Instituto Tecnológico y de Estudios Superiores de Monterrey

Analítica de Datos y
Herramientas de Inteligencia Artificial

Reporte de actividad 2.2

Profesor**:** Alfredo García Suárez

Camila Trujillo Beristain | A01737170
Bernardo Quintana López | A01658064
Fernando Guadarrama González | A01379340
Mauricio Goris García | A01736428

Campus Puebla

6 de abril de 2025

Comenzamos importando las librerías necesarias para hacer el procedimiento. Después cargamos el archivo dado por el socio formador. Identificamos las variables cuantitativas y cualitativas. Igualmente identificamos los valores nulos y atípicos dentro del dataframe. Primeramente, tratamos las cuantitativas con la media de las mismas y las cualitativas con un string “Sin Dato”. En cuanto a los valores atípicos, se trataron con dos métodos, el rango intercuartílico y los percentiles 99 y 1 para las variables que presentaban un mayor registro de datos 0.

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.pyplot as plt
```

✓ 0.0s Python

```python
#Cargamos los datos
data = pd.read_csv('DataAnalytics.csv')
```

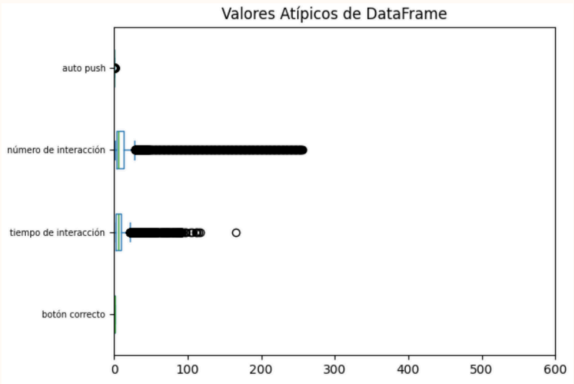
✓ 0.0s Python

```python
numericas = data.select_dtypes(include=['int64', 'float64'])
cualitativas = data.select_dtypes(include=['object'])

numericas_generales = numericas.drop(['tiempo de sesión', 'tiempo de lección'], axis=1)
numericas_generales_sin_nulos = numericas_generales.fillna(numericas_generales.mean())
```

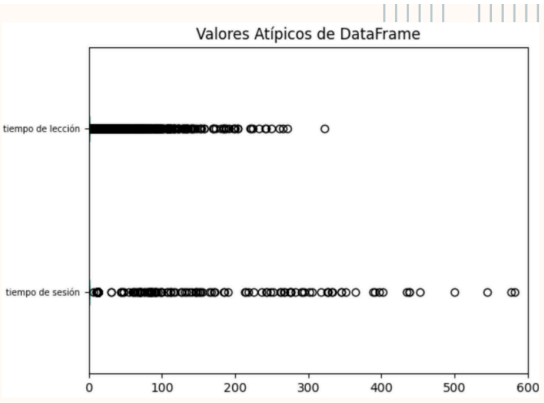
✓ 0.0s Python

```python
y=numericas_generales_sin_nulos

percentile25=y.quantile(0.25) #Q1
percentile75=y.quantile(0.75) #Q3
iqr= percentile75 - percentile25

Limite_Superior_iqr= percentile75 + 1.5*iqr
Limite_Inferior_iqr= percentile25 - 1.5*iqr
print("Limite superior permitido", Limite_Superior_iqr)
print("Limite inferior permitido", Limite_Inferior_iqr)
```

✓ 0.0s Python



Variables: auto push, número de interacción, tiempo de interacción y botón correcto
Método: rango intercuartílico (IQR)



Variables: tiempo de lección y tiempo de sesión (por la gran cantidad de registros “0”)
Método: percentil 1 y 99

Posteriormente, seguimos con la conversión de variables categóricas a numéricas con la ayuda del método de frecuencias, asignando un número según la frecuencia de la variable en el data frame. En este caso ejemplificamos con la variable de “Administrador” el proceso que llevamos a cabo, esto se hizo con todas las demás variables categóricas.

```python
cat1 = data_final.groupby(['Administrador'])['Administrador'].count().sort_values(ascending=False)
cat1
```

✓ 0.0s Python

```
Administrador
ALEIDA            3260
nicolas            440
LEONARDO           371
DENISSE            302
SERGIO ANGEL       243
CARLOS ENRIQUE     228
YAEL DAVID         224
AUSTIN             199
VALENTIN           163
erick              158
IKER BENJAMIN      128
KYTZIA              98
BENJAMIN            51
Name: Administrador, dtype: int64
```

```python
data_final.Administrador = data_final.Administrador.replace({'ALEIDA':'1'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'nicolas':'2'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'LEONARDO':'3'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'DENISSE':'4'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'SERGIO ANGEL':'5'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'CARLOS ENRIQUE':'6'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'YAEL DAVID':'7'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'AUSTIN':'8'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'VALENTIN':'9'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'erick':'10'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'IKER BENJAMIN':'11'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'KYTZIA':'12'}, regex=False)
data_final.Administrador = data_final.Administrador.replace({'BENJAMIN':'13'}, regex=False)
```

✓ 0.0s Python

Ya que teníamos todas las variables categóricas convertidas en numéricas, continuamos con la segmentación por los usuarios solicitados.

```python
usuario1 = data_final[data_final["Usuario"] == "27"]
usuario2 = data_final[data_final["Usuario"] == "30"]
usuario3 = data_final[data_final["Usuario"] == "10"]
usuario4 = data_final[data_final["Usuario"] == "18"]
usuario5 = data_final[data_final["Usuario"] == "11"]
```

✓ 0.0s Python

Pero antes de realizar un heatmap por cada usuario, hicimos uno general para analizar los coeficientes de correlación entre sus variables.

```python
corr_factors = data_final.corr().dropna(how='all', axis=0).dropna(how='all', axis=1)
corr_factors
```

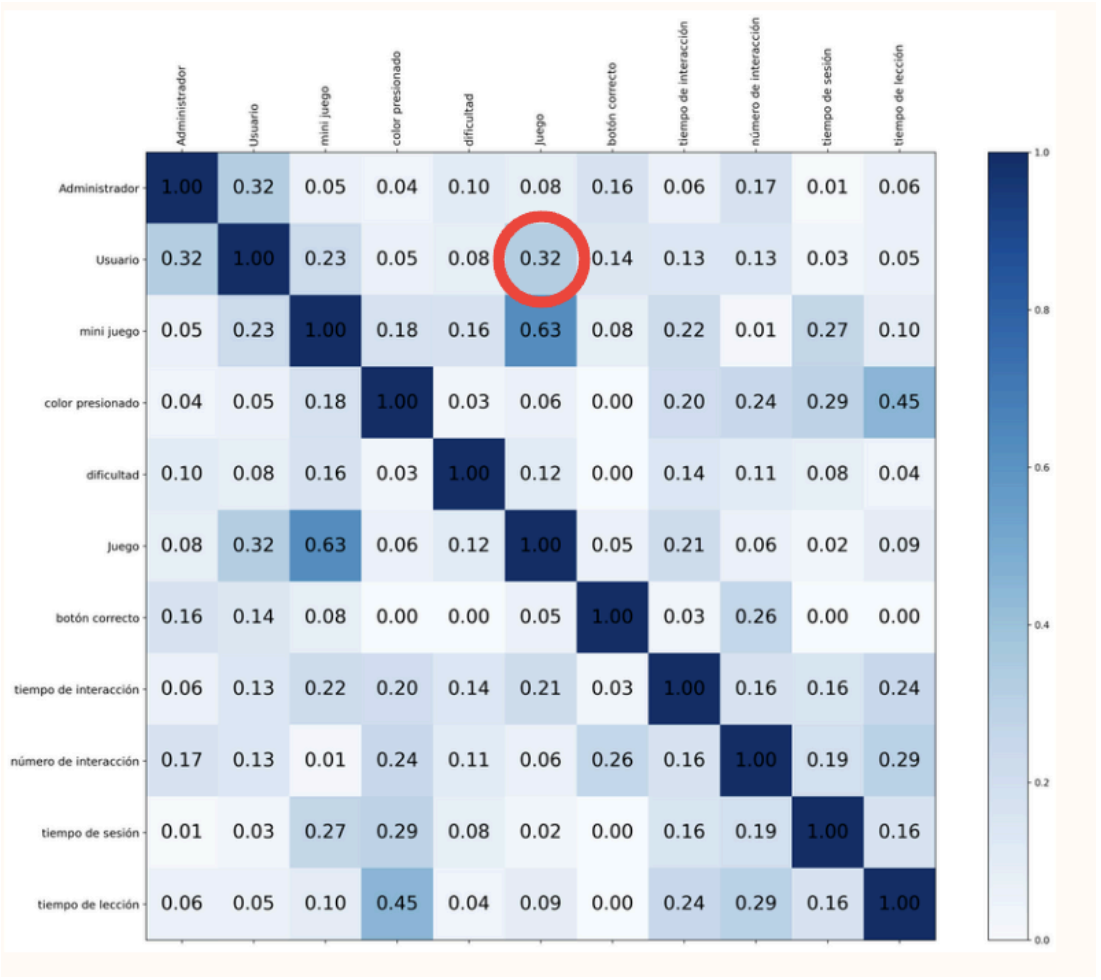
✓ 0.0s Python

	Administrador	Usuario	mini juego	color presionado	dificultad	Juego	botón correcto	tiempo de interacción	número de interacción	tiempo de sesión	tiempo de lección
Administrador	1.000000	0.322260	0.054005	0.041980	-0.099609	0.080074	1.632966e-01	0.056195	-0.167089	-6.272056e-03	5.786454e-02
Usuario	0.322260	1.000000	0.230795	0.054966	-0.078773	0.321093	1.415035e-01	0.131598	-0.126316	2.959428e-02	5.439875e-02
mini juego	0.054005	0.230795	1.000000	0.179507	-0.157252	0.625713	8.406136e-02	0.222250	0.012451	2.726506e-01	9.563100e-02
color presionado	0.041980	0.054966	0.179507	1.000000	0.027161	0.056677	-3.542608e-03	0.202751	0.240833	2.862603e-01	4.490545e-01
dificultad	-0.099609	-0.078773	-0.157252	0.027161	1.000000	-0.115208	2.456645e-03	-0.137693	0.109544	8.336780e-02	-4.185349e-02
Juego	0.080074	0.321093	0.625713	0.056677	-0.115208	1.000000	4.963483e-02	0.213118	-0.060832	2.213791e-02	8.865031e-02
botón correcto	0.163297	0.141503	0.084061	-0.003543	0.002457	0.049635	1.000000e+00	-0.033854	-0.263887	9.987735e-17	-3.852461e-17
tiempo de interacción	0.056195	0.131598	0.222250	0.202751	-0.137693	0.213118	-3.385360e-02	1.000000	0.157743	1.550645e-01	2.432487e-01
número de interacción	-0.167089	-0.126316	0.012451	0.240833	0.109544	-0.060832	-2.638873e-01	0.157743	1.000000	1.880344e-01	2.949683e-01
tiempo de sesión	-0.006272	0.029594	0.272651	0.286260	0.083368	0.022138	9.987735e-17	0.155065	0.188034	1.000000e+00	1.634898e-01
tiempo de lección	0.057865	0.054399	0.095631	0.449055	-0.041853	0.088650	-3.852461e-17	0.243249	0.294968	1.634898e-01	1.000000e+00

```
corr_factors1 = abs(corr_factors)
corr_factors1
```

✓ 0.0s Python

	Administrador	Usuario	mini juego	color presionado	dificultad	Juego	botón correcto	tiempo de interacción	número de interacción	tiempo de sesión	tiempo de lección
Administrador	1.000000	0.322260	0.054005	0.041980	0.099609	0.080074	1.632966e-01	0.056195	0.167089	6.272056e-03	5.786454e-02
Usuario	0.322260	1.000000	0.230795	0.054966	0.078773	0.321093	1.415035e-01	0.131598	0.126316	2.959428e-02	5.439875e-02
mini juego	0.054005	0.230795	1.000000	0.179507	0.157252	0.625713	8.406136e-02	0.222250	0.012451	2.726506e-01	9.563100e-02
color presionado	0.041980	0.054966	0.179507	1.000000	0.027161	0.056677	3.542608e-03	0.202751	0.240833	2.862603e-01	4.490545e-01
dificultad	0.099609	0.078773	0.157252	0.027161	1.000000	0.115208	2.456645e-03	0.137693	0.109544	8.336780e-02	4.185349e-02
Juego	0.080074	0.321093	0.625713	0.056677	0.115208	1.000000	4.963483e-02	0.213118	0.060832	2.213791e-02	8.865031e-02
botón correcto	0.163297	0.141503	0.084061	0.003543	0.002457	0.049635	1.000000e+00	0.033854	0.263887	9.987735e-17	3.852461e-17
tiempo de interacción	0.056195	0.131598	0.222250	0.202751	0.137693	0.213118	3.385360e-02	1.000000	0.157743	1.550645e-01	2.432487e-01
número de interacción	0.167089	0.126316	0.012451	0.240833	0.109544	0.060832	2.638873e-01	0.157743	1.000000	1.880344e-01	2.949683e-01
tiempo de sesión	0.006272	0.029594	0.272651	0.286260	0.083368	0.022138	9.987735e-17	0.155065	0.188034	1.000000e+00	1.634898e-01
tiempo de lección	0.057865	0.054399	0.095631	0.449055	0.041853	0.088650	3.852461e-17	0.243249	0.294968	1.634898e-01	1.000000e+00



Encontramos que la variable juego tenía uno de los coeficientes más altos, así que la utilizamos para un modelo de regresión lineal múltiple y así saber si se puede mejorar ese coeficiente.

```
Vars_Indep= data_final[['Administrador', 'mini juego','color presionado', 'color presionado', 'dificultad', 'Juego', 'botón correcto', 'número de interacción']]
Var_Dep= data_final['Usuario']
```
✓ 0.0s

+ Code + Markdown

```
from sklearn.linear_model import LinearRegression
model= LinearRegression()
```
✓ 0.0s

```
type(model)
```
✓ 0.0s

```
sklearn.linear_model._base.LinearRegression
```

```
model.fit(X=Vars_Indep, y=Var_Dep)
```
✓ 0.0s

```
LinearRegression
LinearRegression()
```

```
coef_Deter=model.score(X=Vars_Indep, y=Var_Dep)
coef_Deter
```
✓ 0.0s

```
0.20249809384993434
```

```
coef_Correl=np.sqrt(coef_Deter)
coef_Correl
```
✓ 0.0s

```
0.44999788205049845
```

Gracias a un modelo de regresión lineal múltiple, logramos mejorar el coeficiente de correlación.

Ahora sí, proseguimos con el análisis individual de los 5 usuarios solicitados por el profesor, generando un mapa de calor con cada uno de ellos, después identificando las variables que tienen el mayor coeficiente de correlación y realizando un modelo de regresión lineal múltiple para mejorar los coeficientes encontrados. El proceso se ejemplifica primero con el usuario número 1 (Benjamín), y este fue el código que aplicamos con todos los usuarios.

```
corr_factorsu1 = usuario1.corr().dropna(how='all', axis=0).dropna(how='all', axis=1)
corr_factorsu1
```
✓ 0.0s Python

	mini juego	color presionado	dificultad	botón correcto	tiempo de interacción	número de interacción	tiempo de sesión	tiempo de lección
mini juego	1.000000	0.264407	0.084592	0.016692	0.096144	0.208386	0.871506	0.027196
color presionado	0.264407	1.000000	-0.332598	-0.142168	0.302118	0.490704	0.344148	0.506580
dificultad	0.084592	-0.332598	1.000000	0.041072	0.036646	0.116633	0.304446	0.013318
botón correcto	0.016692	-0.142168	0.041072	1.000000	-0.071754	0.159280	0.035281	0.051934
tiempo de interacción	0.096144	0.302118	0.036646	-0.071754	1.000000	0.330938	0.081117	0.119402
número de interacción	0.208386	0.490704	0.116633	0.159280	0.330938	1.000000	0.348090	0.512384
tiempo de sesión	0.871506	0.344148	0.304446	0.035281	0.081117	0.348090	1.000000	0.085109
tiempo de lección	0.027196	0.506580	0.013318	0.051934	0.119402	0.512384	0.085109	1.000000

```python
fig, ax = plt.subplots(figsize=(15, 10))
cax = ax.matshow(corr_factorsu1, cmap="Reds")
fig.colorbar(cax)

# Añadir anotaciones manualmente
for i in range(corr_factorsu1.shape[0]):
    for j in range(corr_factorsu1.shape[1]):
        ax.text(j, i, f"{corr_factorsu1.iloc[i, j]:.2f}",
                ha="center", va="center", fontsize=20)

plt.xticks(range(len(corr_factorsu1.columns)), corr_factorsu1.columns, rotation=90, fontsize=12)
plt.yticks(range(len(corr_factorsu1.index)), corr_factorsu1.index, fontsize=12)
plt.savefig('Usuario1.png', dpi=300, bbox_inches='tight')
plt.show()
```

6] ✓ 0.7s Python

```python
epU1= usuario1[['mini juego', 'color presionado', 'dificultad', 'botón correcto', 'tiempo de interacción', 'número de interacción', 'tiempo de lección']]
l= usuario1['tiempo de sesión']
```

537] ✓ 0.0s Python

```python
modelU1 = LinearRegression()
modelU1.fit(X=Vars_IndepU1, y=Var_DepU1)
```

538] ✓ 0.0s Python

```
LinearRegression
LinearRegression()
```

```python
y_predU1= modelU1.predict(X=usuario1[['mini juego', 'color presionado', 'dificultad', 'botón correcto', 'tiempo de interacción', 'número de interacción', 't
y_predU1
```

539] ✓ 0.0s Python

Outputs are collapsed ···

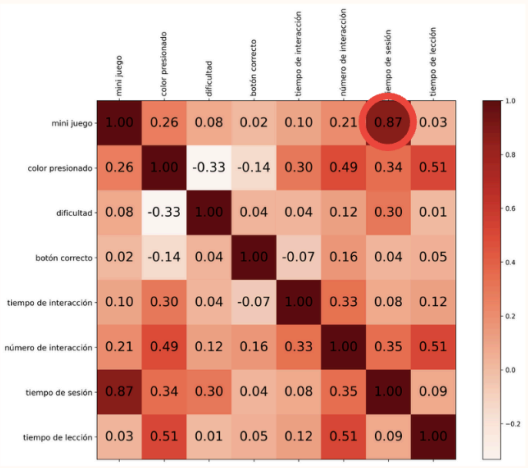
```python
coef_DeterU1=modelU1.score(X=Vars_IndepU1, y=Var_DepU1)
coef_CorrelU1=np.sqrt(coef_DeterU1)
coef_CorrelU1
```

540] ✓ 0.0s Python

```
0.9400219525094239
```

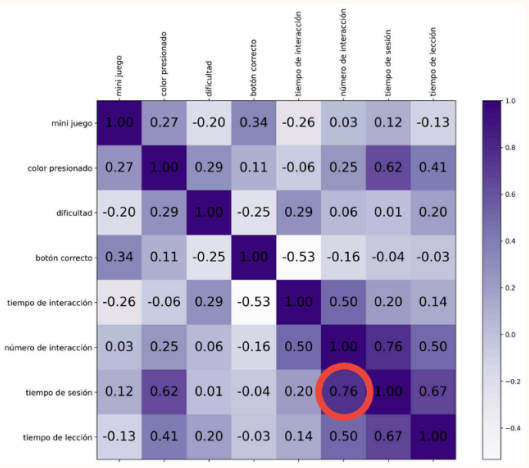
Benjamín

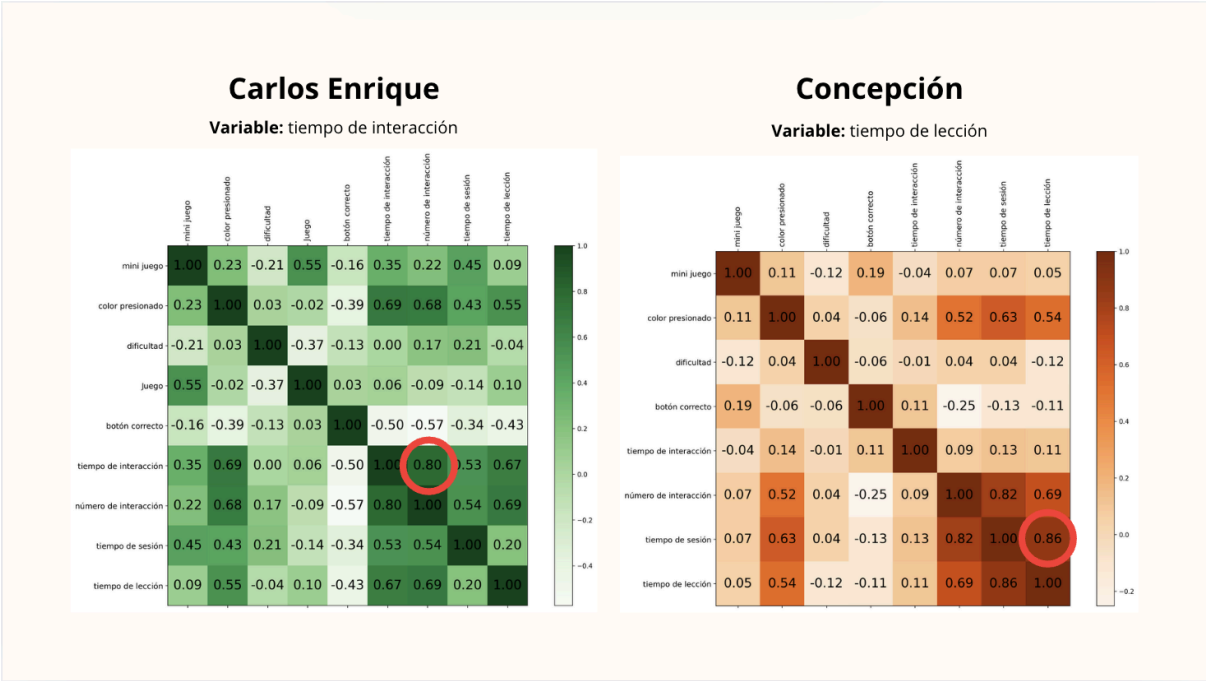
Variable: tiempo de sesión

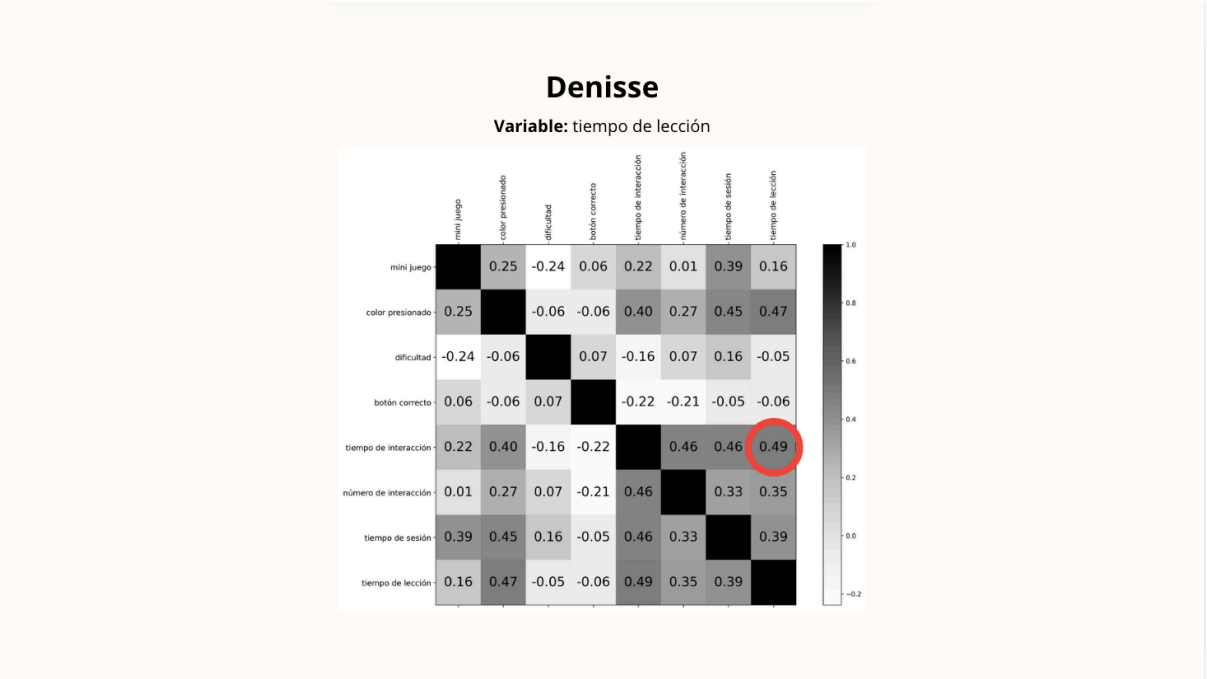


Carlos Abel

Variable: tiempo de sesión







Después de que obtuvimos los heatmaps, los coeficientes más altos y definimos los modelos de regresión lineal múltiple encontramos los siguientes resultados:

Usuario	Variable seleccionada (dependiente)	Coeficiente correlación linear	Coeficiente correlación linear múltiple
Benjamín	tiempo de sesión	**0.87**	**0.94**
Carlos Abel	tiempo de sesión	**0.76**	**0.92**
Carlos Enrique	tiempo de interacción	**0.80**	**0.85**
Concepción	tiempo de lección	**0.86**	**0.88**
Denisse	tiempo de lección	**0.49**	**0.59**

En general, en los 5 escenarios se encontró que los modelos de regresión lineal múltiple mejoraron los coeficientes de correlación, aumentando los 5 analizados.

En conclusión, los coeficientes de correlación lineal y múltiple varían notablemente entre usuarios, destacando que variables como "tiempo de sesión" y "tiempo de interacción" tienen impactos diferenciados en cada caso. En todos los usuarios analizados, el coeficiente de correlación lineal múltiple supera al lineal simple, lo que sugiere que incluir más variables en el modelo mejora la precisión de las predicciones.