«Talento Tech»

Data Analytics

con Python

Clase 09

Buenos Aires Ciudad

Clase N° 09 | Estadística Descriptiva

Temario:

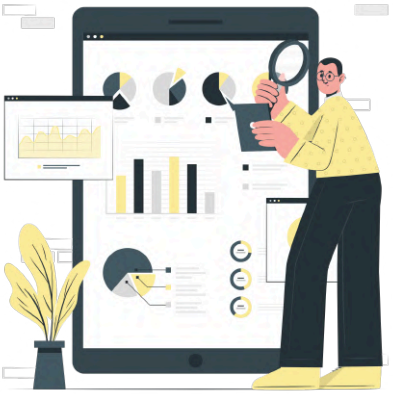
- Repaso y profundización: Estadística descriptiva.
    - Medidas de tendencia central.
    - Medidas de dispersión.

Objetivos de la Clase:

- Conocer algunas de las librerías de Python para estadística.
- Repasar las Medidas de Tendencia Central y Medidas de Dispersión
- Aplicar Conceptos a Conjuntos de Datos: Relacionar las medidas estudiadas con ejemplos prácticos en Python, utilizando bibliotecas como Pandas y NumPy.

1. Librerías de Python para Estadísticas

La estadística es un área muy amplia y compleja, que abarca múltiples temas y técnicas. En el ámbito de Python, existen varias **librerías** que permiten abordar distintas facetas de esta disciplina. A menudo, se utiliza una combinación de varios paquetes para cubrir las necesidades de análisis estadístico, dado que algunas librerías son más adecuadas para ciertos tipos de tareas en comparación con otras.



A continuación, detallamos algunas de las librerías más importantes que utilizaremos en este curso y especificaremos cuál es el uso específico en estadística de las librerías que ya conocemos:

SciPy.stats

scipy.stats es una **sublibrería de SciPy** que proporciona un amplio conjunto de funciones para realizar **análisis estadísticos**. Incluye herramientas para:

- **Distribuciones de probabilidad**: Puedes calcular funciones de densidad y de masa de probabilidad, así como obtener estadísticas descriptivas.
- **Pruebas estadísticas**: Realiza pruebas de hipótesis, como pruebas t, ANOVA y pruebas de chi-cuadrado.
- **Intervalos de confianza**: Estima intervalos de confianza para diferentes parámetros estadísticos.

Este paquete es esencial para realizar análisis estadísticos básicos y es adecuado para tareas que requieren una comprensión fundamental de la teoría estadística.

Statsmodels

Statsmodels es una **librería para realizar análisis más avanzados en estadística**. Algunas de sus principales características incluyen:

- **Regresión**: Permite realizar regresiones lineales y no lineales, modelos de efectos mixtos, y modelos generales de regresión.
- **Modelos lineales**: Proporciona herramientas para construir y evaluar modelos lineales de regresión, incluidas las regresiones múltiples.
- **Análisis de series de tiempo**: Ofrece funciones para modelar, predecir y evaluar series temporales, lo que es crucial en muchas áreas como econometría y finanzas.
- **Extensiones de SciPy.stats**: Abarca técnicas adicionales que no están presentes en SciPy, brindando un enfoque más exhaustivo en análisis complejos.

Esta librería es ideal si deseamos profundizar en técnicas más elaboradas y modelado estadístico.

Pandas

Pandas es una librería fundamental para la **manipulación y análisis de datos en formato tabular**. Sus características más importantes son:

- **Datos tabulares**: Permite manipular, limpiar y transformar datos de forma sencilla, utilizando estructuras como DataFrames.
- **Funcionalidad de series de tiempo**: Incluye herramientas para manejar datos de series temporales, como reindexado, rellenado e interpolación.
- **Interfaces con otros lenguajes estadísticos**: Facilita la importación y exportación de datos desde y hacia otros entornos estadísticos, como R, lo que amplía la versatilidad del análisis de datos.

Seaborn

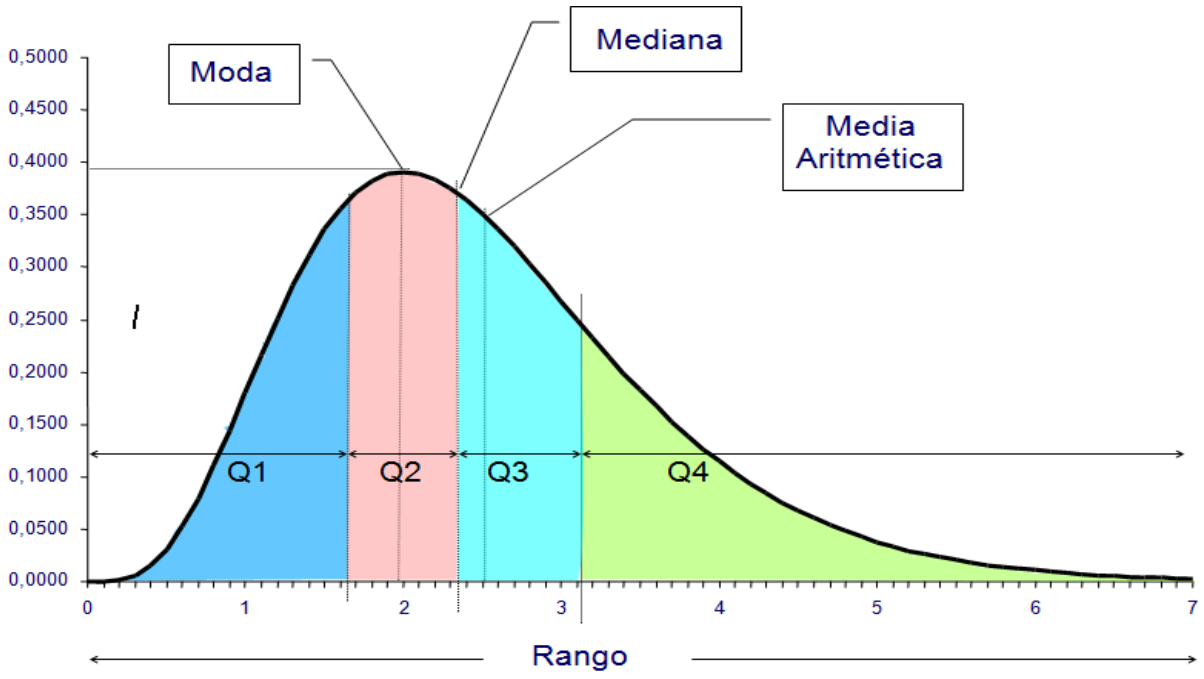
Seaborn es una **librería de visualización de datos basada en Matplotlib**, que se centra específicamente en la **visualización de datos estadísticos**. Algunas de sus características son:

- **Correlación y pruebas estadísticas**: Incluye funciones que permiten visualizar relaciones entre variables, así como realizar pruebas estadísticas.
- **Estadísticas enmascaradas**: Permite representar datos complejos de manera sencilla, manteniendo claro el análisis subyacente.
- **Estimación de densidad de kernel**: Facilita la creación de gráficos que muestran la estimación de densidad para variables continuas, lo que ayuda a entender mejor la distribución de los datos.
- **Funcionalidad cuasi-Monte Carlo**: Se pueden realizar simulaciones para evaluar distribuciones en situaciones específicas.

Seaborn es ideal para crear visualizaciones efectivas que ayuden a comunicar resultados estadísticos y patrones en los datos de forma clara.

2. Medidas de Tendencia Central

Las medidas de tendencia central son fundamentales en estadística, ya que nos permiten resumir un conjunto de datos mediante un valor que representa el centro de la distribución. Las tres medidas más comunes son la **media**, la **mediana** y la **moda**.



Fuente: Allende H y Ahumada S, ILI-280

Media

La media, o **promedio**, es la suma de todos los valores de un conjunto de datos dividida por la cantidad total de valores. Se expresa de la siguiente manera:

$$\text{Media} = \frac{\sum_{i=1}^{n} x_i}{n}$$

Por ejemplo, si tenemos las notas de cinco alumnos: 4, 7, 5, 9 y 6, la media sería:

$$\text{Media} = \frac{4 + 7 + 5 + 9 + 6}{5} = \frac{31}{5} = 6.2$$

En Python, podríamos calcular la media con Numpy, de la siguiente manera:

```
import numpy as np

notas = [4, 7, 5, 9, 6]
media = np.mean(notas)
print(f"La media es: {media}")
```

Mediana

La mediana es el valor que se encuentra en el medio de un conjunto de datos cuando están ordenados. Si el número de observaciones es impar, la mediana es el valor central; si es par, se promedian los dos valores centrales.

Siguiendo nuestro ejemplo:

1. Ordenamos las notas: *4, 5, 6, 7, 9.*
2. La mediana (quinto valor, ya que hay cinco registros) es *6*.

Si tuviéramos un conjunto con seis notas (*4, 5, 6, 7, 8, 9*), ordenaríamos y tendríamos *6* y *7* como valores centrales.

Entonces:

$$\text{Mediana} = \frac{6 + 7}{2} = 6.5$$

En Python, esto se calcularía así:

```
mediana = np.median(notas)
print(f"La mediana es: {mediana}")
```

Moda

La moda es el valor que aparece con mayor frecuencia en un conjunto de datos. Por ejemplo, en las notas *4, 7, 5, 7* y *6*, el *7* es la moda, ya que aparece más veces que los demás valores.

Para calcular la moda en Python, utilizaríamos la librería scipy:

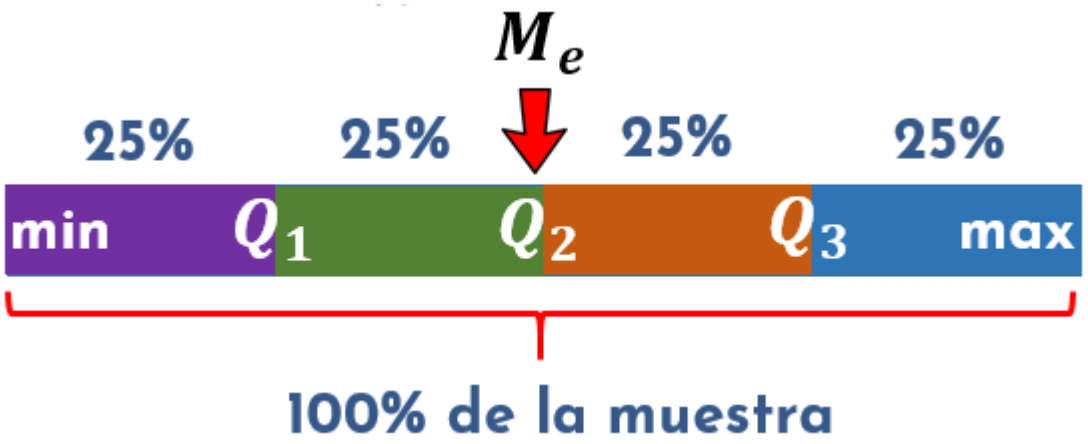
```
from scipy import stats

notas = [4, 7, 5, 7, 6]
moda = stats.mode(notas)[0]
print(f"La moda es: {moda}")
```

Cuartiles y Percentiles

Tanto los cuartiles como los percentiles son **herramientas estadísticas valiosas para resumir dataset**s, identificar outliers (valores atípicos) y entender la variabilidad en los datos. Estas medidas permiten que los analistas tomen decisiones, al señalar dónde se encuentran los valores extremos y cómo se distribuyen en relación con el conjunto total.

Cuartiles



Los cuartiles dividen un **conjunto de datos en cuatro partes iguales**, con el objetivo de facilitar la interpretación de la distribución de los valores. Se representan como Q_1, Q_2 y Q_3:

- Q_1 (primer cuartil): Este valor es el punto que divide el primer 25% de los datos. En otras palabras, el 25% de los datos queda por debajo de este cuartil.
- Q_2 (segundo cuartil o mediana): Este es el valor que se encuentra en el medio del conjunto de datos cuando están ordenados. Distribuye el conjunto en dos mitades, donde el 50% de los datos son menores y el 50% son mayores.
- Q_3 (tercer cuartil): Representa el punto que divide el 75% de los datos, por lo que el 75% de los valores están por debajo de este cuartil.

Ejemplo

Supongamos que tenemos el siguiente conjunto de datos que representa las puntuaciones de un grupo de alumnos en un examen:

56,74,68,82,90,62,78,84,88,9556,74,68,82,90,62,78,84,88,95

Para calcular los cuartiles de este conjunto de datos, primero debemos ordenarlos:

56,62,68,74,78,82,84,88,90,9556,62,68,74,78,82,84,88,90,95

- Q_1 sería el valor en el 25% de los datos, que en este caso es **68**.
- Q_2 o mediana sería **78**.
- Q_3 sería **88**.

Con estos datos, podemos observar que el 25% de los alumnos obtuvo menos de 68 puntos, el 50% menos de 78, y el 75% menos de 88.

Cálculo en Python

Para calcular cuartiles en Python, usaremos la librería NumPy. A continuación, se muestra un ejemplo de cómo hacerlo:

```
import numpy as np

# Datos de puntuaciones
puntuaciones = [56, 74, 68, 82, 90, 62, 78, 84, 88, 95]

# Cálculo de cuartiles
Q1 = np.percentile(puntuaciones, 25)
Q2 = np.percentile(puntuaciones, 50)  # Mediana
Q3 = np.percentile(puntuaciones, 75)

print(f"Primer cuartil (Q1): {Q1}")
print(f"Segundo cuartil (Q2 - Mediana): {Q2}")
print(f"Tercer cuartil (Q3): {Q3}")
```

Rango Intercuartil

El rango intercuartil (RIC o IQR) es una medida de la dispersión de la mitad central de un conjunto de datos. Se calcula restando el primer cuartil (Q_1) del tercer cuartil (Q_3).

- Es una medida de variabilidad.
- Es una buena medida de dispersión para distribuciones sesgadas.
- Se puede visualizar claramente mediante el cuadro en un diagrama de caja.

Usos del IQR

- Es la mejor medida de variabilidad para distribuciones asimétricas o conjuntos de datos con valores atípicos.
- Se utiliza para detectar valores atípicos.

Cálculo del IQR

- El IQR se calcula con la fórmula: $IQR = Q_3 - Q_1$.

BoxPlot

Un boxplot, también conocido como **diagrama de caja y bigotes**, es un gráfico que muestra cómo se distribuyen los datos numéricos.

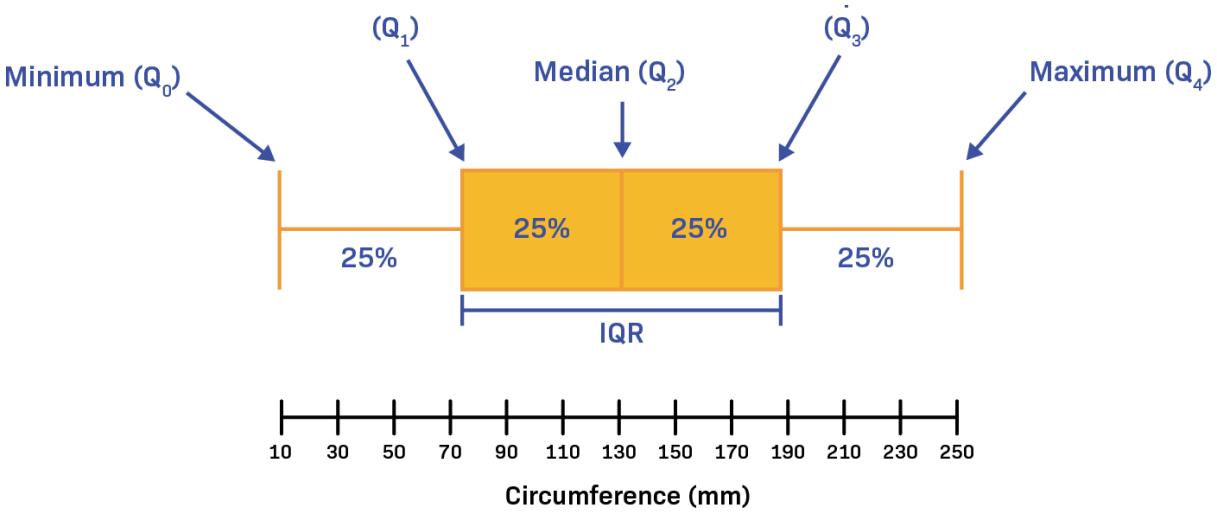
- Muestra la mediana y los cuartiles de los datos
- Identifica valores atípicos
- Permite intuir la morfología y simetría de los datos
- Se puede dibujar más de un diagrama de caja por gráfico

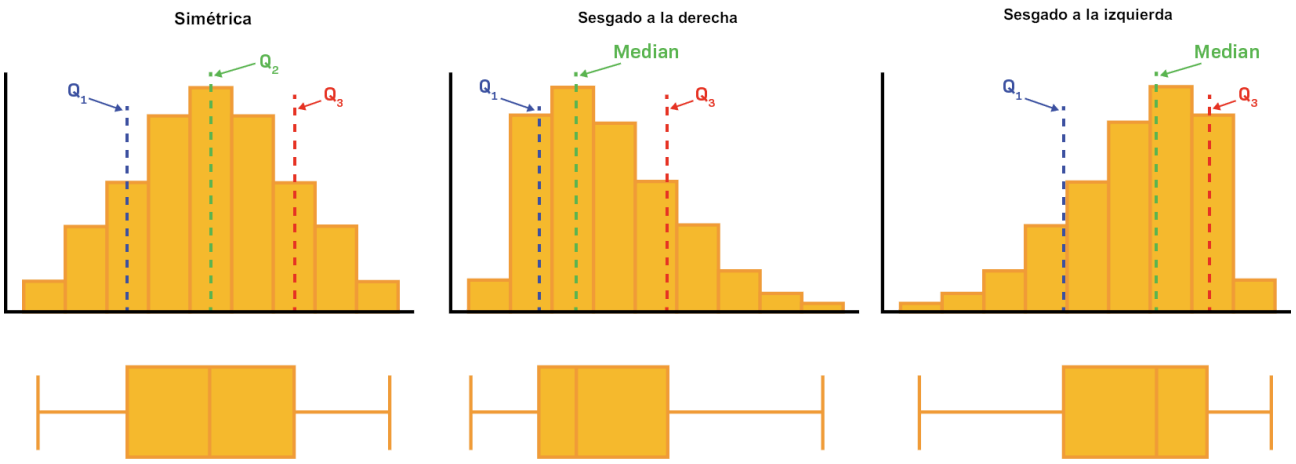
Componentes:

- Cajas: Representan el rango intercuartílico (IQR) de los datos
- Bigotes: Son líneas que se extienden desde ambos extremos de la caja.
- Valores atípicos: Son puntos fuera de los bigotes considerados inusuales o extremos
- Gorros: Son líneas perpendiculares en los extremos de los bigotes

Usos

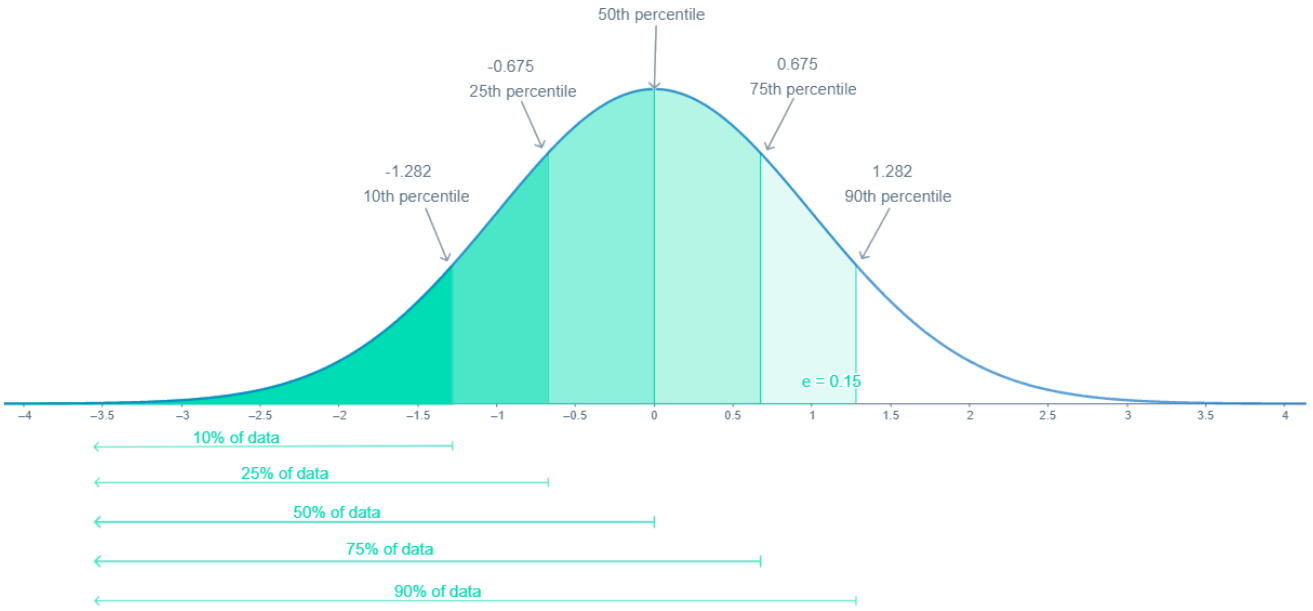
- Es ideal para comparar distribuciones entre grupos
- Es una forma estandarizada para representar gráficamente una serie de datos numéricos
- Es una forma útil de comparar diferentes conjuntos de datos





Percentiles

Los percentiles son similares a los cuartiles, pero **dividen el conjunto de datos en 100 partes iguales**.



Por lo tanto, cada percentil indica el valor bajo el cual se encuentra un cierto porcentaje de los datos.

- P_{25}: Corresponde al primer cuartil, es decir, el 25% de los datos están por debajo de este valor.
- P_{50}: Es la mediana, donde el 50% de los datos son menores que este valor.
- P_{75}: Corresponde al tercer cuartil, donde el 75% de los datos están por debajo.

Ejemplo

Volvemos a usar el conjunto de datos anterior:

56,62,68,74,78,82,84,88,90,9556,62,68,74,78,82,84,88,90,95

Para calcular los percentiles, buscamos el valor que divide el conjunto en diferentes porcentajes.

Cálculo en Python

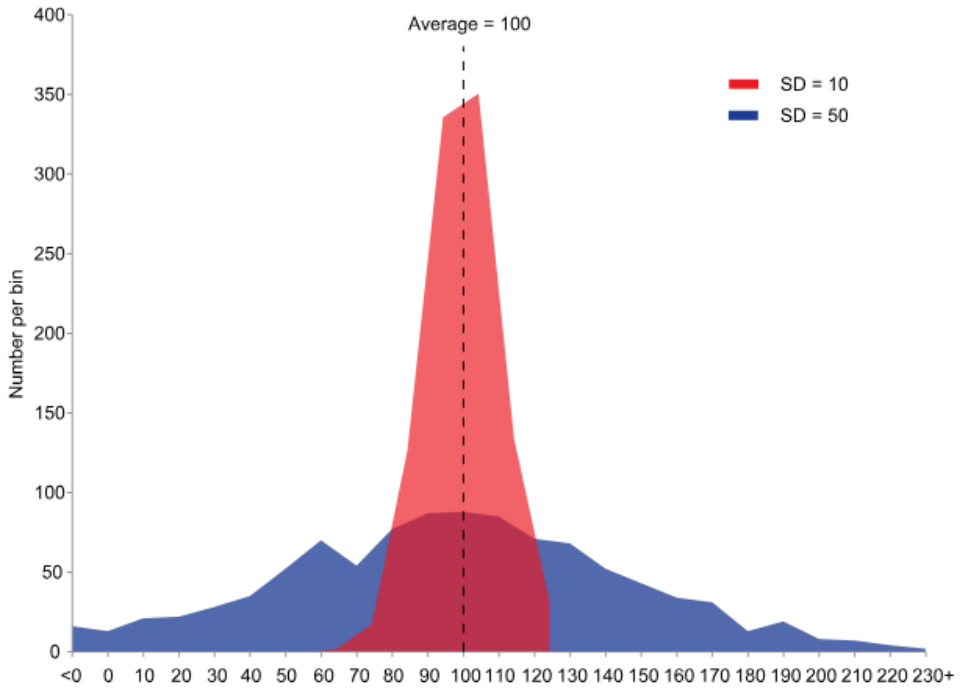
Para calcular percentiles, también utilizamos NumPy:

```
# Cálculo de percentiles
P25 = np.percentile(puntuaciones, 25)
P50 = np.percentile(puntuaciones, 50)  # Mediana
P75 = np.percentile(puntuaciones, 75)

print(f"Percentil 25 (P25): {P25}")
print(f"Percentil 50 (P50 - Mediana): {P50}")
print(f"Percentil 75 (P75): {P75}")
```

3. Medidas de Dispersión

Las medidas de dispersión nos brindan información sobre la variabilidad de los datos, es decir, cuán alejados están estos valores de la tendencia central. Las más



utilizadas son el **rango**, la **varianza** y la **desviación standard**.

Rango

El rango es la **diferencia entre el valor máximo y el mínimo de un conjunto de datos**. De nuestra lista de notas (*4, 7, 5, 9, 6*):

$$\text{Rango} = \text{Valor máximo} - \text{Valor mínimo} = 9 - 4 = 5$$

En Python, podemos calcularlo así:

```
rango = np.max(notas) - np.min(notas)
print(f"El rango es: {rango}")
```

¿Qué significa “desviación”?

Para poder definir las siguientes medidas, debemos conocer primero qué significa **“desviación”** en estadística. Se trata de la **diferencia entre cada dato y la media aritmética o promedio**.

Desviación media

Es el promedio de los valores absolutos de todas las desviaciones.

Varianza

La varianza mide cuán dispersos están los valores respecto a la media. Se calcula como el promedio de las diferencias al cuadrado entre cada valor y la media:

$$\text{Varianza} = \frac{\sum_{i=1}^{n}(x_i - \text{media})^2}{n}$$

Siguiendo nuestro ejemplo de notas, si la media es 6.2, calculamos la varianza:

1. Restamos la media y elevamos al cuadrado cada resultado.
2. Luego sumamos y dividimos entre el número total de valores.

En Python, la varianza puede calcularse con:

```
varianza = np.var(notas)
print(f"La varianza es: {varianza}")
```

Desviación Standard

La desviación standard es la **raíz cuadrada de la varianza**, proporcionando una medida de dispersión en las mismas unidades que los datos originales. Se calcula de la siguiente manera:

$$\text{Desviación Estándar} = \sqrt{\text{Varianza}}$$

Para calcularla en Python, utilizamos:

```
desviacion_estandar = np.std(notas)
print(f"La desviación estándar es: {desviacion_estandar}")
```

Reflexión final

En esta clase, abordamos conceptos clave de la **estadística descriptiva** que son esenciales en el análisis de datos con Python. Con estos fundamentos, tus habilidades en data analytics se verán fortalecidas, permitiéndote realizar un análisis más profundo y significativo de conjuntos de datos. No olvides practicar estos cálculos utilizando tus propios conjuntos de datos para afianzar tu comprensión.

Materiales y recursos adicionales

Módulo scipy.stats

Próximos Pasos

- Técnicas de análisis exploratorio (EDA): visualización y resumen de datos.

Ejercicios Prácticos

Actividad 1: Análisis de Tendencia Central

Contexto

En esta nueva semana de pasantía en SynthData. Silvia, la Project Manager y Data Scientist, te acaba de asignar una tarea crucial para conocer mejor los datos de clientes que la empresa ha recopilado. Tenés que calcular las medidas de tendencia central de las ventas mensuales de un producto específico en el último año. Esto les ayudará a identificar patrones y tomar decisiones informadas sobre el inventario.

Objetivos

- Calcular la media, la mediana y la moda de las ventas mensuales.
- Comprender la importancia de estas medidas en el análisis de datos de ventas.

Ejercicio práctico

Realiza los siguientes cálculos en Python con los datos sobre las ventas mensuales en unidades del último año:

1. Calcular la media de las ventas.
2. Calcular la mediana de las ventas.
3. Calcular la moda de las ventas.

Interpretá y explicá qué significan estos valores.

Set de datos

```
ventas = {
    'Mes': ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio', 'Julio', 'Agosto', 'Septiembre', 'Octubre', 'Noviembre', 'Diciembre'],
    'Ventas (millones)': [1.2, 2.5, 3.1, 18.3, 40.5, 52.1, 54.8, 46.2, 25.5, 13.8, 11.9, 9.2]
}
```

¿Por qué importa esto en SynthData?

Las medidas de tendencia central son esenciales para entender el rendimiento de un producto en el mercado. Ayudan a identificar si las ventas están creciendo, estables o en declive, permitiendo tomar decisiones estratégicas para el negocio.

Actividad 2: Medidas de Dispersión

Contexto

Matías, el Data Analyst, te solicita que analices la variabilidad de las ventas mensuales que calculaste en la actividad anterior. Este análisis es fundamental para saber si las fluctuaciones en las ventas podrían afectar los pronósticos futuros. Así, podrás entender mejor el comportamiento del producto en distintos meses del año.

Objetivos

- Calcular el rango, la varianza y la desviación standard de las ventas mensuales.
- Reconocer cómo estas medidas de dispersión informan sobre la estabilidad de las ventas.

Ejercicio práctico

Utiliza el mismo conjunto de datos de ventas mensuales del ejercicio anterior y realiza lo siguiente en Python:

1. Calcular el rango de las ventas.
2. Calcular la varianza de las ventas.
3. Calcular la desviación standard de las ventas.

Teorizá a partir de las dos actividades: Si tuvieras que adivinar de qué producto se trata, ¿cuál crees que se ajusta más a los valores obtenidos? Justificá tu elección.

[ ] Leche

[ ] Bufanda

[ ] Árbol de navidad

[ ] Kit de geometría escolar

[ ] Bronceador

¿Por qué importa esto en SynthData?

Las medidas de dispersión son esenciales para evaluar el riesgo y la estabilidad de las ventas. Un alto rango, varianza o desviación standard puede indicar comportamientos erráticos, lo que podría afectar la planificación del inventario y ventas futuras.

⛔ **Estos ejercicios son una simulación de cómo se podría resolver el problema en este contexto específico. Las soluciones encontradas no aplican de ninguna manera a todos los casos.**

Recuerda que las soluciones dependen de los sets de datos, el contexto y los requerimientos específicos de los stakeholders y las organizaciones.


Buenos Aires
aprende
Agencia de Habilidades para el Futuro
Buenos
Aires
Ciudad