ACT M1.2

August 15, 2024

1 ACTIVIDAD M1.2 Datos faltantes y Outliers

Agosto 13, 2024

A01285158 | Grace Aviance Silva Arostegui

A01640495 | Carlos Alberto Sánchez Villanueva

Utiliza la variable absences y traveltime del archivo: student-mat_v3.csv

Download student-mat_v3.csvpara realizar lo siguiente:

1.Identificar el porcentaje de datos faltantes.

2.Identificar el mecanismo que ocasiona datos faltantes (MCAR, MAR, NMAR)}

3.Obtener estadísticas descriptivas de los datos (histograma, media, desviación estándar, mediana, moda, etc).

4.Utilizar el método de imputación adecuado para cada una de las variables con datos faltantes.

- Imputación Simple: Media, Mediana, Moda

5.Realizar un boxplot e interpretarlo.

[ ]:
```
import pandas as pd
import statistics
import matplotlib.pyplot as plt
import numpy as np

df = pd.read_csv("student-mat_v3.csv")
traveltime = df["traveltime"]
absences = df["absences"]
df = df.select_dtypes(include='number')
```

1.Identificar el porcentaje de datos faltantes.

[ ]:
```
faltante_traveltime = traveltime.isnull().mean() *100
print("El porcentaje de valores faltantes en la columna de traveltime es:",␣
 ↪faltante_traveltime)

faltante_absences = absences.isnull().mean() * 100
```

```
print("El porcentaje de valores faltantes en la columna de absences es:",␣
 ↪faltante_absences)
```

```
El porcentaje de valores faltantes en la columna de traveltime es:
6.582278481012659
El porcentaje de valores faltantes en la columna de absences es:
5.3164556962025316
```

2.Identificar el mecanismo que ocasiona datos faltantes (MCAR, MAR, NMAR)

- MCAR(Missing completely at Random): la probabilidad de observar el dato faltante es completamente al azar.
- MAR(Missing at Random): la probabilidad de observar el dato faltante no depende de la variable, pero sí de otra variable.
- NMAR(Not missing at Random): la probabilidad de observar el dato faltante depende de la misma variable

[ ]:
```
correlaciones_traveltime = df.corr()['traveltime']
correlaciones_traveltime.sort_values(ascending=False).drop('traveltime')
```

[ ]:
```
Walc         0.121412
Dalc         0.117571
age          0.111686
failures     0.092601
famrel       0.031971
goout        0.007979
G1           0.002483
health      -0.004121
freetime    -0.013955
studytime   -0.039664
absences    -0.039823
G3          -0.065545
G2          -0.071332
Fedu        -0.114431
Medu        -0.141022
Name: traveltime, dtype: float64
```

[ ]:
```
correlaciones_absences = df.corr()['absences']
correlaciones_absences.sort_values(ascending=False).drop('absences')
```

[ ]:
```
age          0.172533
Walc         0.116591
Medu         0.102713
Dalc         0.076914
G3           0.048861
Fedu         0.029680
goout        0.023175
failures     0.013126
G1          -0.010803
```

```
G2           -0.017989
health       -0.020160
traveltime   -0.039823
famrel       -0.044130
freetime     -0.062170
studytime    -0.063694
Name: absences, dtype: float64
```

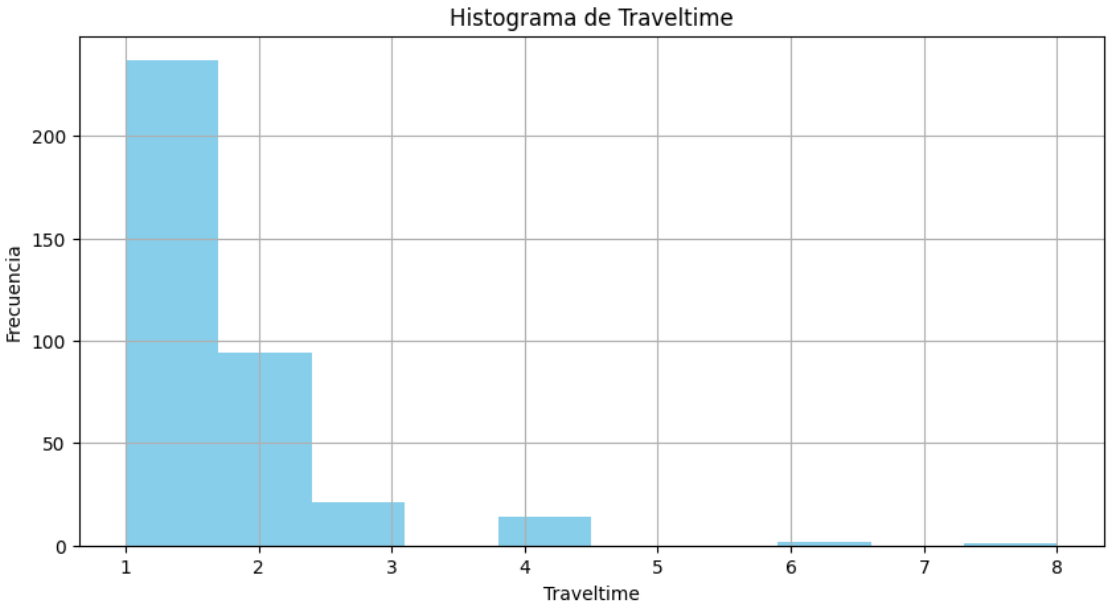
Dado que las correlaciones en ambas columnas son mínimas, podemos decir que no es MAR (Missing at Random). Entonces podemos concluir que el mecanismo que ocasiona datos faltantes puede ser MCAR (Missing completely at Random) o NMAR (Not missing at Random)

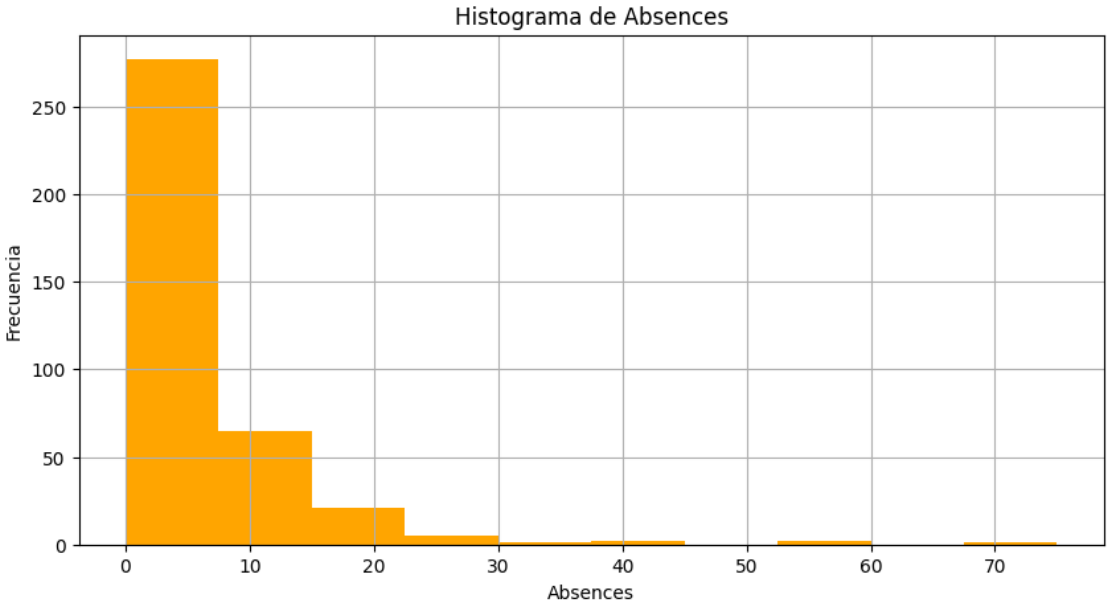
3.Obtener estadísticas descriptivas de los datos (histograma, media, desviación estándar, mediana, moda, etc).

[ ]:
```
# Primeramente eliminar los valores NaN de las columnas
traveltime = traveltime[~np.isnan(traveltime)]
absences = absences[~np.isnan(absences)]
```

Histogramas

[ ]:
```
# Para 'traveltime'
plt.figure(figsize=(10, 5))
plt.hist(traveltime, color='skyblue')
plt.title('Histograma de Traveltime')
plt.xlabel('Traveltime')
plt.ylabel('Frecuencia')
plt.grid(True)
plt.show()

# Para 'absences'
plt.figure(figsize=(10, 5))
plt.hist(absences, color='orange')
plt.title('Histograma de Absences')
plt.xlabel('Absences')
plt.ylabel('Frecuencia')
plt.grid(True)
plt.show()
```





Media

[ ]:
```
traveltime_media = traveltime.mean()
print("Media de traveltime = ", traveltime_media)

absences_media = absences.mean()
print("Media de absences = ", absences_media)
```

```
Media de traveltime =  1.5284552845528456
Media de absences =  5.542780748663102
```

Varianza

[ ]:
```
traveltime_varianza = np.var(traveltime, ddof = 1)
print("La varianza de traveltime es = ", traveltime_varianza)

absences_varianza = np.var(absences, ddof = 1)
print("La varianza de absences es = ", absences_varianza)
```

```
La varianza de traveltime es =  0.8150848356309651
La varianza de absences es =  65.43382173732276
```

Desviacion estandar

[ ]:
```
traveltime_DesvStd = np.sqrt(traveltime_varianza)
print("La desviación estándar de traveltime es = ", traveltime_DesvStd)

absences_DesvStd = np.sqrt(absences_varianza)
print("La desviación estándar de absences es = ", absences_DesvStd)
```

```
La desviación estándar de traveltime es =  0.9028204891510633
La desviación estándar de absences es =  8.08911748816413
```

Moda

[ ]:
```
traveltime_moda = statistics.mode(traveltime)
print("La desviación estándar de traveltime es = ", traveltime_moda)

absences_moda = statistics.mode(absences)
print("La desviación estándar de absences es = ",absences_moda)
```

```
La desviación estándar de traveltime es =  1.0
La desviación estándar de absences es =  0.0
```

Mediana

[ ]:
```
traveltime_mediana = np.median(traveltime)
print("La de mediana de la columna traveltime es: ", traveltime_mediana)

absences_mediana = np.median(absences)
print("La de mediana de la columna absences es: ", absences_mediana)
```

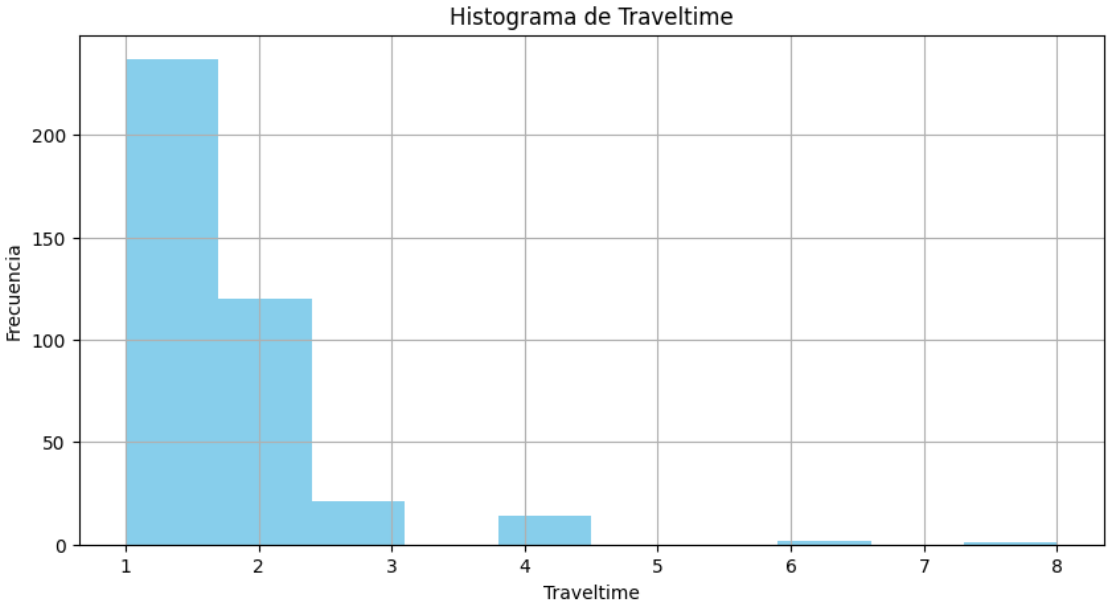
```
La de mediana de la columna traveltime es:  1.0
La de mediana de la columna absences es:  3.5
```

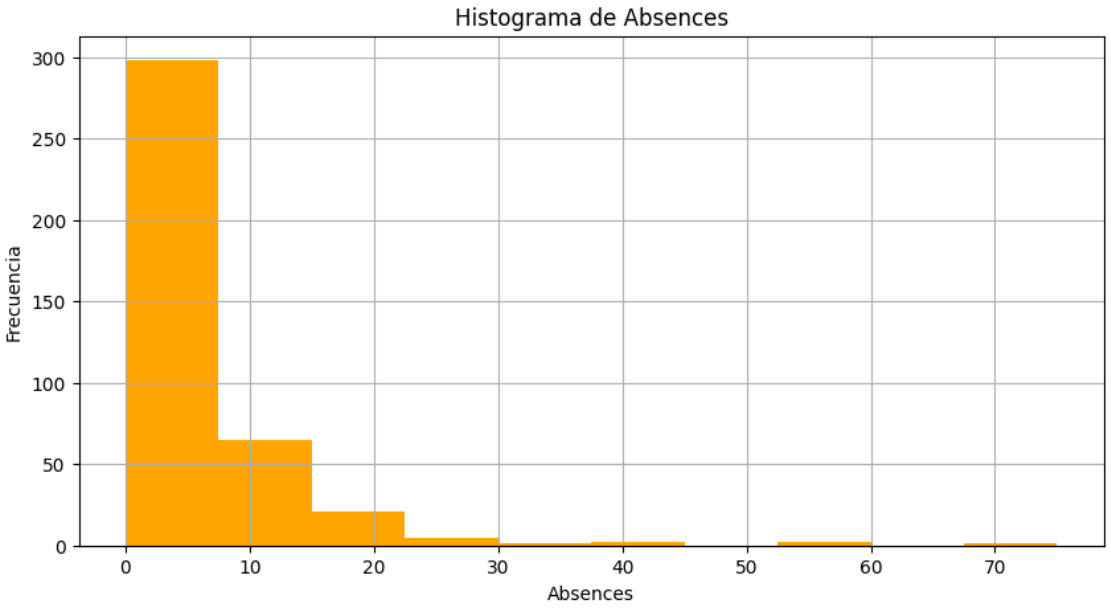
4.Utilizar el método de imputación adecuado para cada una de las variables con datos faltantes.

- Imputación Simple: Media, Mediana, Moda

[ ]:
```
clean_dataframe= df

clean_dataframe['traveltime'] = clean_dataframe['traveltime'].
 ↪fillna(round(clean_dataframe['traveltime'].mean())) # Imputando los valores␣
 ↪perdidos por la mediana

clean_dataframe['absences'] = clean_dataframe['absences'].
 ↪fillna(round(clean_dataframe['absences'].mean())) # Imputando los valores␣
 ↪perdidos por la mediana
```

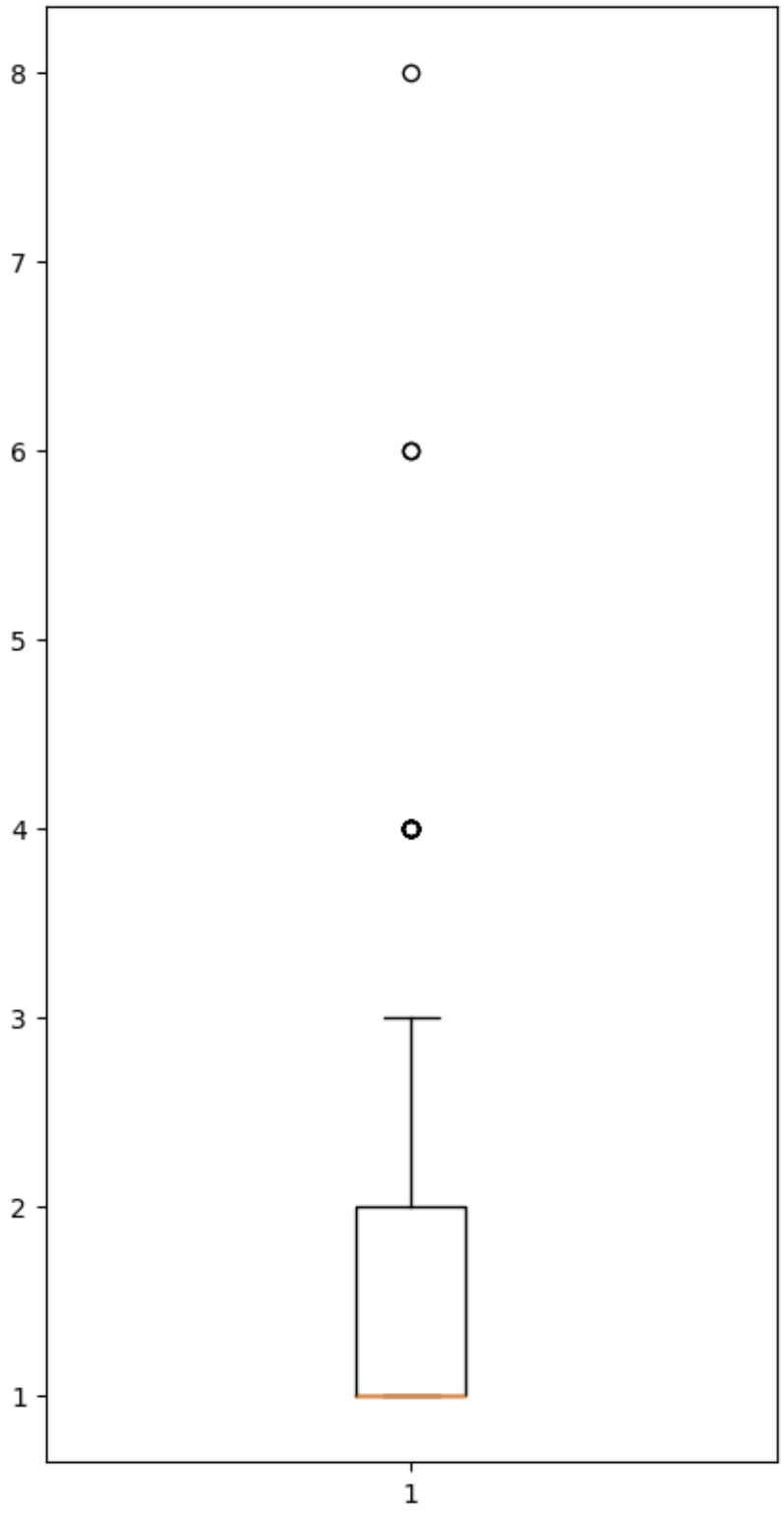
[ ]:
```
plt.figure(figsize=(10, 5))
plt.hist(clean_dataframe['traveltime'], color='skyblue')
plt.title('Histograma de Traveltime')
plt.xlabel('Traveltime')
plt.ylabel('Frecuencia')
plt.grid(True)
plt.show()

# Crear el histograma para 'absences'
plt.figure(figsize=(10, 5))
plt.hist(clean_dataframe['absences'], color='orange')
plt.title('Histograma de Absences')
plt.xlabel('Absences')
plt.ylabel('Frecuencia')
plt.grid(True)
plt.show()
```





5.Realizar un boxplot e interpretarlo.

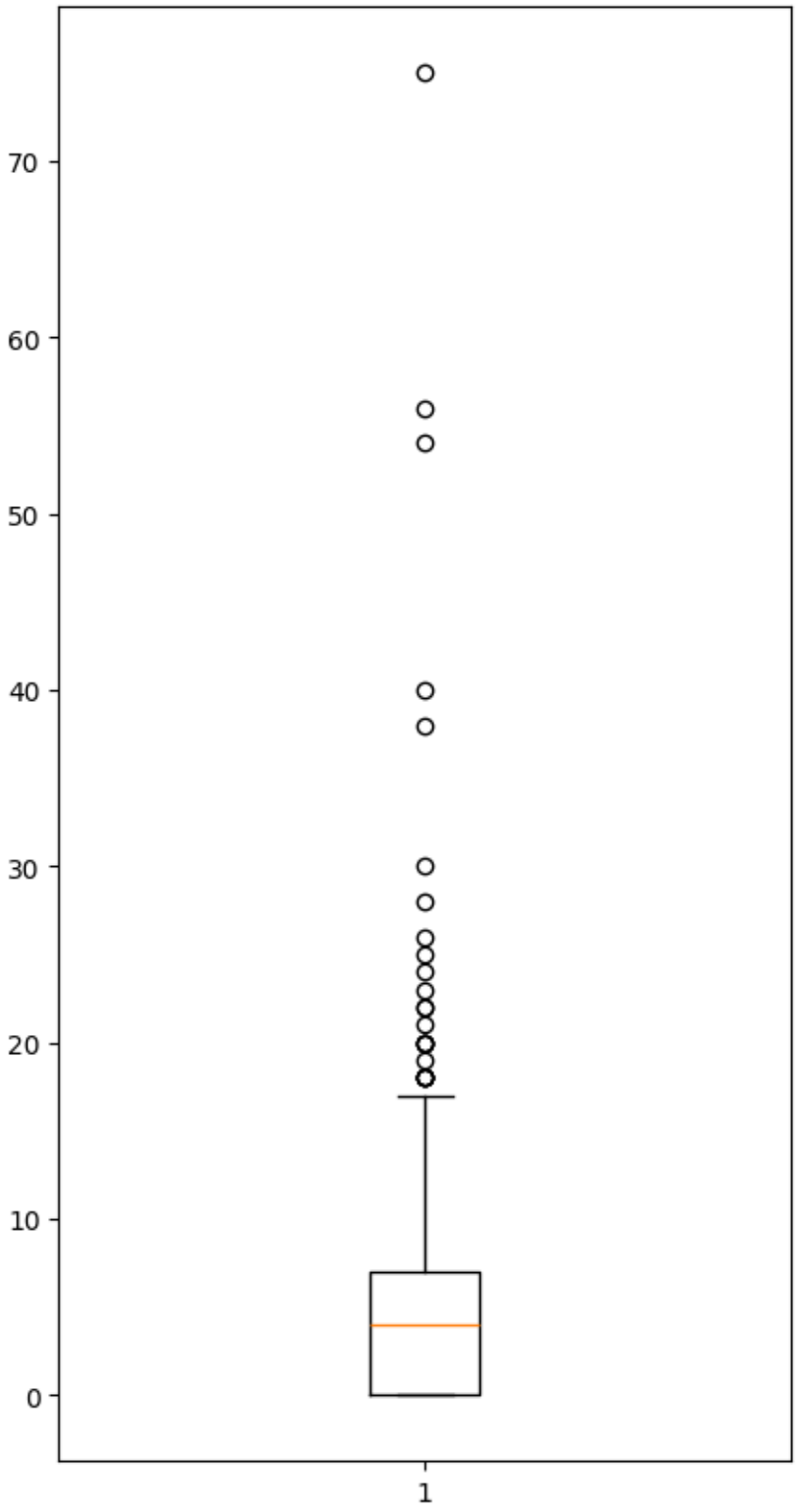
[ ]:
```
fig = plt.figure(figsize =(5, 10))
plt.boxplot(clean_dataframe['traveltime'])
plt.show()
```



Los datos en este gráfico tienen una mediana en torno a 2 o 3, con la mayoría de los datos concen-

trados entre 0 y 5. Sin embargo, hay muchos valores atípicos, lo que indica una posible distribución sesgada hacia valores altos.

[ ]:
```
fig = plt.figure(figsize =(5, 10))
plt.boxplot(clean_dataframe['absences'])
plt.show()
```



Los datos en este gráfico tienen una mediana en 2, con la mayoría de los datos concentrados entre

1 y 3.No se encuentran valores outliers dentro de los bigotes. Sin embargo, hay algunos valores atípicos más altos (4, 6, y 8).