

Transformación de datos

Agregación de nuevas columnas en el csv: Se decidió por agregar 2 columnas, condicionales, llamadas "Performace_Math", "Performace_Reading" y "Performance_Writing" que van a tener categóricamente el desempeño de los estudiantes divididos en 33,67,87 y de 87 para 100 (En el caso de writing las decenas van a estar en cientos) con las categorías Low, Medium, Medium-High y High respectivamente. Además, se decidió llamar la columna en inglés para coincidir y que tenga coherencia en el csv original.

Se consideró sumar los puntajes de la prueba de matemáticas junto con la de escritura, sin embargo, no se hizo debido a que no tenemos información concreta de que tan bien vaya a ir la suma de estas dos columnas para ser evaluadas categóricamente. Por lo tanto, se hizo la categorización (Low, Medium, Medium-High y High) de manera separada

También en la columna "race/ethnicity" se observó que hay filas en blanco, y esto hace que el análisis no sea completo, sin embargo, se optó por no "rellenar" estas filas porque no sabemos si están nulas porque ahí va "grupo A" o "grupo B" o en dado caso va un posible "grupo c". Por esta razón no se rellenaron las columnas en blanco con, por ejemplo, "grupo C"; Y tampoco se decidió eliminarlas porque pueden ser de utilidad en la parte de minería de datos

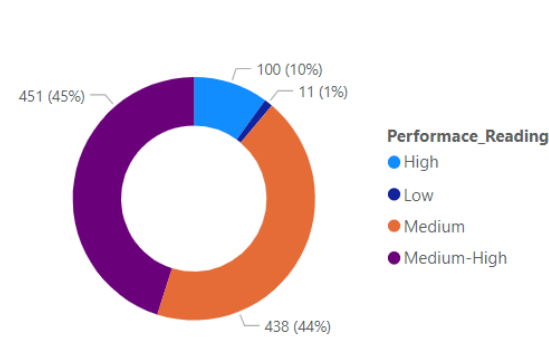
Visualización

El campo índice representa la cantidad de estudiantes, es por eso que se uso el recuento en todas las gráficas. De esta manera hace el recuento del total correcto 1000

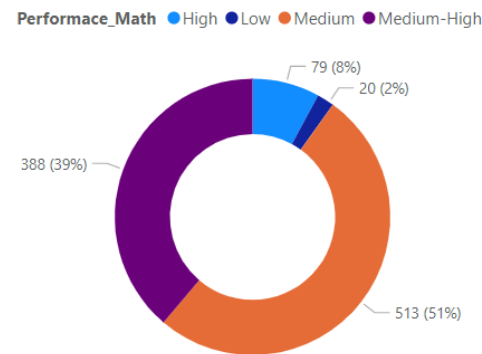


Represento los datos math y Reading performance en gráficos tipo anillo, para mostrar claramente la cantidad en valor y porcentaje de cada resultado

READING PERFORMANCE



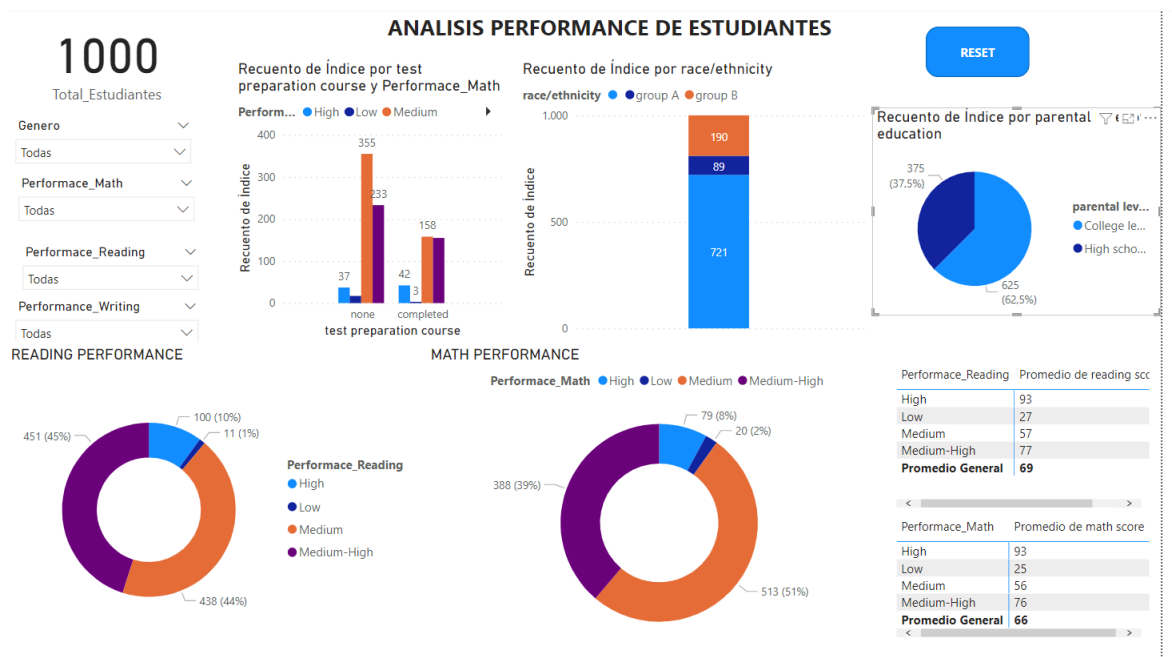
MATH PERFORMANCE



Con una matriz se muestra el promedio aproximado de cada una de las categorías que representan las calificaciones y un promedio general para los campos Performance Reading y Performance Math

Performace_Reading	Promedio de reading scc
High	93
Low	27
Medium	57
Medium-High	77
Promedio General	69

Performace_Math	Promedio de math score
High	93
Low	25
Medium	56
Medium-High	76
Promedio General	66



Para obtener el análisis primero se realiza el import de las librerías para visualización y minería de datos. Luego se carga el dataset

```
[50] import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from matplotlib.patches import Circle
import numpy as np

[2] df = pd.read_csv('StudentsPerformance.csv')
```

Después de realizado el paso anterior ejecutamos la función head() con la cual obtenemos las 5 primeras filas del dataset dándonos una muestra rápida de cómo está conformado el dataset

df.head()

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	College level	standard	none	72	72	74.0
1	female	NaN	College level	standard	completed	69	90	88.0
2	female	group B	College level	standard	none	90	95	93.0
3	male	group A	College level	free/reduced	none	47	57	44.0
4	male	NaN	College level	standard	none	76	78	75.0

Mediante la función info() podemos observar la estructura del dataset, los campos por los cual está compuesto y su respectivo tipo de dato. También nos da información de la cantidad de entradas que existen actualmente en el dataset

df.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 8 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   gender                                986 non-null    object
1   race/ethnicity                        279 non-null    object
2   parental level of education           1000 non-null   object
3   lunch                                1000 non-null   object
4   test preparation course               1000 non-null   object
5   math score                           1000 non-null   int64
6   reading score                        1000 non-null   int64
7   writing score                         990 non-null    float64
dtypes: float64(1), int64(2), object(5)
memory usage: 62.6+ KB
```

La función `columns` nos da los nombres de las columnas del dataset. De esta manera podemos saber con exactitud el nombre y además si existe algún campo en blanco (espacio en blanco dentro del nombre) que no hemos visto

```
[5] df.columns  
  
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',  
      'test preparation course', 'math score', 'reading score',  
      'writing score'],  
      dtype='object')
```

La función `isnull` es usada con el complemento `sum()` para ver la cantidad de filas nulas por campos en el dataset.

se detectó, con la función `isnull()`. `sum` que los campos: `gender`, `race/ethnicity` y `writing score` tienen ítems nulos

```
df.isnull().sum()  
  
gender                14  
race/ethnicity        721  
parental level of education    0  
lunch                 0  
test preparation course    0  
math score            0  
reading score          0  
writing score         10  
dtype: int64
```

Con la función `describe()` se obtienen los valores de medida (promedio, moda, suma etc.) de las variables numéricas. Esta información nos ayudara para las gráficas de boxplot

df.describe()


	math score	reading score	writing score
count	1000.00000	1000.000000	990.000000
mean	66.08900	69.169000	68.093939
std	15.16308	14.600192	15.231621
min	0.00000	17.000000	10.000000
25%	57.00000	59.000000	58.000000
50%	66.00000	70.000000	69.000000
75%	77.00000	79.000000	79.000000
max	100.00000	100.000000	100.000000


Con la información del paso anterior de la función isnull, se decide como solución llenar las filas nulas del dataset para que estas no afecten el análisis según lo recomendado en clase. No se pudo eliminar toda la columna race/ethnicity debido a la importancia de estos datos en el análisis.

Para las variables categóricas se usó la moda para rellenar los valores nulos. Para los datos numéricos se usó el promedio para rellenar los

```
[8] mode_genre = df['gender'].mode()[0]#rellenar los valores nulos de genero
[9] df['gender'].fillna(mode_genre,inplace=True)
[10] mode_race = df['race/ethnicity'].mode()[0]#rellenar los valores nulos de raza
[11] df['race/ethnicity'].fillna(mode_race,inplace=True)
[12] mean_writing = df['writing score'].mean()#rellenar los valores nulos de writing score
[13] df['writing score'].fillna(mean_writing,inplace=True)
```



Se puede observar que después de usar los métodos fillna ya no existen valores nulos


 `df.isnull().sum()`

 `gender` `0`
`race/ethnicity` `0`
`parental level of education` `0`
`lunch` `0`
`test preparation course` `0`
`math score` `0`
`reading score` `0`
`writing score` `0`
`dtype: int64`

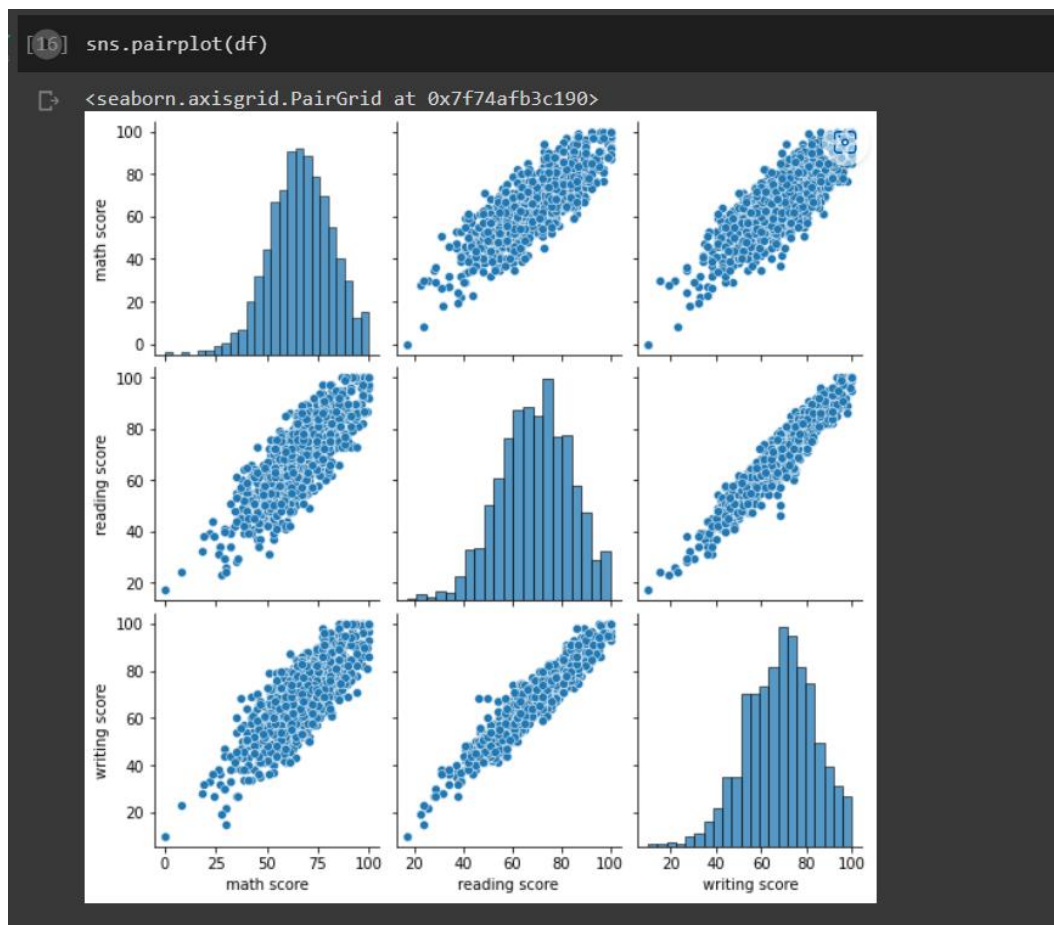
En las gráficas que representa los valores del campo genero primero se hizo un recuento

GENERO

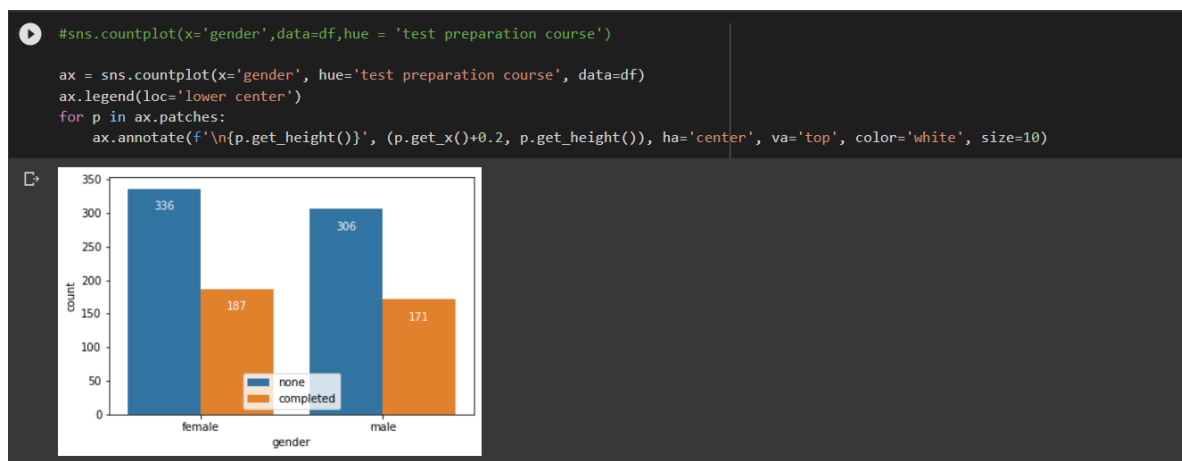
  `df['gender'].value_counts()`

 `female` `523`
`male` `477`
`Name: gender, dtype: int64`

Se realizó con la función pairplot un estudio de la distribución de los valores de manera general del dataset



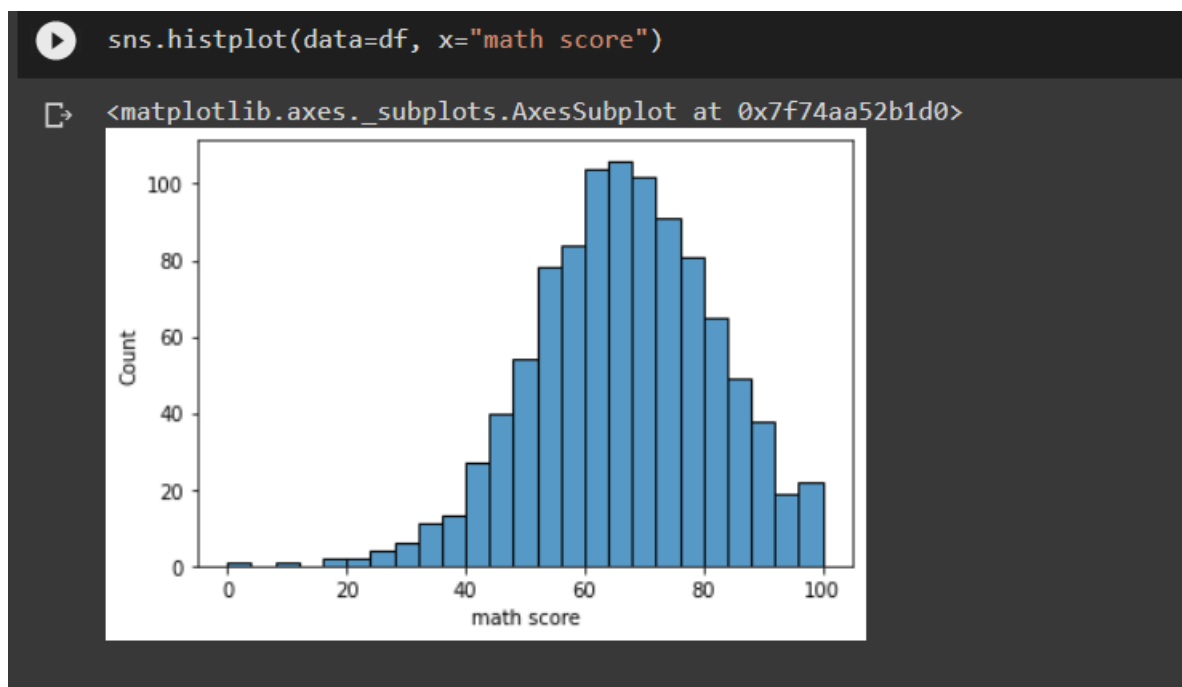
Usando la librería seaborn, se crea la gráfica de conteo donde se representa por genero la cantidad de estudiantes que realizaron un test de preparación mostrando los que completaron y no completaron el test



Con la información del dataset respectivo a cada raza, se hizo una gráfica para ver la distribución de la cantidad de estudiantes por raza A o B



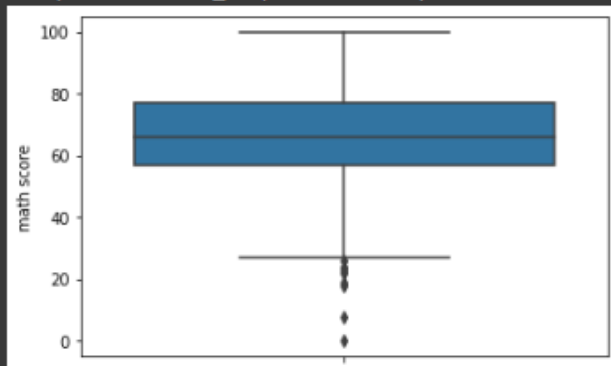
Para el análisis del score de la prueba de matemáticas se usó un gráfico del histograma donde se representan los diferentes resultados en los puntajes obtenidos por los estudiantes



Para los campos Math, Reading y Writing Score se usó una gráfica tipo boxplot para ver la dispersión de los datos atípicos de los resultados de las pruebas

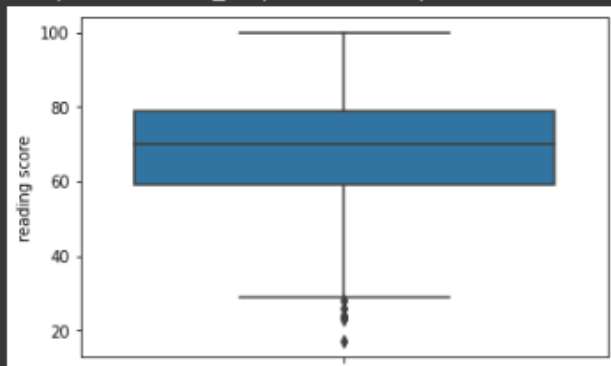
```
sns.boxplot(y='math score',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fed14b1c990>
```



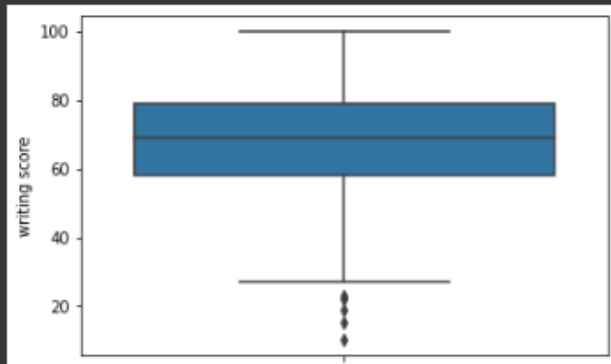
```
[ ] sns.boxplot(y='reading score',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fed14c320d0>
```



```
[ ] sns.boxplot(y='writing score',data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fed14ba4390>
```



Para generar este tipo de grafico en cual representa la cantidad total de estudiantes en el test de preparación, se uso la función pie, y se genero un circulo para convertir el grafico en un anillo

```
gender_agg=df.groupby('gender').count()['test preparation course'].reset_index().copy()
gender_agg
# grafico de torta
plt.pie(gender_agg['test preparation course'],autopct='%.1f%%')

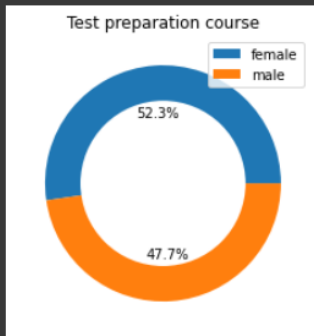
# dibujamos un circulo
centre_circle = plt.Circle((0, 0), 0.70, fc='white')
fig = plt.gcf()

# añadimos un circulo al grafico de tortas para convertirlo en anillo
fig.gca().add_artist(centre_circle)

# titulo
plt.title('Test preparation course')

# labels
plt.legend(labels = gender_agg['gender'], loc="upper right")

plt.show()
```



En el paso de minería de datos primero separamos las variables (X, y) en donde ubicamos los campos a los que queremos hacer la predicción. Luego uso todas las variables para hacer una copia del dataset original y que evitar corrupción de datos en el dataset original

MINERIA DE DATOS

```
x = df[['reading score']]
y = df['math score']

df2 = df[['math score', 'reading score']].copy()
#vamos a analizar si hay alguna correlacion entre el puntaje de matematicas
#con el resultado del puntaje de lectura
df2
```

	math score	reading score
0	72	72
1	69	90
2	90	95
3	47	57
4	76	78
...
995	88	99
996	62	55
997	59	71
998	68	78
999	77	86

1000 rows × 2 columns

Se carga la librería SkLearn con el método LinearRegression para hacer la predicción de los datos que se inicializaron en el punto anterior.

```
[25] from sklearn.linear_model import LinearRegression

[ ] regresion_lineal = LinearRegression()

[ ] regresion_lineal.fit(X,y)

LinearRegression()

[ ] predicciones = regresion_lineal.predict(X)
df2['Predicciones']=predicciones
```

Una vez ejecutada la función LinearRegression () se procede a llamar el dataset copia (df2) para observar la predicción del modelo lineal. En el caso de la primera predicción se obtiene un resultado donde la correlación entre u estudiante que obtenga cierto puntaje en resultado de matemáticas puede o no depender de su resultado en lectura.

```
df2
#se escogio una regresion lineal en este caso
#porque priemro las unicas columnas numericas son las de score
#y ademas, estamso solamente comparando en este caso lectura con matematicas
```

	math score	reading score	Predicciones
0	72	72	68.492803
1	69	90	83.776606
2	90	95	88.022107
3	47	57	55.756300
4	76	78	73.587404
...
995	88	99	91.418508
996	62	55	54.058099
997	59	71	67.643702
998	68	78	73.587404
999	77	86	80.380206

1000 rows × 3 columns

Para la predicción número 2 se hace una correlación entre el puntaje obtenido de escritura y mirar si este puntaje es signo de un buen o mal resultado en la prueba de lectura

Prediccion #2

```
[69] X = df[['reading score']]
      y = df['writing score']

df2 = df[['writing score','reading score']].copy()
#vamos a analizar si hay alguna correlacion entre el puntaje de lectura
#con el resultado del puntaje de escritura
df2
```

	writing score	reading score
0	74.0	72
1	88.0	90
2	93.0	95
3	44.0	57
4	75.0	78
...
995	95.0	99
996	55.0	55
997	65.0	71
998	77.0	78
999	86.0	86

1000 rows x 2 columns

```
[70] regresion_lineal = LinearRegression()

[71] regresion_lineal.fit(X,y)

LinearRegression()

[72] predicciones = regresion_lineal.predict(X)
      df2['Predicciones']=predicciones

[73] df2
```

	writing score	reading score	Predicciones
0	74.0	72	70.891172
1	88.0	90	88.676470
2	93.0	95	93.616830
3	44.0	57	56.070090
4	75.0	78	76.819604
...
995	95.0	99	97.569119
996	55.0	55	54.093946
997	65.0	71	69.903099
998	77.0	78	76.819604
999	86.0	86	84.724181

1000 rows x 3 columns

