

✓ Ejemplo práctico durante la clase (9-jul-2024, 18:47)

Por: Andrés Felipe Escallón Portilla

```
# Import libraries
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np

# Create a dataset
# Let's build the values lists
value_n_list = ['Andrew', 'Phillip', 'John']
print(value_n_list)


value_w_list = [50,      78,      91.5] # [kg]
value_h_list = [1.76,    1.87,    1.69] # [m]
# value_bmi_list will be calculated: bmi = w/(h^2)
#value_bmi_list = [25,25,25] # init values [kg/m2]

# Let's use a dictionary ({key:value})
# with the format {"str":list}
data_dict = {"name":value_n_list,
             "weight":value_w_list,
             "height":value_h_list} #,
             #"BMI":value_bmi_list}

print(data_dict)


# Turn a dictionary into a dataframe
df = pd.DataFrame(data_dict)

print(df)
# Call a variable
df
```




```
[ 'Andrew', 'Phillip', 'John']
{'name': [ 'Andrew', 'Phillip', 'John'], 'weight': [50, 78, 91.5], 'height': [1.76, 1.87, 1.69]}
```

	name	weight	height
0	Andrew	50.0	1.76
1	Phillip	78.0	1.87
2	John	91.5	1.69




	name	weight	height
0	Andrew	50.0	1.76
1	Phillip	78.0	1.87
2	John	91.5	1.69

```
df
```




	name	weight	height
0	Andrew	50.0	1.76
1	Phillip	78.0	1.87
2	John	91.5	1.69

```
df
```



	name	weight	height
0	Andrew	50.0	1.76
1	Phillip	78.0	1.87
2	John	91.5	1.69


```
df
```



	name	weight	height
0	Andrew	50.0	1.76
1	Phillip	78.0	1.87
2	John	91.5	1.69


```
# prompt: Con DataFrame df: add another column

df['bmi'] = df['weight'] / (df['height'] ** 2)
df
```



	name	weight	height	bmi
0	Andrew	50.0	1.76	16.141529
1	Phillip	78.0	1.87	22.305471
2	John	91.5	1.69	32.036693


```
df.describe()
```



	weight	height	bmi
count	3.000000	3.000000	3.000000
mean	73.166667	1.773333	23.494564
std	21.167979	0.090738	8.014020
min	50.000000	1.690000	16.141529
25%	64.000000	1.725000	19.223500
50%	78.000000	1.760000	22.305471
75%	84.750000	1.815000	27.171082
max	91.500000	1.870000	32.036693

✖ Making some transformations on the df

```
# z = x / y*y
df["BMI"] = df["weight"] / df["height"]**2
df
```



	name	weight	height	bmi	BMI
0	Andrew	50.0	1.76	16.141529	16.141529
1	Phillip	78.0	1.87	22.305471	22.305471
2	John	91.5	1.69	32.036693	32.036693

De acuerdo a los resultados puede concluir que:
-El valor max es: -

```
df["BMI"].dtype

dtype('float64')
```


```
# TO-DO: create another column with integers and turn it to float, such as:
df["BMI"] = df["BMI"].astype(float)
```

```
df["BMI"].dtype

dtype('float64')
```

```
df["name"].dtype
# Homework: how to verify that a column contains string

dtype('O')
```




	name	weight	height	bmi	BMI
0	Andrew	50.0	1.76	16.141529	16.141529
1	Phillip	78.0	1.87	22.305471	22.305471
2	John	91.5	1.69	32.036693	32.036693

✖ Init steps towards EDA (Exploratory Data Analysis):

Homework: perform a full EDA with your own invented dataset with at least 10 rows

```
# Descriptive stats
df_stats = df.describe()
df_stats
```



	weight	height	bmi	BMI
count	3.000000	3.000000	3.000000	3.000000
mean	73.166667	1.773333	23.494564	23.494564
std	21.167979	0.090738	8.014020	8.014020
min	50.000000	1.690000	16.141529	16.141529
25%	64.000000	1.725000	19.223500	19.223500
50%	78.000000	1.760000	22.305471	22.305471
75%	84.750000	1.815000	27.171082	27.171082
max	91.500000	1.870000	32.036693	32.036693

✖ TO-DO: please analyze the results (univariate: describe() / multivariate: correlation matrix)

```
#df["health_condition"] = 'within the normal range' if 18.0<=df['BMI']<=25.0 else 'out of the normal range'
#df
```

✖ Prompt used in chat GPT (by OpenAI) / Gemini (by Google) / Bing (Copilot by Microsoft) to solve the above error:

You are a python expert, please help with the error I will show later that is appearing when I run this code in Google collab:
"df["health_condition"] = 'within the normal range' if 18.0<=df['BMI']<=25.0 else 'out of the normal range'".
The error is:

ValueError Traceback (most recent call last) in <cell line: 1>() ----> 1 df["health_condition"] = 'within the normal range' if 18.0<=df['BMI']<=25.0 else 'out of the normal range' 2 df

[/usr/local/lib/python3.10/dist-packages/pandas/core/generic.py](#) in nonzero(self) 1525 @final 1526 def nonzero(self) -> NoReturn: -> 1527 raise ValueError(1528 f"The truth value of a {type(self).name} is ambiguous. " 1529 "Use a.empty, a.bool(), a.item(), a.any() or a.all()."

ValueError: The truth value of a Series is ambiguous. Use a.empty, a.bool(), a.item(), a.any() or a.all().

df

	name	weight	height	bmi	BMI
0	Andrew	50.0	1.76	16.141529	16.141529
1	Phillip	78.0	1.87	22.305471	22.305471
2	John	91.5	1.69	32.036693	32.036693

print(df)

	name	weight	height	bmi	BMI
0	Andrew	50.0	1.76	16.141529	16.141529
1	Phillip	78.0	1.87	22.305471	22.305471
2	John	91.5	1.69	32.036693	32.036693

df_stats

	weight	height	bmi	BMI
count	3.000000	3.000000	3.000000	3.000000
mean	73.166667	1.773333	23.494564	23.494564
std	21.167979	0.090738	8.014020	8.014020
min	50.000000	1.690000	16.141529	16.141529
25%	64.000000	1.725000	19.223500	19.223500
50%	78.000000	1.760000	22.305471	22.305471
75%	84.750000	1.815000	27.171082	27.171082
max	91.500000	1.870000	32.036693	32.036693

print(df_stats)

	weight	height	bmi	BMI
count	3.000000	3.000000	3.000000	3.000000
mean	73.166667	1.773333	23.494564	23.494564
std	21.167979	0.090738	8.014020	8.014020
min	50.000000	1.690000	16.141529	16.141529
25%	64.000000	1.725000	19.223500	19.223500
50%	78.000000	1.760000	22.305471	22.305471
75%	84.750000	1.815000	27.171082	27.171082
max	91.500000	1.870000	32.036693	32.036693

df["health_condition"] = np.where((df['BMI'] >= 18.0) & (df['BMI'] <= 25.0) , 'within the normal range', 'out of the normal range')
df

	name	weight	height	bmi	BMI	health_condition
0	Andrew	50.0	1.76	16.141529	16.141529	out of the normal range
1	Phillip	78.0	1.87	22.305471	22.305471	within the normal range
2	John	91.5	1.69	32.036693	32.036693	out of the normal range

```
# Define a function to determine health condition
def determine_health_condition(bmi):
    if 18.0 <= bmi <= 25.0:
        return 'within the normal range'
    else:
        return 'out of the normal range'

# Apply the function to create the new column
df["health_condition4"] = df["BMI"].apply(determine_health_condition)
```

print(df)
df

	name	weight	height	bmi	BMI	health_condition	\
0	Andrew	50.0	1.76	16.141529	16.141529	out of the normal range	
1	Phillip	78.0	1.87	22.305471	22.305471	within the normal range	
2	John	91.5	1.69	32.036693	32.036693	out of the normal range	
	health_condition4						
0	out of the normal range						
1	within the normal range						
2	out of the normal range						
	name	weight	height	bmi	BMI	health_condition	health_condition4
0	Andrew	50.0	1.76	16.141529	16.141529	out of the normal range	out of the normal range
1	Phillip	78.0	1.87	22.305471	22.305471	within the normal range	within the normal range
2	John	91.5	1.69	32.036693	32.036693	out of the normal range	out of the normal range

```
# Define a function to determine health condition
def determine_health_condition7(bmi):
    """
    if 18.0 <= bmi <= 25.0:
        return 'within the normal range'
    else:
        return 'out of the normal range'
    """
    return 'within the normal range' if 18.0 <= bmi <= 25.0 else 'out of the normal range'

# Apply the function to create the new column
df["health_condition6"] = df["BMI"].apply(determine_health_condition7)
df
```

↗

	name	weight	height	bmi	BMI	health_condition	health_condition4	health_condit:
0	Andrew	50.0	1.76	16.141529	16.141529	out of the normal range	out of the normal range	out of the normal range
1	Phillip	78.0	1.87	22.305471	22.305471	within the normal range	within the normal range	within the normal range

◀ ▶

Please remove the unnecessary columns and search how to upload/download the df as .csv and .json and .xlsx

> Correlation matrix (14-jun-2024)

ChatGPT prompt used:

"you're a python expert, please give code to get the correlation matrix of a dataframe with a simple example and calculus explanation"

[] ↴ 1 celda oculta

√ A more complete example with synthetic data given by ChatGPT

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Generate synthetic data
np.random.seed(42) # for reproducibility

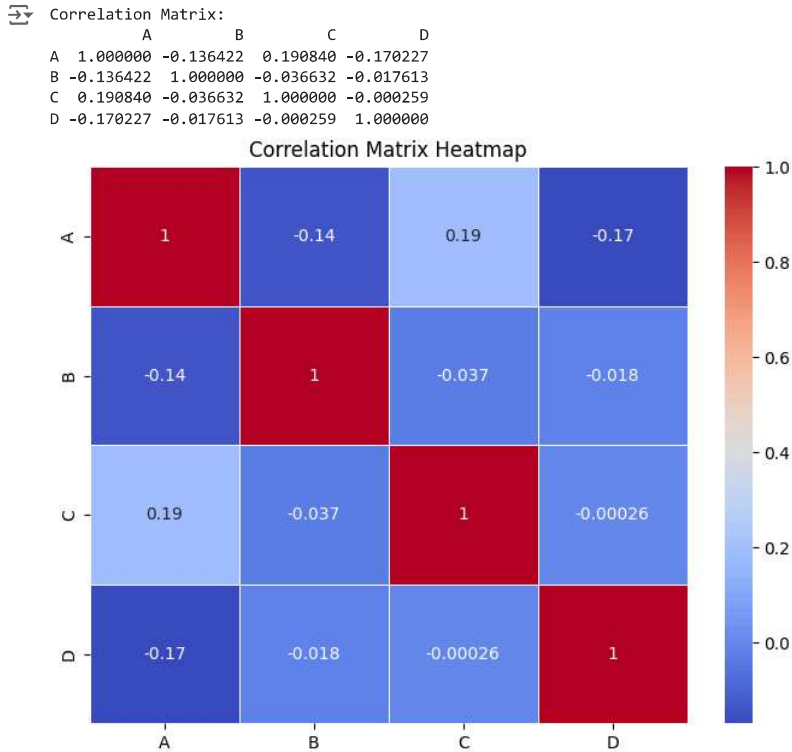
data = {
    'A': np.random.randn(100),
    'B': np.random.randn(100) * 2,
    'C': np.random.randn(100) + 1,
    'D': np.random.randn(100) - 1,
}

df = pd.DataFrame(data)

# Calculate the correlation matrix
correlation_matrix = df.corr()

# Print the correlation matrix
print("Correlation Matrix:")
print(correlation_matrix)

# Visualize the correlation matrix using a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)
plt.title('Correlation Matrix Heatmap')
plt.show()
```




data



```
-3.10132686, 0.13712595, -2.12460745, 0.94718486, -1.83884847,
3.09986881, -1.56650658, -0.64412303, 1.62703443, -2.46172863,
0.45491987, 2.61428551, -3.21496647, 0.36926772, 0.51976559,
1.56364574, -2.47390142, -2.64091323, 1.04388313, 0.59396935,
0.5009857 , 0.69289642, -1.36004944, 0.46450739, 0.58614495,
-1.42870284, 3.73154902, 0.94766584, -2.38260699, 1.31310722,
-1.94936334, 1.57416921, 2.31719116, -1.64136464, 1.92675226,
0.82556185, 1.64412032, 3.79358597, -0.49077623, -1.50747233,
-1.77902886, -1.63162057, -0.15420342, 0.68230395, 0.5533816 ,
1.6543665 , 0.02600378, 2.90706815, -0.52931367, 5.44033833,
1.2513347 , -1.71431511, -2.141785 , 0.96494483, -0.44692557,
1.42800099, 0.94647525, -0.14565783, -1.69358744, -3.02969445,
-0.8930299 , 1.71279759, 0.42818749, -2.49147756, 0.34636185,
0.77063476, -1.76771487, 0.30745021, 0.11641744, -2.2859406 ]),
'C': array([ 1.35778736,  1.56078453,  2.08305124,  2.05380205, -0.37766937,
 0.06217496,  1.51503527,  1.51378595,  1.51504769,  4.85273149,
 1.57089051,  2.13556564,  1.95400176,  1.65139125,  0.68473076,
 1.75896922,  0.22717479,  0.76318139,  0.51463645,  1.08187414,
 3.31465857, -0.86726519,  1.68626019, -0.61271587,  0.52806813,
 2.0889506 ,  1.06428002, -0.07774478,  0.28469629,  1.67959775,
 0.26963337,  1.21645859,  1.04557184,  0.34839965,  3.14394409,
 1.63391902, -1.02514259,  1.18645431,  0.33821354,  1.85243333,
 0.20747926,  0.88526356,  1.50498728,  1.86575519, -0.20029641,
 0.66549876,  0.52505469,  0.34667077,  2.76545424,  1.40498171,
 -0.26088395,  1.91786195,  3.1221562 ,  2.03246526, -0.51936997,
 0.51576593,  2.26691115,  0.29233053,  1.44381943,  1.77463405,
 0.07306953,  0.94047464, -2.24126734, -0.02438764,  0.74743185,
 -0.24778318,  2.6324113 , -0.43014138,  0.55995551,  1.13074058,
 2.44127329, -0.43586215,  2.16316375,  1.01023306,  0.01849135,
 1.46210347,  1.1990597 ,  0.39978312,  1.06980208,  0.6146864 ,
 1.11351735,  1.66213067,  2.58601682, -0.2378155 ,  3.13303337,
 -0.9520878 ,  0.8482149 ,  1.58831721,  1.28099187,  0.37730048,
 0.79187775,  0.50699907,  0.41063524,  1.8496021 ,  1.35701549,
 0.3070904 ,  1.89959988,  1.30729952,  1.81286212,  1.62962884]),
'D': array([-1.82899501, -1.56018104, -0.25270639, -0.38962973, -1.02090159,
 -0.88267262,  0.2776649 , -1.59157139, -0.45290262, -1.20219265,
 -1.2176812 ,  0.09877685, -0.17458365, -0.18649036,  0.30547881,
 -0.97899616, -0.31804703, -1.31026676, -0.67583365, -1.13014305,
 -0.90300404, -0.40484297, -1.81822068,  1.09238728, -2.00601738,
 -2.21418861,  0.15811087, -0.20833731, -0.37588018, -0.37165449,
 -1.01224677, -1.89725437, -0.92419544, -1.67716171, -0.02488027,
 -1.14705738, -1.8254972 , -1.32138584, -0.58706855, -1.56372455,
 -1.8222204 , -0.75631279, -0.75503343, -1.50694318, -1.47103831,
 -0.76795006, -2.44808434, -2.40746377, -1.71844422, -1.21344715,
 -0.68909243,  0.47535622, -0.14234038, -1.15993853, -1.01901621,
 -2.00252936, -1.01851314, -1.28865864, -0.67728144, -1.82723094,
 -0.48065349,  0.53273891, -1.10876015, -0.59828828, -0.30985601,
 -1.40122047, -0.77590752, -0.9874076 , -0.9023239 , -1.77300978,
 -0.97548983, -0.50200171,  0.45114361, -0.04072917,  1.15318246,
 -1.76734756, -0.12767936, -0.81665799,  1.18980293, -1.80829829,
 -1.83972184, -1.59939265, -3.12389572, -1.52575502, -1.75913266,
 -0.84960621, -0.65824402,  0.87617084, -0.04957616, -1.57690366,
 -1.89841467, -0.50808083, -2.32023321,  0.83145877,  0.17944012,
 -1.46917565, -2.71313453,  0.35387237, -1.11453985,  0.237816311])}
```

df



	A	B	C	D
0	0.496714	-2.830741	1.357787	-1.828995
1	-0.138264	-0.841291	1.560785	-1.560181
2	0.647689	-0.685429	2.083051	-0.252706
3	1.523030	-1.604555	2.053802	-0.389630
4	-0.234153	-0.322571	-0.377669	-1.020902
...
95	-1.463515	0.770635	0.307090	-1.469176
96	0.296120	-1.767715	1.899600	-2.713135
97	0.261055	0.307450	1.307300	0.353872
98	0.005113	0.116417	1.812862	-1.114540
99	-0.234587	-2.285941	1.629629	0.237816

100 rows × 4 columns

✖ Consultar cómo es el cálculo de la matriz de correlación (PEND):

Chat GPT Prompt: "Please, give me the explanation mathematically about the way the correlation matrix computation is performed"

Es una herramienta estadística que muestra las correlaciones entre todas las variables en un conjunto de datos. Se utiliza principalmente para entender cómo las variables se relacionan entre sí y para identificar patrones o dependencias lineales.

El cálculo de la matriz de correlación implica los siguientes pasos básicos:

- 1- Organización de datos: Tener un conjunto de datos donde se mide al menos dos variables para cada observación. Por ejemplo, podríamos tener variables como edad, ingreso, gasto mensual, etc., para cada individuo en un estudio.
- 2- Cálculo de la correlación: Para calcular la correlación entre dos variables (digamos X e Y), se utiliza un coeficiente de correlación. El coeficiente de correlación más comúnmente usado es el coeficiente de correlación de Pearson, que mide la relación lineal entre dos variables continuas. Se calcula de la siguiente manera:

$$r_{XY} = \frac{\sum (X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum (X_i - \bar{X})^2 \sum (Y_i - \bar{Y})^2}}$$

donde:

- X_i, Y_i son los valores individuales de las variables X e Y.
- \bar{X}, \bar{Y} son las medias de X e Y, respectivamente.

3- Construcción de la matriz: Una vez calculados todos los coeficientes de correlación entre todas las variables del conjunto de datos, estos se organizan en una matriz cuadrada, donde las filas y columnas representan las variables. La diagonal principal de la matriz contiene siempre unos (correlación de una variable consigo misma).

4- Interpretación: La matriz de correlación resultante permite interpretar la fuerza y la dirección de las relaciones entre las variables. Los valores del coeficiente de correlación van de -1 a 1:

- Cerca de 1 indica una correlación positiva fuerte (las variables aumentan juntas).
- Cerca de -1 indica una correlación negativa fuerte (una variable aumenta mientras la otra disminuye).
- Cerca de 0 indica una correlación débil o inexistente.

5- Visualización: A menudo, se visualiza la matriz de correlación mediante un mapa de calor (heatmap), donde los valores más altos de correlación se muestran en tonos más oscuros o más intensos.

Es importante recordar que la correlación no implica causalidad; es decir, solo porque dos variables estén correlacionadas, no significa necesariamente que una causa la otra. Sin embargo, la matriz de correlación es una herramienta poderosa para explorar y comprender las relaciones dentro de un conjunto de datos.

Resultado:

Incluir formulas como si estuvieran escritas en un tablero

```
# E`sto tiene formato de código`
```

↯ Tarea: Consultar qué es y cómo funciona la matriz de confusión (clasificación)

La matriz de confusión es una herramienta fundamental en el campo del aprendizaje automático y la estadística, especialmente en problemas de clasificación. Su función principal es mostrar el desempeño de un modelo de clasificación comparando las predicciones del modelo con los valores reales de los datos.

Estructura de la Matriz de Confusión

En una matriz de confusión típica, los elementos diagonales (de izquierda arriba a derecha abajo) representan las predicciones correctas, mientras que los elementos fuera de la diagonal indican errores de predicción. Aquí están los componentes principales de la matriz:

- Verdaderos positivos (TP): Estas son las instancias que el modelo predijo Elemento de lista correctamente como pertenecientes a la clase positiva.
- Falsos positivos (FP): Estas son las instancias que el modelo predijo incorrectamente como pertenecientes a la clase positiva (cuando en realidad pertenecen a otra clase).
- Verdaderos negativos (TN): Estas son las instancias que el modelo predijo correctamente como no pertenecientes a la clase positiva.
- Falsos negativos (FN): Estas son las instancias que el modelo predijo incorrectamente como no pertenecientes a la clase positiva (cuando en realidad sí pertenecen a la clase positiva).

Funcionamiento de la Matriz de Confusión

Generación de predicciones: Primero, el modelo de clasificación realiza predicciones sobre un conjunto de datos de prueba.

Comparación con valores reales: Luego, las predicciones del modelo se comparan con los valores reales conocidos de las etiquetas de clase (ground truth).

Construcción de la matriz: Con base en estas comparaciones, se construye la matriz de confusión, donde se cuentan las instancias correctamente clasificadas y las incorrectamente clasificadas en cada clase.

Interpretación:

- Diagonal principal: Muestra el número de predicciones correctas para cada clase.
- Fuera de la diagonal principal: Muestra los errores cometidos por el modelo, donde cada celda (i,j) representa cuántas instancias de la clase i se predijeron como clase j .

Importancia de la Matriz de Confusión

La matriz de confusión proporciona una visión detallada del rendimiento del modelo de clasificación en términos de precisión, sensibilidad, especificidad y otros indicadores clave:

1. Exactitud (Accuracy): Proporción de predicciones correctas respecto al total de predicciones.
2. Precisión (Precision): Proporción de predicciones positivas correctas respecto al total de predicciones positivas hechas por el modelo.
3. Recuperación o Sensibilidad (Recall o Sensitivity): Proporción de instancias positivas que fueron correctamente predichas por el modelo.
4. Especificidad (Specificity): Proporción de instancias negativas que fueron correctamente predichas por el modelo.

La matriz de confusión es fundamental para evaluar y comparar diferentes modelos de clasificación y para ajustar los umbrales de decisión según los requisitos específicos del problema.

En resumen, la matriz de confusión proporciona una representación visual y cuantitativa del rendimiento del modelo de clasificación, ayudando a entender dónde y cómo el modelo está cometiendo errores en sus predicciones.

```
# T0-D0
```

Las metricas...

↯ Consultar las métricas de desempeño para regresión y clasificación

Las métricas de desempeño son herramientas fundamentales para evaluar cuantitativamente el rendimiento de modelos en problemas de regresión y clasificación. A continuación, te proporciono un resumen de las principales métricas utilizadas en cada tipo de problema:

Métricas de Desempeño para Regresión En problemas de regresión, el objetivo es predecir valores numéricos continuos. Aquí están algunas de las métricas comunes utilizadas para evaluar modelos de regresión:

1. Eror Absoluto Medio (Mean Absolute Error, MAE):

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

Donde y_i son los valores reales, \hat{y}_i son las predicciones del modelo, y n es el número de muestras.

El MAE mide el promedio de las diferencias absolutas entre las predicciones y los valores reales. Es

fácil de entender y de interpretar.

2. Error Cuadrático Medio (Mean Squared Error, MSE):

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

El MSE es similar al MAE, pero eleva al cuadrado las diferencias entre las predicciones y los valores reales, lo que da más peso a los errores grandes. Es útil porque penaliza más fuertemente los errores grandes.

3. Raíz del Error Cuadrático Medio (Root Mean Squared Error, RMSE): oeficiente de Determinación (R-cuadrado o R2):

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

Es simplemente la raíz cuadrada del MSE y proporciona una métrica en la misma escala que la variable objetivo original, lo que facilita su interpretación en términos del dominio del problema.

4. Coeficiente de Determinación (R-cuadrado o R^2):

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

Donde \bar{y} es la media de los valores reales y_i . El R^2 indica qué tan bien las predicciones del modelo se ajustan a los datos reales. Un R^2 de 1 indica un ajuste perfecto, mientras que valores menores indican que el modelo no explica bien la variabilidad de los datos.

Métricas de Desempeño para Clasificación

En problemas de clasificación, el objetivo es predecir la clase (etiqueta) a la que pertenece una muestra. Aquí están algunas de las métricas comunes utilizadas para evaluar modelos de clasificación:

1. Exactitud (Accuracy):

Número de predicciones correcta

Accuracy =

Número total de predicciones

2. Precisión y Recall (Sensibilidad): 2.1 Precisión (Presicion):

TP

Precisión =

TP + FP

2.2 Recall o Sensibilidad (Recall o Sensitivity):

TP

Recall =

TP + FN

Donde FN es el número de falsos negativos. Recall mide la proporción de instancias positivas que fueron correctamente pre



Estas dos métricas son complementarias y se utilizan para evaluar el rendimiento del modelo, especialmente en problemas donde una clase puede ser más relevante que otra (por ejemplo, en diagnósticos médicos).

3. F1 - Score:

Presicion * Recall

F1 = 2 *

Presicion + Recall

El F1-score es la media armónica de precision y recall, proporcionando una métrica única que equilibra ambas medidas. Es útil cuando hay un desequilibrio entre las clases en el conjunto de datos.

4. Matriz de Confusión:

Aunque no es una métrica en sí misma, la matriz de confusión es una herramienta esencial para visualizar y entender el rendimiento del modelo de clasificación, como se explicó anteriormente.

Selección de Métricas

La elección de las métricas adecuadas depende del tipo de problema que estés abordando y de los objetivos específicos de tu modelo. Es importante considerar las características del conjunto de datos, como el desbalance de clases en problemas de clasificación, y seleccionar métricas que proporcionen información relevante sobre el rendimiento del modelo en el contexto del problema particular que estás tratando de resolver.

Las métricas...

T0-D0

Las métricas...

Empieza a programar o a [crear código](#) con IA.

Fuentes de consulta:

- <https://chatgpt.com/>
- <https://eragroup.eu/ejemplos-del-coeficiente-de-correlacion-de-pearson/>