.
  - Valores que sólo estén disponibles en test set → el modelo no podrá interpretarlos.

2.2 Cardinalidad. Overfitting.

- Las variables con muchas etiquetas dominarán sobre las que tengan menos etiquetas, especialmente en los algoritmos basados en árboles de decisión.
- Un número grande de etiquetas empeora la relación señal/ruido.
- Reducir la cardinalidad puede contribuir a mejorar el desempeño de algunos modelos.

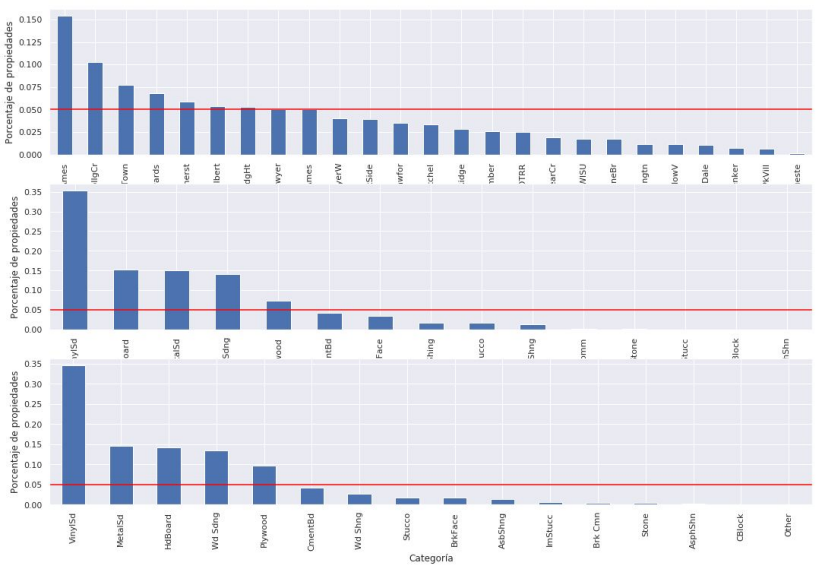
2.2 Cardinalidad. Overfitting.



Entrenamiento de modelos con dos versiones del dataset (original vs con reducción de categorías) (ejemplo completo en notebook)

2.3 Etiquetas poco frecuentes

- Son aquellas que aparecen en una pequeña proporción de observaciones de todo el dataset.
- Por ejemplo en un censo nacional, si la variable es “ciudad de residencia”:
  - “Buenos Aires” puede ser un valor muy frecuente
  - “Faro” probablemente será una categoría poco frecuente (tiene ~14 habitantes) (*).



Categorías de propiedades (ejemplo completo en notebook)

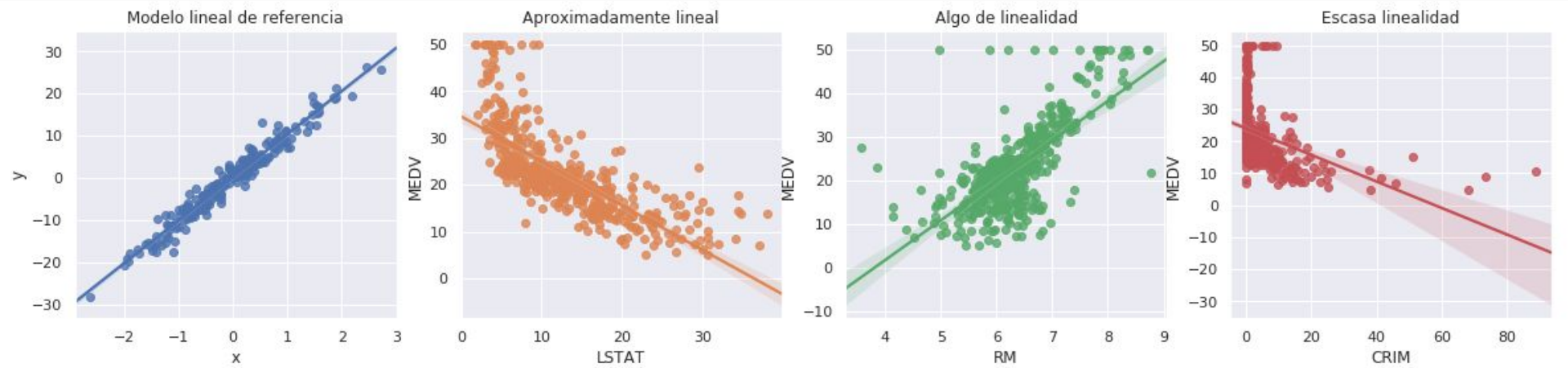
(*) https://es.wikipedia.org/wiki/Faro_(Buenos_Aires)

2.4 Supuestos de linealidad

- Los modelos lineales(*) realizan los siguientes supuestos sobre las variables independientes:
  - Existe una relación lineal entre variables de entrada y de salida.
  - Normalidad.
  - Ausencia total de colinealidad, o muy baja.
  - Homocedasticidad (homogeneidad de la varianza).

(*) Ejemplos: regresión lineal, regresión logística, PCA.

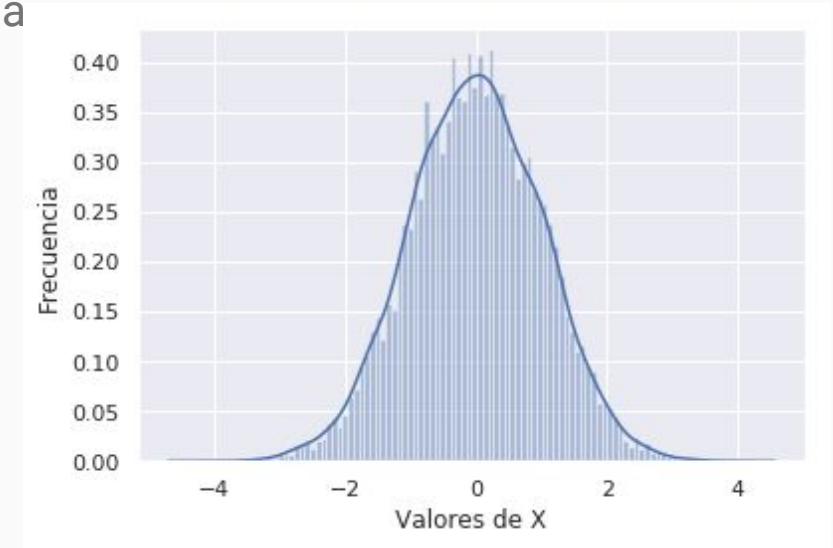
Supuestos de linealidad. Relación lineal.



- y ~= b0 + b1X1 + b2X2 + … bnXn
- Un método de verificación es mediante scatter plots.
- A veces, aplicar transformaciones no lineales a las variables puede mejorar la relación lineal.

Supuestos de linealidad. Normalidad.

- Las variables independientes obedecen una distribución gaussiana.
- Puede evaluarse con gráficos Q-Q.
- También puede ser evaluada con tests estadísticos, como el test de Kolmogorov-Smirnov.
- A veces, cuando una variable no tiene una distribución normal, puede aplicarse una transformación (por ejemplo log) para asemejarse a una distribución normal.

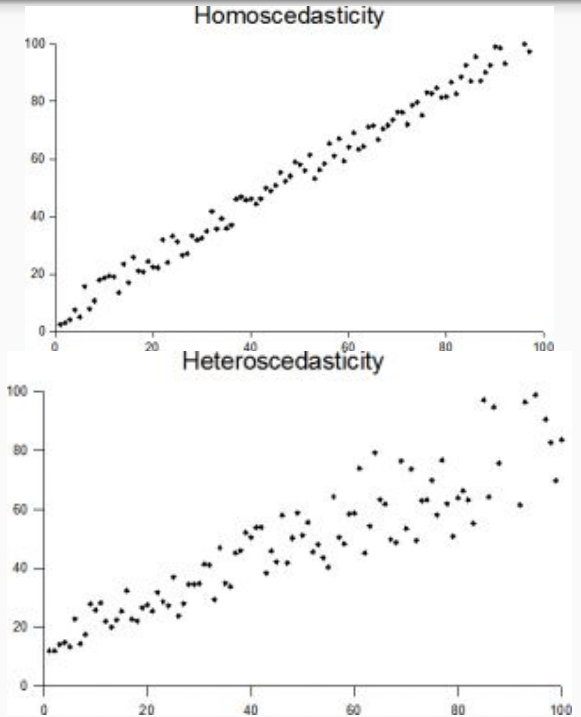


2.4.3 Supuestos lineales. Ausencia de colinealidad.

- La multi-colinealidad ocurre cuando las variables independientes están correlacionadas entre sí.
- Puede evaluarse con una matriz de correlación.

2.4.4 Supuestos de linealidad. Homocedasticidad (homogeneidad de la varianza)

- Las variables independientes tienen la misma varianza finita.
- También conocida como homogeneidad de la varianza.
- Existen gráficos y tests que permiten determinarla, por ejemplo:
  - Gráfico de residuales.
  - Test de Levene.
  - Test de Barlett.
  - Test de Goldfeld-Quandt.
- A veces, la homogeneidad de la varianza puede mejorarse aplicando transformaciones no lineales.



2.5 Distribución

- En la segunda clase se presentaron distribuciones:
  - Normales:
    - La media, mediana y moda coinciden.
  - Con oblicuidad
    - La media está influenciada por la cola.
- Los modelos lineales asumen que las variables independientes obedecen a una distribución normal.
- Otros modelos no realizan una suposición de cómo están distribuidas las variables, pero a veces una mejor distribución puede mejorar su desempeño.

2.5 Distribución. Normalización mediante transformación y discretización.

- Se estudiarán dos grupos de métodos para llevar una distribución a una forma normal:
  - Transformaciones
    - ln(x)
    - exp(x)
    - 1/x
    - Box-Cox, Yeo Johnson
  - Discretización
    - Bins de frecuencia fija (suelen mejorar la distribución)
    - Bins de ancho fijo (por lo general no mejoran la distribución)

2.6 Valores extremos

- Un valor extremo es un punto significativamente diferente del resto de los datos.

  “An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.” [D. Hawkins. Identification of Outliers, Chapman and Hall , 1980.]

Valores extremos

- En algunos casos, como detección de anomalías, el mayor interés está en la identificación de los valores extremos.
- En esta clase, sin embargo, nos enfocaremos en cómo tratarlos para mejorar el desempeño general de un modelo regresivo o de clasificación.

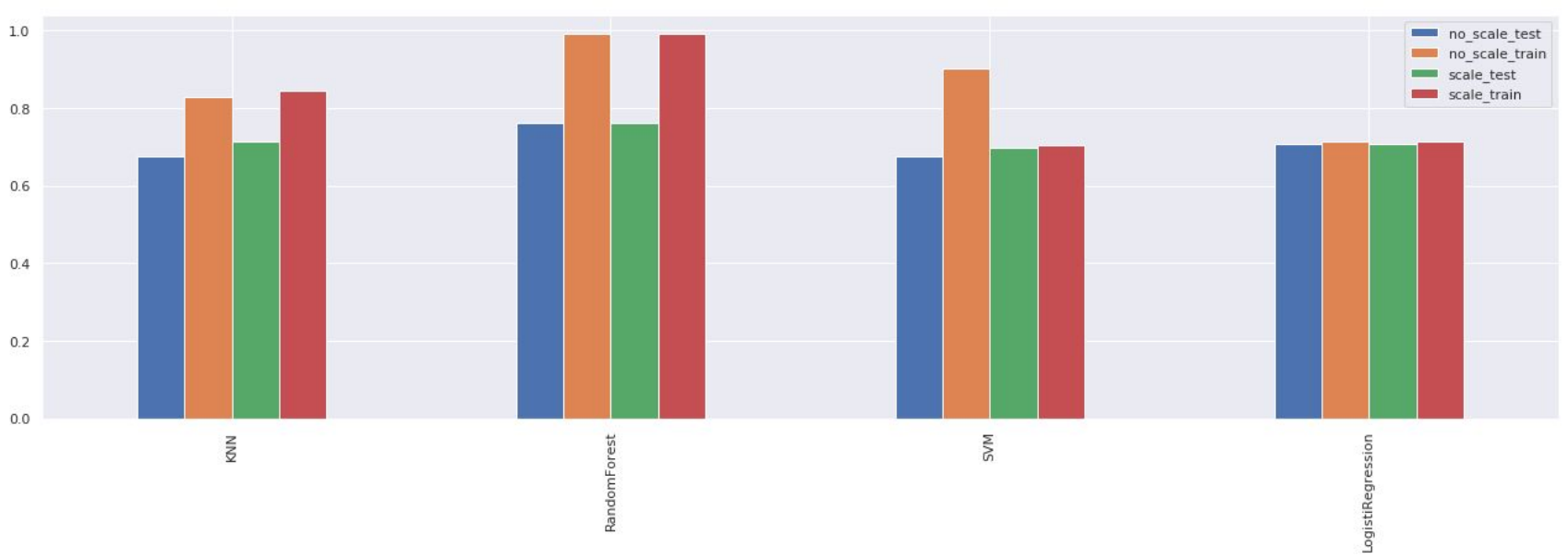
2.6 Magnitud de las variables de entrada.

- Los coeficientes de regresión dependen de la magnitud de la variable.
- Las variables de entrada con mayor magnitud pueden dominar a las de menor magnitud.

Magnitud de las variables de entrada.

- Algoritmos afectados
  - Regresión lineal y logística.
  - Redes neuronales.
  - Support Vector Machines.
  - KNN.
  - K-means.
  - Principal Component Analysis (PCA).
- No afectados (por ejemplo, los basados en árboles)
  - Árboles de clasificación y regresión.
  - Random Forest
  - Gradient Boosted Trees.

Magnitud de las variables de entrada.



Efecto del escalado en distintos modelos (ejemplo completo en notebook)

Ejemplos en jupyter + recursos de interés

Clase 4.2 - Preparación de datos - Caracterización de variables.ipynb

QQPlots (visualización interactiva): https://xiongge.shinyapps.io/QQplots/

3. Imputación de datos faltantes

Imputación de datos faltantes

- La imputación es el acto de reemplazar datos faltantes con estimaciones estadísticas de los valores ausentes.
- El objetivo de cualquier técnica de imputación es producir un dataset completo que permita entrenar un modelo de aprendizaje automático.
- Pueden agruparse las técnicas de imputación en dos categorías:
  - **Univariada**: cuando la imputación se realiza de manera independiente para cada variable de entrada, sin considerar las otras variables.
  - **Multivariada**: cuando el valor imputado para cada variable es una función de dos o más variables.

Técnicas de imputación univariada

- Algunas técnicas de imputación univariada (por tipo de variable al que aplican):
  - **Variables numéricas:**
    - Imputación por promedio/mediana.
    - Imputación por valor arbitrario.
    - Imputación de “fin de cola” (en of tail).
  - **Variables categóricas:**
    - Imputación por categoría frecuente.
    - Agregar categoría “FALTANTE”.
  - **Ambas:**
    - Análisis de caso completo.
    - Agregar indicador “FALTANTE”.
    - Imputación por muestreo aleatorio.

Análisis de caso completo (CCA)

- CCA (por sus siglas en inglés: *Complete Case Analysis*) consiste en descartar todas las observaciones en las que cualquiera de las variables están ausentes.
- Por lo tanto, se analizan sólo las observaciones para las cuales están todos los datos disponibles.

Supuestos de CCA

- MCAR (Missing completely at random)

Ventajas de CCA

- Simple.
- No se requiere manipulación previa de los datos.
- Preserva la distribución de las variables.

Limitaciones de CCA

- Puede excluir una gran parte del dataset original.
- Las observaciones excluidas pueden ser informativas para el análisis (si no se cumple el supuesto MCAR, por ejemplo).
- CCA creará un dataset sesgado si los casos completos difieren de aquellos en los que falta información (MAR/NMAR).
- El modelo puesto en producción no podrá manejar observaciones incompletas.

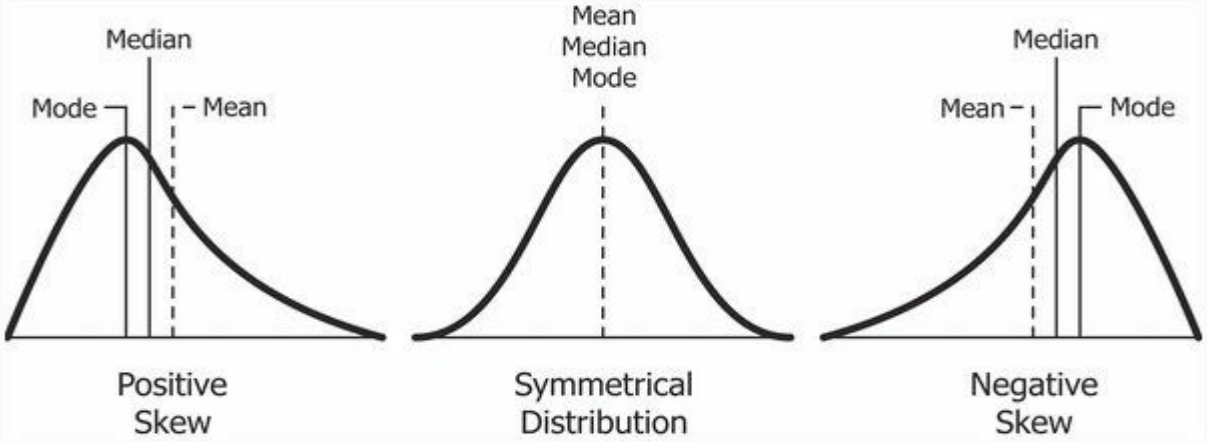
¿Cuando aplicar CCA?

- Se cumple el supuesto MCAR.
- Las observaciones a eliminar no superan el 5% del total del dataset.

Imputación por media o mediana

- Consiste en reemplazar las ocurrencias de valores faltantes por la media o la mediana de esa variable.
- Si la variable está normalmente distribuida la media y la mediana son similares.
- Si en cambio, la distribución tiene oblicuidad, la mediana es una mejor representación de los datos faltantes.

Imputación por media o mediana



- Si la variable está normalmente distribuida la media y la mediana son similares.
- Si en cambio, la distribución tiene oblicuidad, la mediana es una mejor representación de los datos faltantes.

Imputación por media o mediana. Supuestos.

- MCAR, MAR.
- Las observaciones faltantes se asemejan a la mayoría (por eso podemos reemplazarlas por la media o la mediana).

Imputación por media o mediana. Ventajas.

- Fácil de implementar.
- Se puede integrar en producción.

Imputación por media o mediana. Limitaciones.

- Distorsiona la distribución original.
- Distorsiona la varianza original.
- Distorsiona la covarianza con las variables restantes.
- A mayor cantidad de datos faltantes, mayor distorsión.

Imputación por media o mediana. ¿Cuando aplicar?.

- Se cumple el supuesto MCAR, MAR.
- No más del 5% de observaciones incompletas del total del dataset.
- Por lo general, se utiliza agregando una variable binaria indicando si ese valor estaba ausente.

Imputación por valor arbitrario. Definición

- Consiste en reemplazar las ocurrencias de los valores faltantes (NA) por una variable de un valor arbitrario.
- Los valores típicos son:
  - En distribuciones numéricas: 0, 999, -999 o -1 (para distribuciones positivas).
  - En variables categóricas: “MISSING”/”FALTANTE”.

Imputación por valor arbitrario. Supuestos

- MNAR.
- En este caso, sospechamos que existe una razón por la cual estos valores están ausentes, por lo cual la media o mediana no necesariamente son representativos.

Imputación por valor arbitrario. Ventajas.

- Fácil de implementar.
- Método rápido para obtener datasets completos.
- Puede integrarse en producción.
- Captura la importancia de los datos “faltantes”, en caso de haberla.

Imputación por valor arbitrario. Limitaciones.

- Distorsiona la distribución original de los datos.
- Distorsiona la varianza original.
- Distorsiona la covarianza con las variables restantes.
- Si el valor arbitrario está al final de la distribución, puede introducir outliers.
- Se debe tener cuidado de no elegir un valor arbitrario demasiado similar a la media o mediana.
- Cuanto mayor porcentaje de NA, mayor la distorsión.

Imputación por valor arbitrario. Cuando aplicar.

- Se cumple MNAR.

Imputación de “fin de cola”. Definición

- Es equivalente a la imputación por valor arbitrario, pero en este caso automáticamente se seleccionan valores arbitrarios al final de la distribución.
- Si la variable tiene una distribución normal se puede utilizar el promedio +- 3 veces el desvío estándar.
- Si la variable tiene una distribución con oblicuidad, se puede utilizar la regla de proximidad IQR.
- Sólo aplica para tipos numéricos.

Imputación por categoría frecuente o moda. Definición.

- La imputación por moda o categoría frecuente consiste en reemplazar los valores faltantes (NA) por la moda o valor más frecuente.
- Por lo general, esta técnica se utiliza para variables categóricas.

Imputación por categoría frecuente o moda. Supuestos.

- MCAR o MAR.
- Las observaciones faltantes se asemejan a la mayoría.

Imputación por categoría frecuente o moda. Ventajas.

- Fácil de implementar.
- Método rápido para obtener datasets completos.
- Puede integrarse en producción.

Imputación por categoría frecuente o moda. Limitaciones.

- Distorsiona la relación entre la etiqueta más frecuente con otras variables del dataset.
- Puede exagerar la presencia de la etiqueta más frecuente si el número de NAs es alto.
- A mayor cantidad de NA, mayor distorsión.

Imputación por categoría frecuente o moda. Cuando aplicar.

- MCAR.
- No más del 5% del total de observaciones faltantes.

Imputación por categoría faltante. Definición.

- Este método consiste en crear una categoría adicional (típicamente “MISSING” o “FALTANTE”) y asignarla a los valores con NA.
- Es el método más utilizado para imputación categórica.

Imputación por categoría faltante. Ventajas.

- Fácil de implementar.
- Método rápido para obtener datasets completos.
- Puede integrarse en producción.
- Captura la importancia de los tipos faltantes, en caso de haberla.
- No realiza supuestos sobre los datos.

Imputación por categoría faltante. Limitaciones.

- Si la cantidad de NA es baja, introduce un nuevo problema: agrega una nueva variable poco frecuente.

Imputación por muestreo aleatorio. Definición.

- Consiste en tomar observaciones aleatorias de las disponibles, y utilizarlas para completar los valores ausentes.
- Este método es válido tanto para variables numéricas como categóricas.

Imputación por muestreo aleatorio. Supuestos.

- MCAR.
- Se espera que conservar la distribución de la variable original.

Imputación por muestreo aleatorio. Ventajas.

- Fácil de implementar.
- Método rápido para obtener un dataset completo.
- Puede integrarse en producción.
- Preserva la varianza de la variable original.

Imputación por muestreo aleatorio. Limitaciones.

- Aleatoriedad.
- La relación entre las variables imputadas con otras variables puede verse afectada si el número de NAs es grande.
- Si bien es apto para producción, implica disponer del dataset original para extraer y reemplazar valores NA en las nuevas observaciones.

Imputación multivariada.

- Las técnicas anteriores reemplazan los valores faltantes de una observación sin utilizar información del resto de las variables.
- Dos técnicas de imputación multivariada:
  - MICE (Multiple Imputation by Chained Equations)
  - Imputación por vecinos cercanos (KNN)

Imputación multivariada. MICE: Multiple imputation by chained equations

- Es una de las técnicas de imputación multivariada más utilizada.
- Consiste en calcular valores faltantes para cada columna como una regresión lineal de las restantes (si bien también se pueden usar otros modelos).
- Es iterativo porque repite este proceso hasta que la diferencia entre las dos últimas iteraciones sea cercana a cero (varianza estable).
- Referencias:
  - Publicación original (2011): https://www.jstatsoft.org/article/view/v045i03
  - Multiple imputation by chained equations: what is it and how does it work? https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3074241/

MICE. Algoritmo. Ejemplo

- Un banco tiene una base de datos de clientes que compraron una tarjeta de crédito.
- Quiere desarrollar un modelo para saber si un cliente es un potencial comprador o no.
- Cada observación representa un cliente, y las variables observadas son:
  - Edad
  - Experiencia
  - Salario
- La variable a predecir para cada cliente es si decidió comprar la tarjeta (0=No,1=Sí).

	age	experience	salary	purchased
0	25.0	NaN	50.0	0
1	27.0	3.0	NaN	1
2	29.0	5.0	110.0	1
3	31.0	7.0	140.0	0
4	33.0	9.0	170.0	1
5	NaN	11.0	200.0	0

MICE. Algoritmo. Ejemplo

- Supondremos, por un momento que, disponemos de los datos originales.
- ¿Si imputáramos por promedio, esa imputación sería representativa?

	age	experience	salary	purchased
0	25	1	50	0
1	27	3	80	1
2	29	5	110	1
3	31	7	140	0
4	33	9	170	1
5	35	11	200	0

Datos originales.

	age	experience	salary
0	25.0	7.0	50.0
1	27.0	3.0	134.0
2	29.0	5.0	110.0
3	31.0	7.0	140.0
4	33.0	9.0	170.0
5	29.0	11.0	200.0

Imputación por media.

MICE. Algoritmo. Ejemplo

- Paso 1. Construir un dataset inicial, que llamaremos “0”, realizando una imputación estadística (por promedio o mediana).

	age	experience	salary
0	25.0	**7.0**	50.0
1	27.0	3.0	134.0
2	29.0	5.0	110.0
3	31.0	7.0	140.0
4	33.0	9.0	170.0
5	**29.0**	11.0	200.0

Dataset 0

MICE. Algoritmo. Ejemplo

- Paso 2. Crear un Dataset “1” completando los datos faltantes de cada columna a partir de un modelo regresivo sobre las columnas restantes.

	age	experience	salary
0	25.0	7.0	50.0
1	27.0	3.0	134.0
2	29.0	5.0	110.0
3	31.0	7.0	140.0
4	33.0	9.0	170.0
5	?	11.0	200.0

Dataset 1

```
def impute_column(df, col_to_predict, feature_columns):
        """ Imputar valores faltantes de una columna a partir de un
            modelo LR sobre columnas restantes
        """
        # Establecer en NaN los valores a predecir
        nan_rows = np.where(np.isnan(df[col_to_predict]))
        all_rows = np.arange(0,len(df))

        # Separar datos de entrenamiento de datos para predicciones
        train_rows_idx = np.argwhere(~np.isin(all_rows,nan_rows)).ravel()
        pred_rows_idx =  np.argwhere(np.isin(all_rows,nan_rows)).ravel()
         X_train,y_train = df[feature_columns].iloc[train_rows_idx],
       df[col_to_predict].iloc[train_rows_idx]

        # Predecir
        X_pred = df[feature_columns].iloc[pred_rows_idx]
        model = LinearRegression()
        model.fit(X_train,y_train)
        df[col_to_predict].iloc[pred_rows_idx] = model.predict(X_pred.values.reshape(1,-1))
    return df
```

MICE. Algoritmo. Ejemplo

- Paso 3. Calcular la diferencia entre los dos datasets anteriores. Repetir hasta la diferencia sea próxima a cero, o haber alcanzado una cantidad de iteraciones.

	age	experience	salary
0	25.000000	1.853863	50.00000
1	27.000000	3.000000	72.77481
2	29.000000	5.000000	110.00000
3	31.000000	7.000000	140.00000
4	33.000000	9.000000	170.00000
5	36.253165	11.000000	200.00000

Dataset 1

−

	age	experience	salary
0	25.0	7.0	50.0
1	27.0	3.0	134.0
2	29.0	5.0	110.0
3	31.0	7.0	140.0
4	33.0	9.0	170.0
5	29.0	11.0	200.0

Dataset 0

=

	age	experience	salary
0	0.000000	-5.146137	0.00000
1	0.000000	0.000000	-61.22519
2	0.000000	0.000000	0.00000
3	0.000000	0.000000	0.00000
4	0.000000	0.000000	0.00000
5	7.253165	0.000000	0.00000

Diferencia

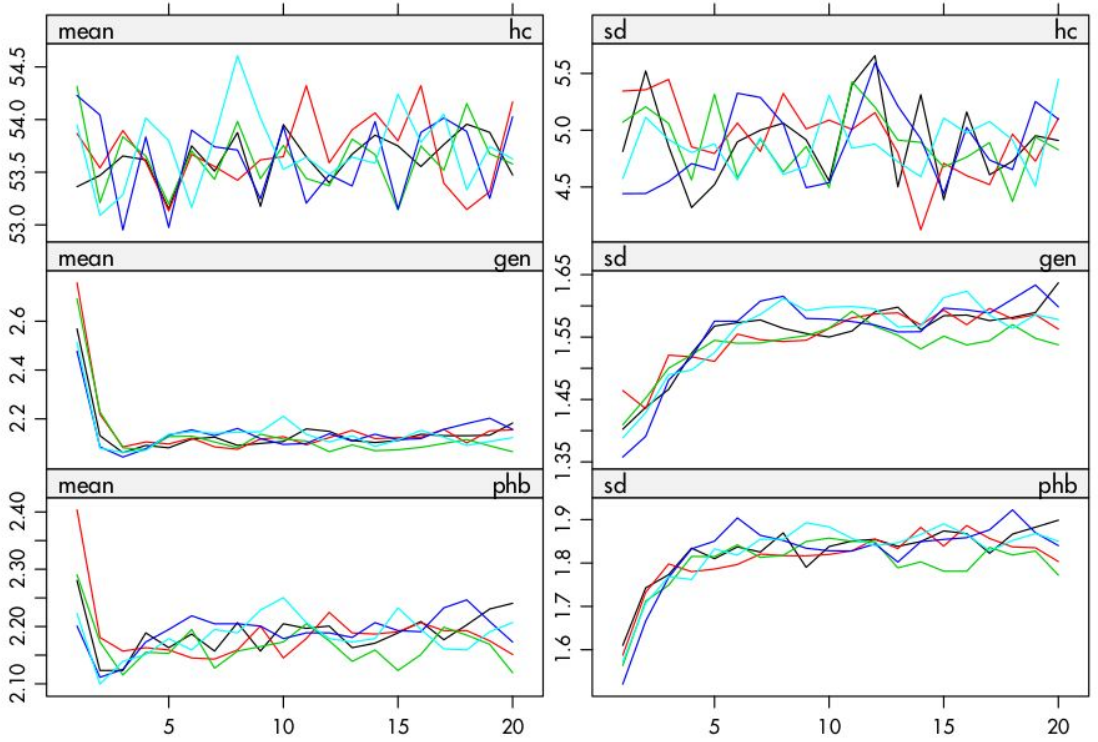
4.3. Assessing convergence

There is no clear-cut method for determining whether the MICE algorithm has converged. What is often done is to plot one or more parameters against the iteration number. The `mice()` function produces m parallel imputation streams. The `mids` object contains components `chainMean` and `chainVar` with the mean and variance of the imputations per stream, respectively. These can be plotted by the `plot.mids` object. On convergence, the different streams should be freely intermingled with each other, without showing any definite trends. Convergence is diagnosed when the variance between different sequences is no larger than the variance with each individual sequence.

Inspection of the stream may reveal particular problems of the imputation model. Section 3.4 shows that passive imputation should carefully set the predictor matrix. The code below is a pathological example where the MICE algorithm is stuck at the initial imputation.

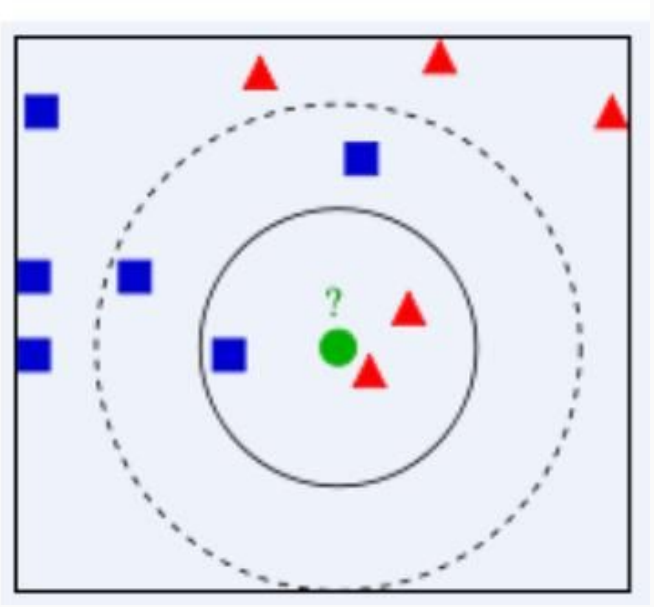
MICE. Algoritmo. Convergencia

Figure 9: Healthy convergence of the MICE algorithm for `hgt`, `wgt` and `bmi`, where feedback loop of `bmi` into `hgt` and `wgt` is broken (solution `imp.idx`).



Imputación multivariada. K-Nearest Neighbours

- Consiste en utilizar el algoritmo KNN para estimar los valores faltantes por semejanza a los más próximos.
- Se suele utilizar con una función de distancia que contemple la presencia de NaNs, como por ejemplo ***nan_euclidean_distance***.
- Al igual que en KNN, el número de vecinos suele ser un número impar.



Imputación multivariada. K-Nearest Neighbours. Ejemplo

```
from sklearn.metrics.pairwise import nan_euclidean_distances
import numpy as np
#dist(x,y) = sqrt(weight * sq. distance from present coordinates)
#where, weight = Total # of coordinates / # of present coordinates

row_to_impute = df.iloc[2]
feature_cols = ['Metallica', 'Luis Miguel', 'Astor Piazzolla', 'Jimi Hendrix']
music_df['nan_euc_dist'] = music_df[feature_cols].apply(
    lambda row: nan_euclidean_distances( row_to_impute.values.reshape(1,-1),
            row.values.reshape(1,-1))[0][0],axis=1)
music_df
```

	Metallica	Luis Miguel	Astor Piazzolla	Jimi Hendrix	nan_euc_dist
0	80.0	30.0	7	27.0	120.461612
1	44.0	NaN	10	29.0	121.155547
2	NaN	85.0	25	88.0	168.103143
3	50.0	70.0	74	49.0	90.917545
4	29.0	54.0	49	NaN	90.347477

39.5

```
(50+29)/2
```

```
39.5
```

(ejemplo para dos vecinos cercanos)

Imputación multivariada. K-Nearest Neighbours

- Demo interactiva: http://vision.stanford.edu/teaching/cs231n-demos/knn/
- nan_euclidean_distance: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.nan_euclidean_distances.html

Ejemplos en jupyter

Clase 4.3 - Preparación de datos - Imputación de datos faltantes.ipynb

4. Codificación de variables categóricas

Codificación de variables categóricas.

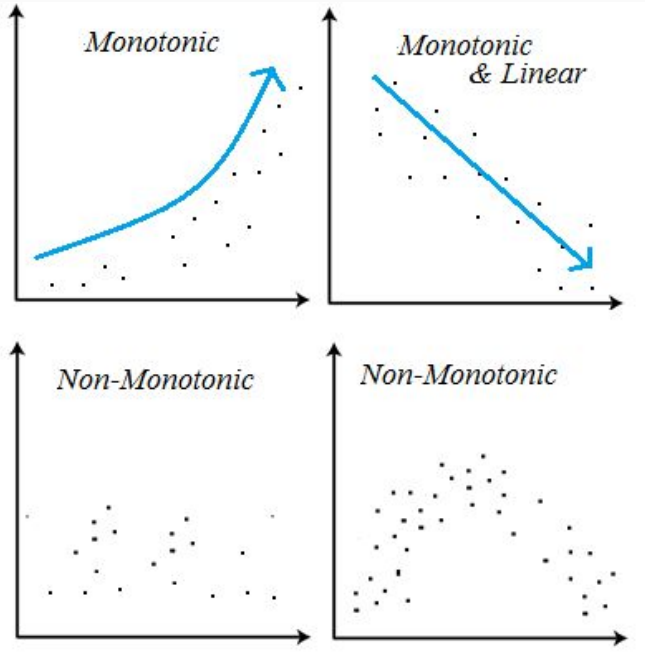
- Consiste en reemplazar una categoría (típicamente representada en forma de texto) por una representación numérica.
- El objetivo es disponer de variables que puedan ser utilizadas en los modelos de AA.

Codificación de variables categóricas. Técnicas.

Tradicionales	• One Hot Encoding • Count/frequency encoding • Ordinal/label encoding
Relación monotónica	• Label encoding ordenado. • Encoding por promedio • Peso de la evidencia (WoE)
Alternativas	• Binary encoding • Feature hashing • Otros

Relación monotónica

- Decimos que existe una relación monotónica entre una variable independiente X e una variable objetivo Y cuando:
  - Si se incrementa el valor de X también se incrementa el valor de Y ó
  - Si se incrementa el valor de X, Y disminuye.



Relación monotónica. Importancia.

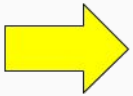
- Las relaciones monotónicas entre variables de entrada y de salida pueden mejorar el desempeño de los modelos lineales.
- En los modelos de árboles, puede traducirse en menor profundidad (sin comprometer su desempeño).
- A menudo, estas relaciones aparecen de manera natural en los datos. Por ejemplo: las primas de los seguros suelen disminuir a medida que aumenta la edad de los asegurados.
- Algunas de las formas de codificación que se mostrarán, estarán orientadas intentar obtener esta relación entre la variable transformada y la variable objetivo.

One Hot Encoding

- Consiste en codificar cada valor de una variable categórica con un conjunto de variables booleanas que pueden tomar 0 o 1, indicando si esa categoría está o no presente en cada observación.
- Si la variable tiene k-valores posibles, puede codificarse utilizando k o k-1 nuevas variables (en el último caso se suele llamar dummy encoding).

Color
Red
Red
Yellow
Green
Yellow

OHE



Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

DE

	Travel_Class_2	Travel_Class_3
Passenger 1	0	0
Passenger 2	1	0
Passenger 3	0	1
Passenger 4	0	0

One Hot Encoding. k o k-1?

- Codificar utilizando k-1 variables disminuye el costo adicional de creación de variables (tener presente que es común realizar entrenamientos sobre el dataset completo).
- En algunos modelos que tienen un término de sesgo (bias), como por ejemplo regresión lineal, la presencia de una matriz OHE con k variables haría que la matriz sea singular y por lo tanto no invertible, imposibilitando la solución por fórmula cerrada.
- No obstante, para los siguientes escenarios se aconseja utilizar k variables:
  - Algoritmos basados en árboles.
  - Feature selection por algoritmos recursivos.
  - Para determinar la importancia de cada categoría.

One Hot Encoding. Ventajas.

- No realiza ningún supuesto sobre la distribución de las categorías de la variable.
- Mantiene toda la información de la variable categórica.
- Es apta para modelos lineales.

One Hot Encoding. Limitaciones.

- Aumenta la dimensión del espacio de variables de entrada.
- No agrega información.
- Introduce muchas variables con información redundante.

One Hot Encoding. Variante para top-n categorías.

- Una variante de OHE es utilizarla sólo para las top n categorías más frecuentes.
- Esta técnica fue la solución ganadora en KDD 2009: http://www.mtome.com/Publications/CiML/CiML-v3-book.pdf

Label/integer encoding. Definición.

- Consiste en reemplazar las categorías por valores numéricos de 0 a N-1.
- Los números se asignan de manera arbitraria.

Label/integer encoding. Ventajas.

- Fácil de implementar, no expande la dimensión del espacio de variables de entrada.
- Puede funcionar bien con algoritmos basados en árboles.

Label/integer encoding. Limitaciones.

- La codificación no aporta información.
- No es apropiado para modelos lineales.
- En producción, no maneja automáticamente nuevas categorías.

Count/Frequency encoding. Definición.

- Cada categoría se reemplaza por el valor o porcentaje de observaciones en que aparece en el dataset.
- Se captura la representación de cada categoría.
- Muy utilizado en competencias de Kaggle.
- Se hace un supuesto importante: la cantidad de observaciones para cada categoría está relacionada con la variable a predecir.

Count/Frequency encoding. Ventajas.

- Fácil de implementar.
- No expande las dimensiones del espacio de variables de entrada.
- Puede tener un desempeño aceptable con modelos basados en árboles.

Count/Frequency encoding. Limitaciones.

- No apropiado para modelos lineales.
- No maneja automáticamente nuevas categorías en test set/producción.
- Si aparecen dos categorías la misma cantidad de veces, pueden ser reemplazadas por el mismo número, con la consecuente pérdida de información.

Ordinal encoding c/ orden. Definición

- Consiste en reemplazar las categorías por valores numéricos de 0 a N-1, pero en este caso el ordenamiento no es arbitrario.
- La asignación de cada entero asignado corresponde con el promedio de la variable objetivo de cada categoría.

Ordinal encoding c/ orden. Ventajas

- Fácil de implementar.
- No expande las dimensiones del espacio de variables de entrada.
- Crea una relación monotónica entre las categorías y la variable objetivo.

Ordinal encoding c/ orden. Limitaciones

- Puede introducir overfitting.
- Las librerías estándar como SkLearn no lo soportan directamente, por lo que no es directo su uso en un esquema de cross-validation o k-folds validation.

Mean encoding. Definición

- Consiste en reemplazar cada categoría por el promedio de la variable objetivo para esa categoría.

Mean encoding. Ventajas

- Fácil de implementar.
- No expande las dimensiones del espacio de características.
- Crea una relación monotónica entre las categorías y la variable objetivo.

Mean encoding. Limitaciones

- Puede introducir overfitting.
- Las librerías estándar como SkLearn no lo soportan directamente, por lo que no es directo su uso en un esquema de cross-validation o k-folds validation.
- Puede ocurrir un escenario en el que dos categorías tengan un promedio muy similar, con la consecuente pérdida de información.

Peso de Evidencia (Weight of Evidence). Definición.

$$WoE = ln\frac{P(X=1)}{P(X=0)}$$

- Esta técnica se originó para modelos financieros y riesgo crediticio.
- Consiste en reemplazar una variable binaria por el logaritmo natural del ratio del valor positivo respecto del valor negativo o por defecto (el orden puede invertirse dependiendo del caso, por ejemplo, en modelos financieros se suele usar p(0)/p(1)).

Peso de Evidencia (Weight of Evidence). Ventajas.

- Crea una relación monotónica entre la variable objetivo y las variables de entrada.
- Funciona muy bien con regresión logística, por la forma en que quedan ordenadas las categorías.
- Las variables transformadas pueden ser comparadas porque están en la misma escala, por lo tanto, es casi inmediato determinar cuál es más predictiva.

Peso de Evidencia (Weight of Evidence). Limitaciones.

- Puede llevar a overfitting.
- Indefinida cuando el denominador es cero.

Codificación de etiquetas poco frecuentes.

- Las etiquetas poco frecuentes aparecen en una proporción pequeña de las observaciones del dataset.
- Suelen presentarse en los siguientes escenarios:
    - Variables con una categoría predominante.
    - Variables con pocas categorías.
    - Variables con alta cardinalidad.
- La recomendación es agrupar todas las categorías poco frecuentes en una nueva categoría “raras”.

Binary encoding / feature hashing. Definición

- Binary encoding es una combinación de OHE y ordinal encoding. Se codifica cada variable de entrada como una composición de variables binarias.
- Feature hashing, aplica una función de hash a cada valor de entrada para obtener un entero.
- En ambos casos:
  - Ventaja → eficiencia en la representación
  - Desventaja → pérdida de interpretabilidad.

			Binary Encoded			
Categorical Feature		=	x1	x2	x4	x8
Louise	=>	1	1	0	0	0
Gabriel	=>	2	0	1	0	0
Emma	=>	3	1	1	0	0
Adam	=>	4	0	0	1	0
Alice	=>	5	1	0	1	0
Raphael	=>	6	0	1	1	0
Chloe	=>	7	1	1	1	0
Louis	=>	8	0	0	0	1
Jeanne	=>	9	1	0	0	1
Arthur	=>	10	0	1	0	1

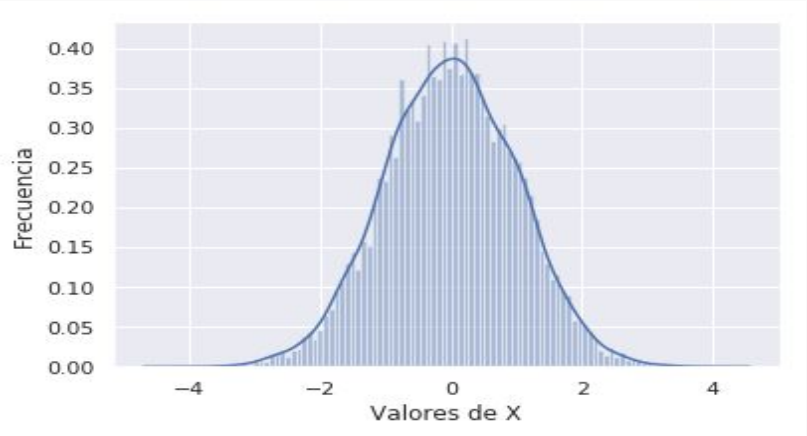
Ejemplos en jupyter

Clase 4.4 - Preparación de datos - Codificación de variables categóricas.ipynb

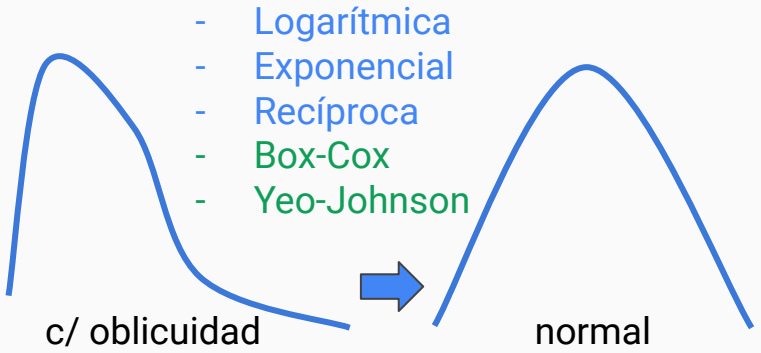
5. Transformación de variables

Distribución normal en modelos lineales

- Es deseable que los valores de cada variable independiente (X) tengan una distribución normal.



- Es muy común que esto no se cumpla en los datos originales. En este caso, puede intentar obtenerse una distribución más semejante a una normal luego de aplicar una transformación.



Transformaciones matemáticas

- Logarítmica: log(X), X>0
- Recíproca: 1/X, X!=0
- Potencia/exponencial
  - X exp(λ)
  - X^(½)/X^3
  - No definido para todo X.
- Exponencial (casos especiales):
  - Box-Cox, X>0
  - Yeo-Johnson

Transformación de Box Cox

- La transformación de Box-Cox estima un valor de lambda que minimiza la desviación estándar de una variable transformada estandarizada.
- El método de Box-Cox busca entre muchos tipos de transformaciones.

$$y_i^{(\lambda)} = \begin{cases} \dfrac{y_i^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0, \\ \ln(y_i) & \text{if } \lambda = 0, \end{cases}$$

L	Y'
-2	$Y^{-2} = 1/Y^2$
-1	$Y^{-1} = 1/Y^1$
-0.5	$Y^{-0.5} = 1/(\text{Sqrt}(Y))$
0	log(Y)
0.5	$Y^{0.5} = \text{Sqrt}(Y)$
1	$Y^1 = Y$
2	Y^2

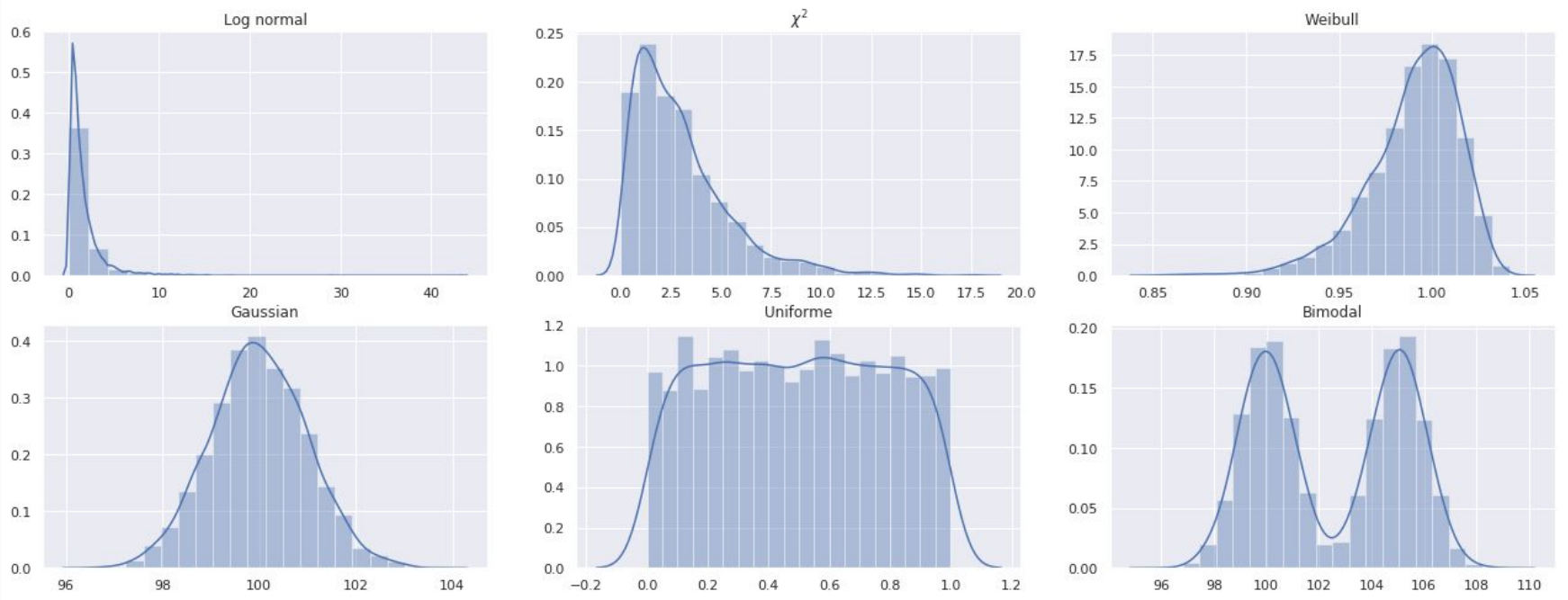
Source: Box and Cox (1964). Where, 'Y' is the transformation of t
Note that for Lambda = 0, the transformation is NOT Y^0 (because t
every

Transformación de Yeo-Johnson

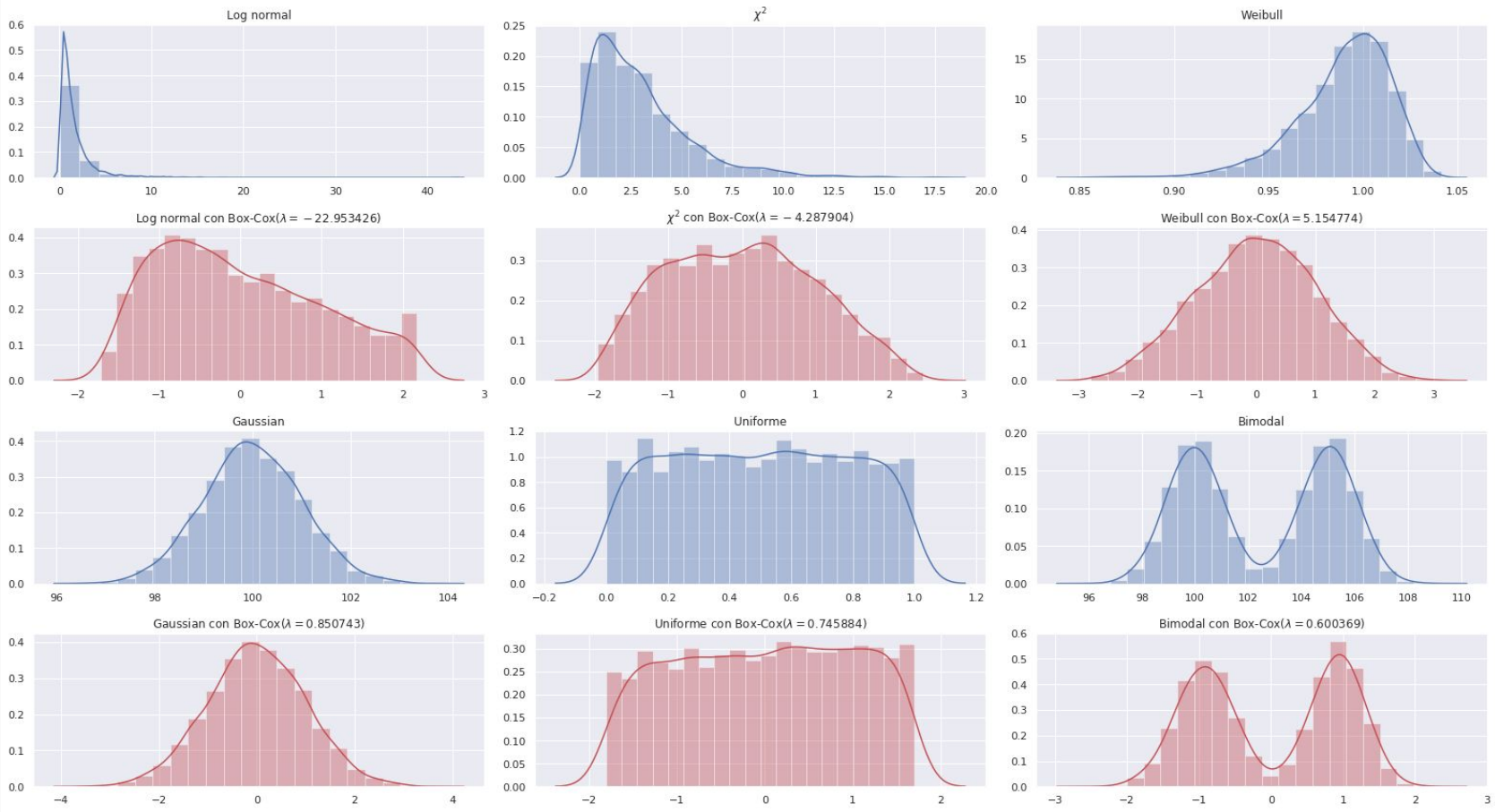
- Junto con Box-Cox, es otra de las transformaciones más utilizadas.
- Admite que X tome valores negativos.

$$y_i^{(\lambda)} = \begin{cases} ((y_i + 1)^{\lambda} - 1)/\lambda & \text{if } \lambda \neq 0, y \geq 0 \\ \log(y_i + 1) & \text{if } \lambda = 0, y \geq 0 \\ -[(-y_i + 1)^{(2-\lambda)} - 1]/(2 - \lambda) & \text{if } \lambda \neq 2, y < 0 \\ -\log(-y_i + 1) & \text{if } \lambda = 2, y < 0 \end{cases}$$

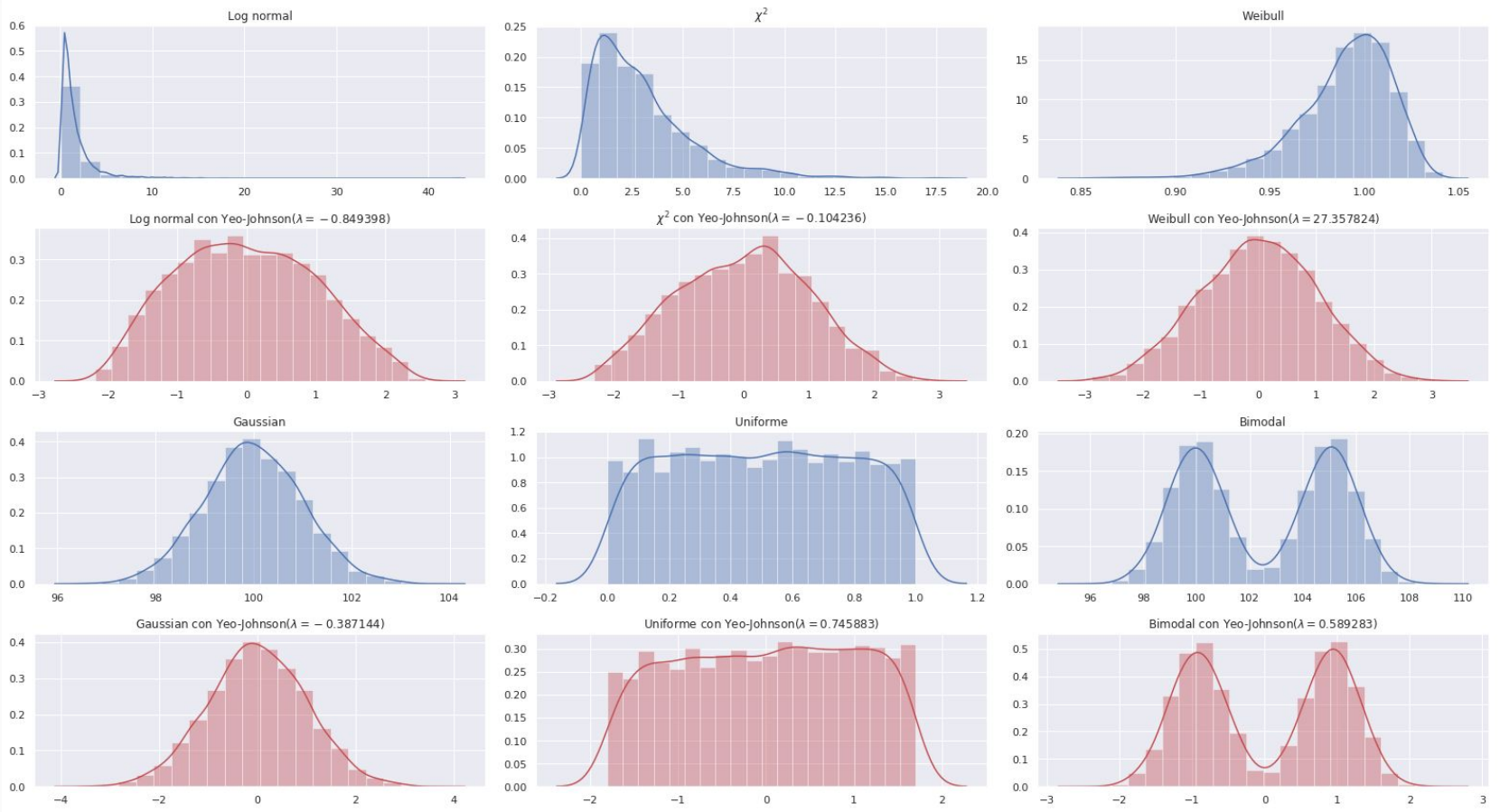
Ejemplos



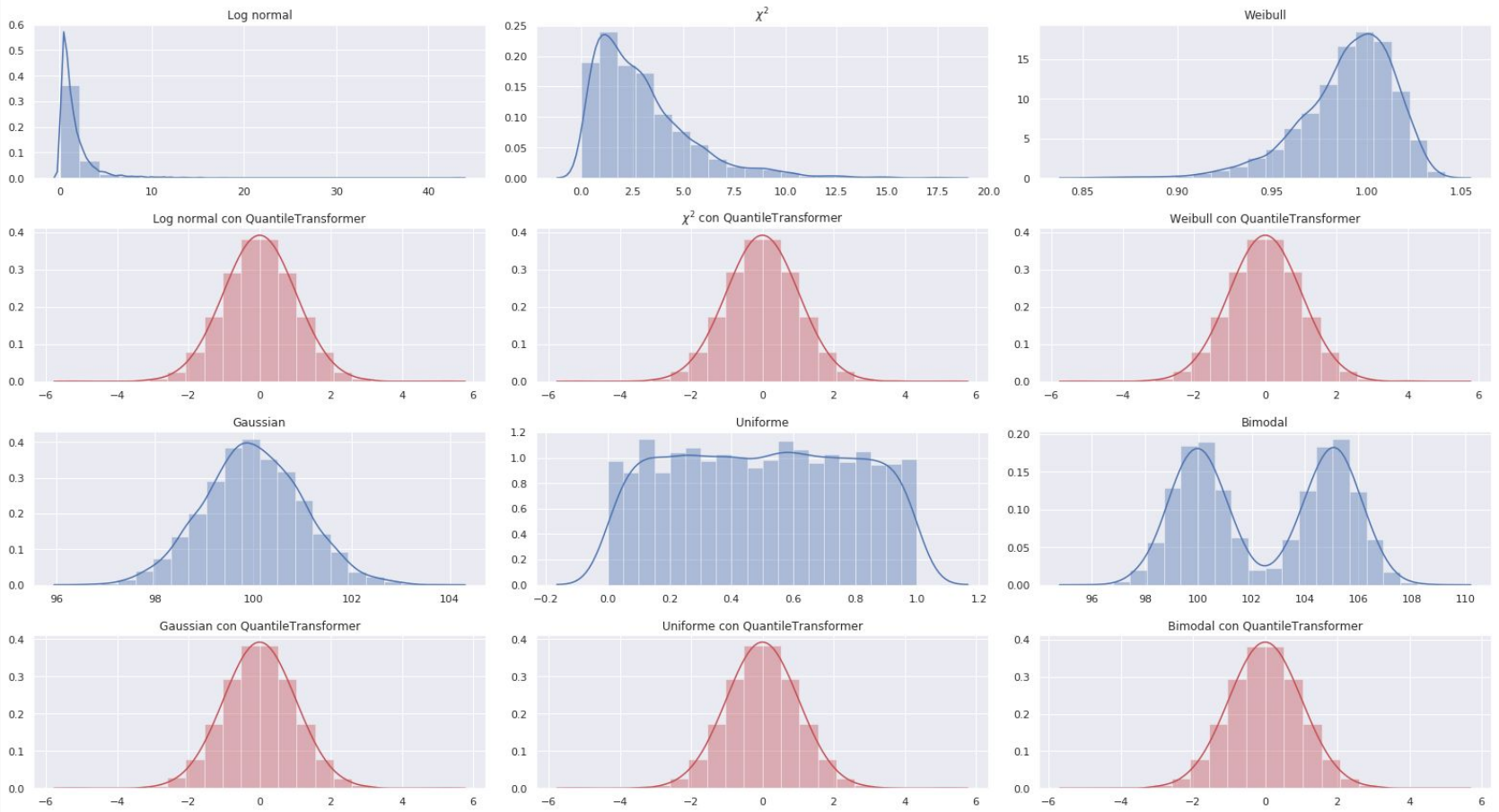
Ejemplos. Box-Cox



Ejemplos. Yeo-Johnson



Ejemplos. QuantileTransformer



Ejemplos en jupyter

Clase 4.5 - Preparación de datos - Transformación de variables.ipynb

6. Discretización

Definición

- Es el proceso de transformar variables continuas en variables con valores discretos mediante la creación de intervalos contiguos que cubren el espectro de valores que toma la variable.
- También se le llama **binning**, donde “bin” (en inglés “cesto”) es un nombre alternativo para un intervalo.

Motivación

- En preparación de datos para modelos de ML, es de interés discretizar por una o más de las siguientes razones:
  - algunas técnicas para discretizar permiten convertir distribuciones con oblicuidad en distribuciones normales o uniformes.
  - pueden agrupar valores extremos en el primer y último intervalo.

Métodos de discretización

- No supervisados:
  - Ancho fijo
  - Frecuencia fija
  - K-means
- Supervisados:
  - Árboles de decisión

Métodos de discretización

- No supervisados:
  - Ancho fijo
  - Frecuencia fija
  - K-means
- Supervisados:
  - Árboles de decisión

Ancho fijo. Definición y características.

- Divide el espectro de valores posible en N bins del mismo ancho.
- El tamaño de cada intervalo está dado por:
  - ancho = (max-min)/N
- Características:
  - No mejora la distribución.
  - Permite manejar valores extremos.
  - Crea una variable discreta.
  - Puede combinarse con codificaciones categóricas.

Frecuencia fija. Definición y características.

- Divide el espectro de valores posible en N bins, donde cada bin tiene la misma cantidad de observaciones.
- La definición de cada intervalo se realiza a partir de los cuantiles
- Características:
  - Mejora la distribución.
  - Permite manejar valores extremos.
  - Crea una variable discreta.
  - Puede combinarse con codificaciones categóricas.

Frecuencia fija. Definición y características.

- Consiste en aplicar k-means para hallar n intervalos.
- Explicación (pizarra).
- Características:
  - No mejora la distribución.
  - Permite manejar valores extremos, si bien pueden influenciar la ubicación de los centroides.
  - Crea una variable discreta.
  - Puede combinarse con codificaciones categóricas.

Discretización + codificación

- FIXME!

Discretización con árboles de decisión. Definición.

- Consiste en utilizar un árbol de decisión para hallar los bins óptimos.
- Una decisión de un árbol asigna una observación a una de sus N hojas.
- Los árboles generan una salida discreta, cuyos valores son las predicciones para cada una de sus N hojas.
- Explicación (pizarra).

Discretización con árboles de decisión. Características.

- No mejora la distribución.
- Maneja outliers.
- Crea una variable discreta.
- Crea una relación monotónica.

7. Tratamiento de valores extremos

Tratamiento de valores extremos. Métodos.

Podado

- Remover los valores extremos del dataset.

Datos faltantes

- Tratar los valores extremos como datos faltantes.

Discretización

- Ubicar los valores extremos en primer y último bin.

Censurar

- Cortar
- Codificación extremo inferior/superior.
- Winsorization

Cada aproximación tiene ventajas y desventajas.

8. Feature scaling

Motivación. Cómo influye la magnitud de las variables de entrada (1 / 2)

- El coeficiente de regresión está directamente influenciado por la escala de la variable.
- Las variables con mayor rango de magnitud/valor dominan sobre las que tienen un menor rango de magnitud/valor.
- Los algoritmos de la familia de Gradient Descent convergen más rápidamente cuando las variables de entrada están en la misma escala.
- Las distancias euclídeas son sensibles a la magnitud de las variables.

Motivación Cómo influye la magnitud de las variables de entrada (2 / 2)

- Algoritmos afectados
  - Regresión lineal y logística.
  - Redes neuronales
  - Support Vector Machines
  - KNN
  - K-means
  - Principal Component Analysis (PCA)
- No afectados (por ejemplo, los basados en árboles)
  - Árboles de clasificación y regresión.
  - Random Forest
  - Gradient Boosted Trees.

Feature scaling (escalado de variables)

- En ML, se refiere a los métodos utilizados para normalizar el rango de valores que puede tomar una variable independiente.
- Suele ser el último paso en una cadena de procesamiento de datos (realizado justo antes del entrenamiento de algoritmos).

Feature scaling. Métodos

- Los más utilizados:
  - Estandarización
  - Escalado a mínimo-máximo.
- Otros métodos:
  - Normalización de media.
  - Escalado a valor absoluto.
  - Escalado a mediana y cuantiles.
  - Escalado a norma unitaria.

Estandarización

$$Z = \frac{x - \mu}{\sigma}$$

- Centra la variable en cero y establece su varianza a 1.

Estandarización. Efecto

- Centra la media en 0.
- Escala la varianza a 1.
- Preserva la forma de la distribución original.
- Los mínimos y máximos pueden variar.
- Preserva los valores extremos (outliers).

Escalado a mínimo-máximo.

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- Esta transformación restringe el valor de la variable a un rango (típicamente x_min=0,x_max=1).

Escalado a mínimo-máximo. Efecto.

- La media puede variar.
- La varianza puede variar.
- Puede modificar la forma de la distribución original.
- Los mínimos y máximos quedan restringidos.
- Preserva los valores extremos (outliers).

Implementación de cadenas de procesamiento con SKLearn

- Ejemplos de uso de clases **Pipeline** y **ColumnTransformer** para problemas de clasificación y regresión.
- Referencias:
  - https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html
  - https://scikit-learn.org/stable/modules/compose.html#columntransformer-for-heterogeneous-data
  - Sección de trransformaciones de datos de documentación de SKLearn: https://scikit-learn.org/stable/data_transforms.html

Resumen

- El objetivo de esta clase fue presentar algunas de las técnicas más utilizadas para la preparación de datasets y cómo combinarlas para construir cadenas de procesamiento en SKLearn.
- En las siguientes clases, se estudiarán otros aspecto de la **preparación de datos**:
  - La **selección de variables** de entrada: ¿Qué variables son más relevantes para el modelo de AA?
  - **Ingeniería de variables** (como generar variables que aporten mayor información, a partir de las existentes).
  - **Reducción de dimensiones**. Creación de un nuevo espacio de variables entrada, a partir de proyecciones compactas.

Bibliografía y referencias

- *“Python Feature Engineering Cookbook” .Soledad Galli. Packt (2020)*
- *“Applied Predictive Modeling”. Max Kuhn; Kjell Johnson. Springer (2016)*
- *“Feature Engineering and Selection”. Max Kuhn; Kjell Johnson. CRC Press (2016)*
- *“Feature Selection for Data and Pattern Recognition”. Urszula Stańczyk; Lakhmi C. Jain. Springer (2014)*
- *“Feature Engineering for Machine Learning”. Alice Zheng; Amanda Casari. O’Reilly (2018)*
- *“The Art of Feature Engineering”. Pablo Doboue. Cambridge University Press (2020).*
- *“Feature Engineering IFT6758 - Data Science”* Université de Montréal.