Instituto Tecnológico y de Estudios Superiores de Monterrey

Analítica de Datos y
Herramientas de Inteligencia Artificial

Reporte de actividad 3.2

Profesor**:** Alfredo García Suárez

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

Campus Puebla

13 de abril de 2025

Reporte Explicativo del Análisis de Datos

El reporte documenta el análisis de un conjunto de datos relacionados con la interacción de usuarios en una plataforma de aprendizaje, en este caso WUUPI. El objetivo es detectar patrones de comportamiento, identificar datos faltantes o atípicos, transformar variables para facilitar su análisis y aplicar modelos de ajuste que permitan predecir tendencias individuales.

Comenzamos nuestro análisis importando las bibliotecas necesarias. En la **Celda 3** incluimos pandas y numpy para la manipulación de datos, matplotlib.pyplot y seaborn para la visualización, scipy.optimize.curve_fit para el modelado no lineal, y r2_score de sklearn para medir el ajuste de los modelos. Estas herramientas nos permitieron estructurar el trabajo de manera técnica y analítica desde el inicio.

Estas herramientas nos permiten trabajar con estructuras de datos, generar gráficas, ajustar modelos no lineales y evaluar la precisión de predicciones.

Cargamos el archivo DataAnalytics.csv utilizando pd.read_csv(). Este archivo contiene registros de múltiples variables relacionadas con el comportamiento de los usuarios en una plataforma educativa (WUUPI), incluyendo datos como el número de interacción, el tiempo invertido en cada fase, y la precisión en tareas ejecutadas. Almacenamos esta información en la variable data, que usamos como base para todo el análisis.

Evaluamos la calidad de los datos identificando los valores nulos con .isnull().sum(). Detectamos que varias columnas, particularmente las relacionadas con el tiempo, contenían datos faltantes. Despues, dividimos el conjunto en variables numéricas (numericas) y cualitativas (cualitativas), y realizamos imputaciones de los valores faltantes usando la media de cada columna con .fillna(). Esta técnica nos permitió conservar los registros sin distorsionar en exceso las distribuciones originales.

```python
valores_nulos = data.isnull().sum()
print(valores_nulos)
```

Python

```
Administrador             0
Usuario                   0
botón correcto          762
tiempo de interacción   762
mini juego              156
número de interacción   762
color presionado        762
dificultad                0
fecha                     0
Juego                     0
auto push               762
tiempo de lección       177
tiempo de sesión        606
```

```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())
```

Python

En la siguiente etapa, aislamos las variables tiempo de sesión y tiempo de lección, que son fundamentales para comprender la duración e intensidad de la participación del usuario. Les imputamos sus valores faltantes también con la media. Al tratar estas variables por separado, pudimos analizar patrones de comportamiento más detalladamente, ya que reflejan distintos momentos de interacción con el sistema.

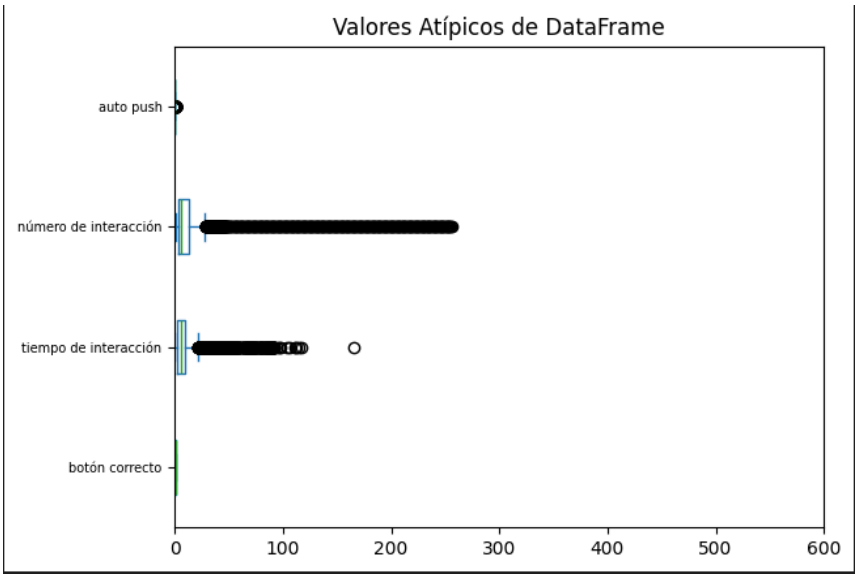
```python
tiempo_sesion = numericas['tiempo de sesión']
tiempo_leccion = numericas['tiempo de lección']

media_sesion_sin_ceros = tiempo_sesion.mean()
media_leccion_sin_ceros = tiempo_leccion.mean()

tiempo_sesion_sin_nulos = tiempo_sesion.fillna(media_sesion_sin_ceros)
tiempo_leccion_sin_nulos = tiempo_leccion.fillna(media_leccion_sin_ceros)
```

Python

Generamos boxplots para observar la dispersión de las variables y detectar valores atípicos. Esto lo hicimos con .plot(kind='box'), ajustando el eje horizontal para facilitar la interpretación. Estos diagramas nos ayudaron a identificar usuarios con tiempos de sesión o interacción extremadamente altos, que podrían estar sesgando los resultados. A partir de estas observaciones, consideramos el posible tratamiento posterior de esos casos extremos.

```python
fig = plt.figure(figsize = (15,10))
numericas_generales_sin_nulos.plot(kind='box', vert=False)
plt.xlim([0, 600])
plt.title('Valores Atípicos de DataFrame')
plt.yticks(fontsize=7, rotation=0)
plt.show()
```

Python



Estas gráficas mostraron la presencia de valores atípicos en variables como tiempo de interacción, lo cual permitió tomar decisiones más informadas sobre el tratamiento de datos extremos.

Creamos un nuevo DataFrame (numericas_con0) donde imputamos los valores faltantes con ceros en lugar de medias. Esta comparación visual nos permitió explorar cómo diferentes métodos de imputación afectan la distribución de los datos. Observar esta diferencia fue útil para evaluar qué técnica era más adecuada según el tipo de variable, dado que imputar con ceros puede subestimar el tiempo real invertido por los usuarios.

```python
numericas_con0 = pd.concat([tiempo_sesion_sin_nulos, tiempo_leccion_sin_nulos], axis=1)
```

Python

Realizamos la conversión de los valores de texto en las columnas Usuario y Administrador a valores numéricos mediante el método .replace(). Esta transformación fue necesaria porque

muchas funciones de análisis requieren trabajar con datos numéricos. Al asignar números a cada nombre, preparamos los datos para filtrarlos y modelarlos de forma eficiente.

```
data_final.Usuario = data_final.Usuario.replace({'LEONARDO':'1'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'ALEIDA':'2'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'nicolas':'3'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'JOSE JAVIER ':'4'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'LEONARDO ':'5'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'JESUS ALEJANDRO':'6'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'ramiro isai':'7'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'ADRIAN ':'8'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'SERGIO ANGEL':'9'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'CARLOS ENRIQUE':'10'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'DENISSE':'11'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'YAEL DAVID':'12'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'VALENTIN':'13'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'erick ':'14'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'IKER BENJAMIN':'15'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'NICOLAS |':'16'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'ERICK OSVALDO':'17'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'CONCEPCION':'18'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'KYTZIA':'19'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'AUSTIN':'20'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'JOSE IGNACIO TADEO':'21'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'JOSE IAN':'22'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'ASHLEY ':'23'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'JOSHUA':'24'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'YEREMI YAZMIN ':'25'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'MA DEL ROSARIO ':'26'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'BENJAMIN':'27'}, regex=False)
data_final.Usuario = data_final.Usuario.replace({'INGRID':'28'}, regex=False)
```

Calculamos matrices de correlación para identificar relaciones entre variables como botón correcto, color presionado, tiempo de interacción, y dificultad. Utilizamos matshow() de Matplotlib para representar estos valores visualmente. Gracias a estas visualizaciones, detectamos relaciones fuertes entre variables específicas que más adelante usamos para construir modelos predictivos.

```
fig, ax = plt.subplots(figsize=(20, 15))
cax = ax.matshow(corr_factors1, cmap="Blues")
fig.colorbar(cax)

# Añadir anotaciones manualmente
for i in range(corr_factors1.shape[0]):
    for j in range(corr_factors1.shape[1]):
        ax.text(j, i, f"{corr_factors1.iloc[i, j]:.2f}",
                ha="center", va="center", fontsize=20)

plt.xticks(range(len(corr_factors1.columns)), corr_factors1.columns, rotation=90, fontsize=12)
plt.yticks(range(len(corr_factors1.index)), corr_factors1.index, fontsize=12)
plt.savefig('General.png', dpi=300, bbox_inches='tight')
plt.show()
```

Python

Este análisis se repitió para distintos subconjuntos de datos por usuario, visualizando la relación entre variables como:

- Tiempo de interacción
- Color presionado
- Botón correcto
- Dificultad

Filtramos subconjuntos por usuario y modelamos sus datos de forma independiente. En cada caso, establecimos una variable dependiente (por ejemplo, tiempo de interacción) y una independiente (por ejemplo, número de interacción). Utilizamos la función curve_fit() para ajustar modelos cuadráticos o exponenciales que explicaran el comportamiento observado.

```python
Vars_Indep= data_final[['mini juego']]
Var_Dep= data_final['Juego']
```

Python

```python
x = Vars_Indep
y = Var_Dep
```

Python

```python
def func1(x, a, b, c):
    return a*np.exp(-b*x) + c
```

Python

```python
parametros, covs= curve_fit(func1, data_final['mini juego'], data_final['Juego'])
parametros
```

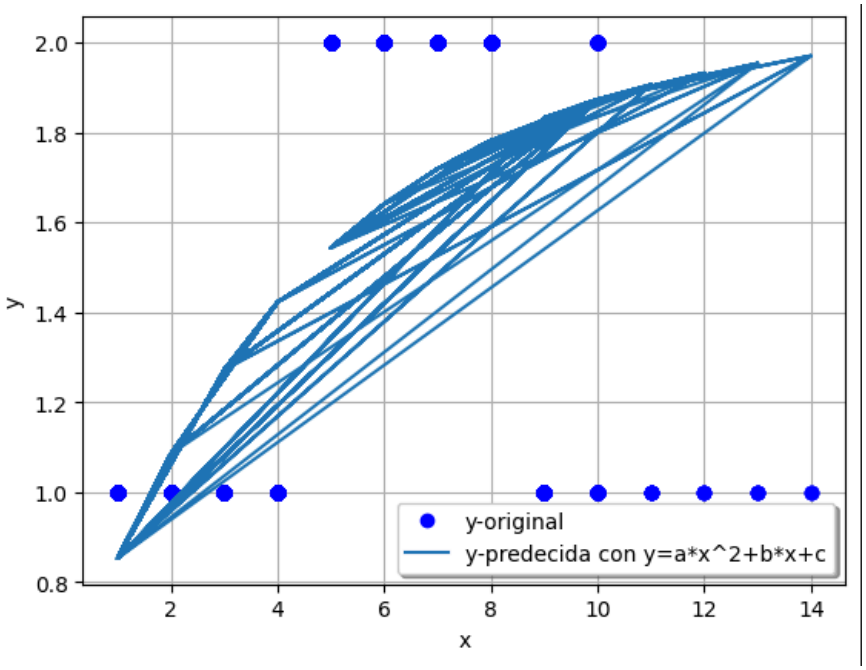
Python

Estos modelos nos permitieron explorar cómo evoluciona el desempeño de los usuarios a medida que interactúan más con la plataforma. Observamos patrones crecientes, decrecientes o estables dependiendo del tipo de tarea y del usuario.

Para cada modelo ajustado, generamos gráficos con los datos reales (puntos azules) y la predicción del modelo (línea continua). Esto lo hicimos usando plt.plot(). Estas gráficas nos dieron una herramienta visual potente para evaluar si el modelo se adaptaba bien a los datos. También pudimos comparar el comportamiento de distintos usuarios entre sí.

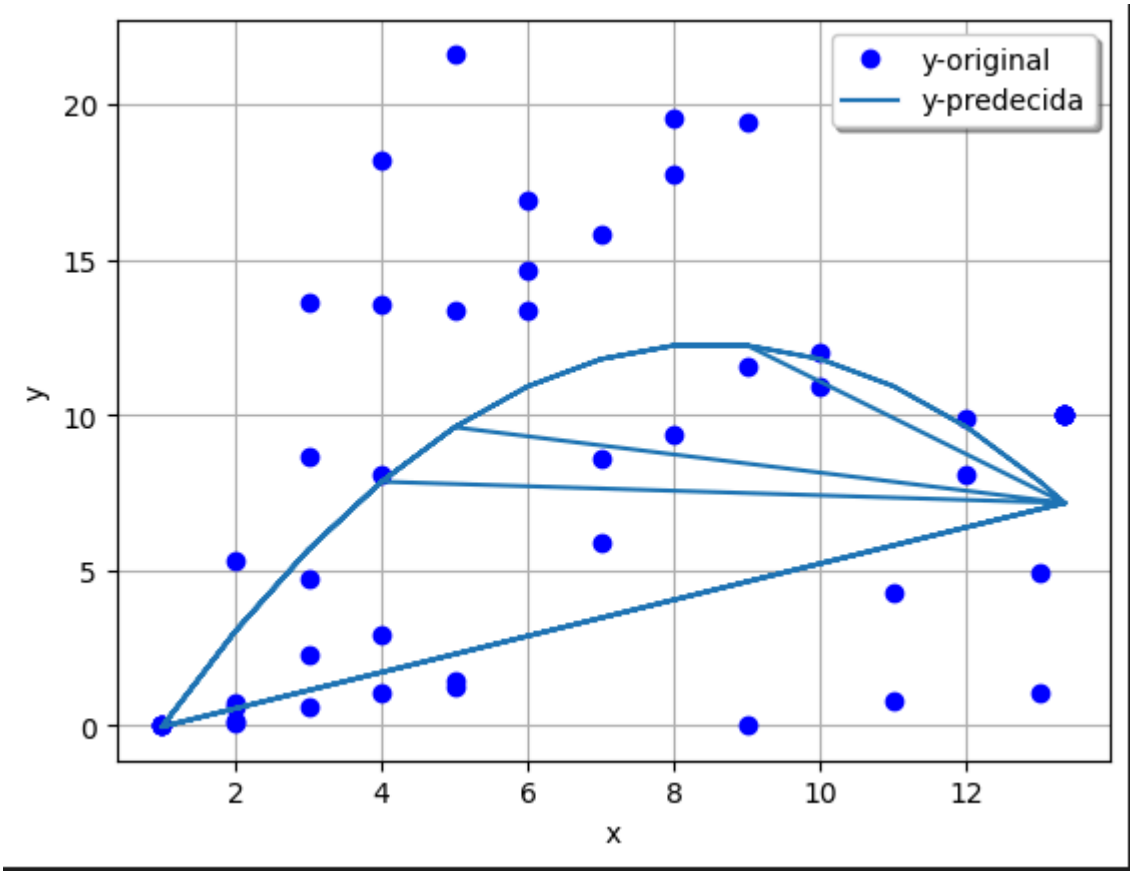
```python
plt.plot(x, y, 'bo', label="y-original")
plt.plot(x, y_pred, label="y-predecida con y=a*x^2+b*x+c")
plt.xlabel('x')
plt.ylabel('y')
plt.legend(loc='best', fancybox=True, shadow=True)
plt.grid(True)
plt.show()
```

Python



Se aplicó este procedimiento a combinaciones como:

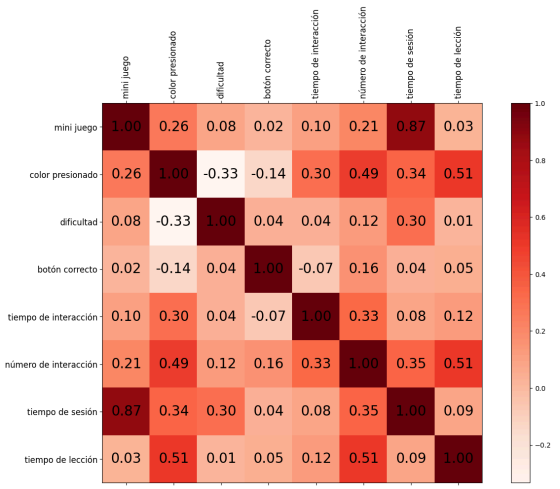
- Color presionado → botón correcto
- Tiempo de sesión → dificultad
- Número de interacción → tiempo de interacción

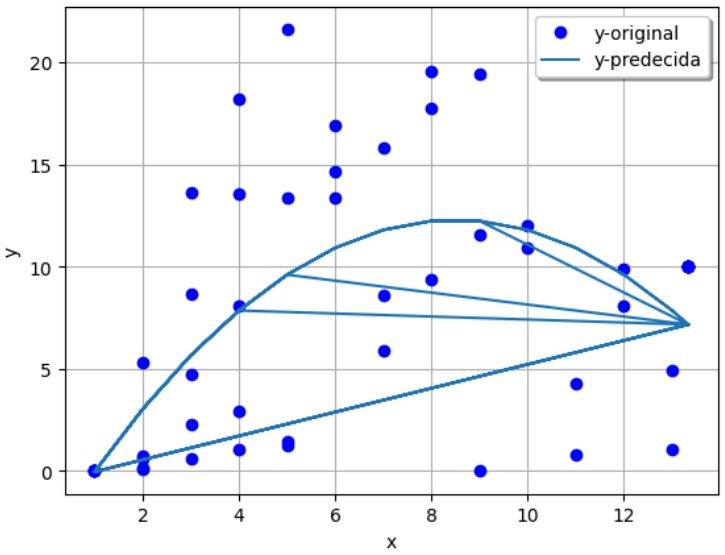


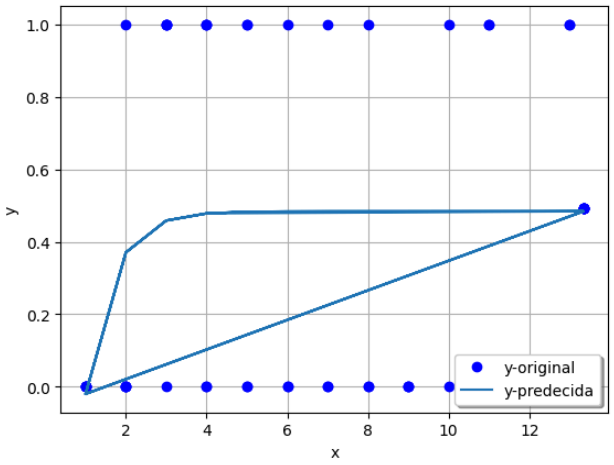
Cada uno de los modelos anteriores fue visualizado con gráficos de dispersión ('bo') y líneas de predicción. Estas gráficas permiten observar si la tendencia del usuario era creciente, decreciente o estable en variables como precisión, interacción o desempeño.

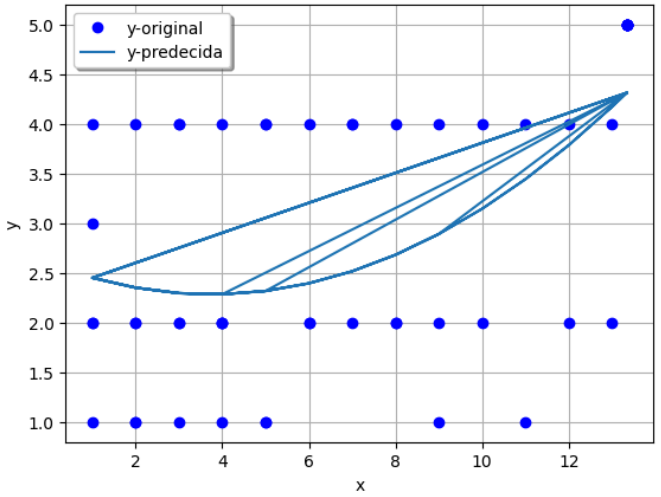
Finalmente tuvimos los resultados por usuario en orden de relacion (heatmap, tiempo de interaccion, boton correcto, color presionado)

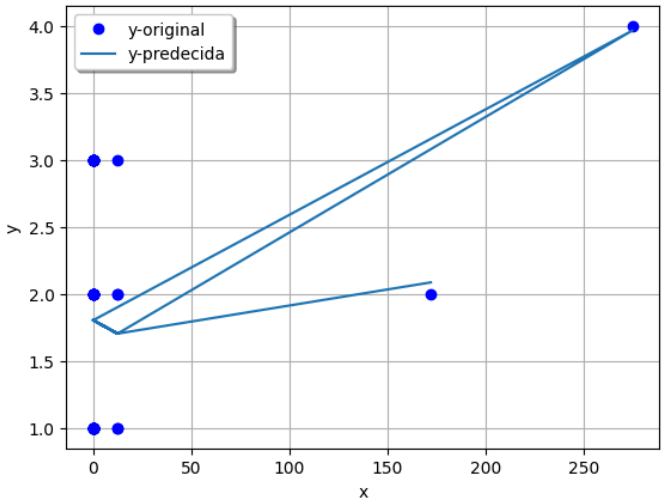
- **Benjamin**


mini juego
color presionado
dificultad
botón correcto
tiempo de interacción
número de interacción
tiempo de sesión
tiempo de lección
mini juego 1.00 0.26 0.08 0.02 0.10 0.21 0.87 0.03
color presionado 0.26 1.00 -0.33 -0.14 0.30 0.49 0.34 0.51
dificultad 0.08 -0.33 1.00 0.04 0.04 0.12 0.30 0.01
botón correcto 0.02 -0.14 0.04 1.00 -0.07 0.16 0.04 0.05
tiempo de interacción 0.10 0.30 0.04 -0.07 1.00 0.33 0.08 0.12
número de interacción 0.21 0.49 0.12 0.16 0.33 1.00 0.35 0.51
tiempo de sesión 0.87 0.34 0.30 0.04 0.08 0.35 1.00 0.09
tiempo de lección 0.03 0.51 0.01 0.05 0.12 0.51 0.09 1.00

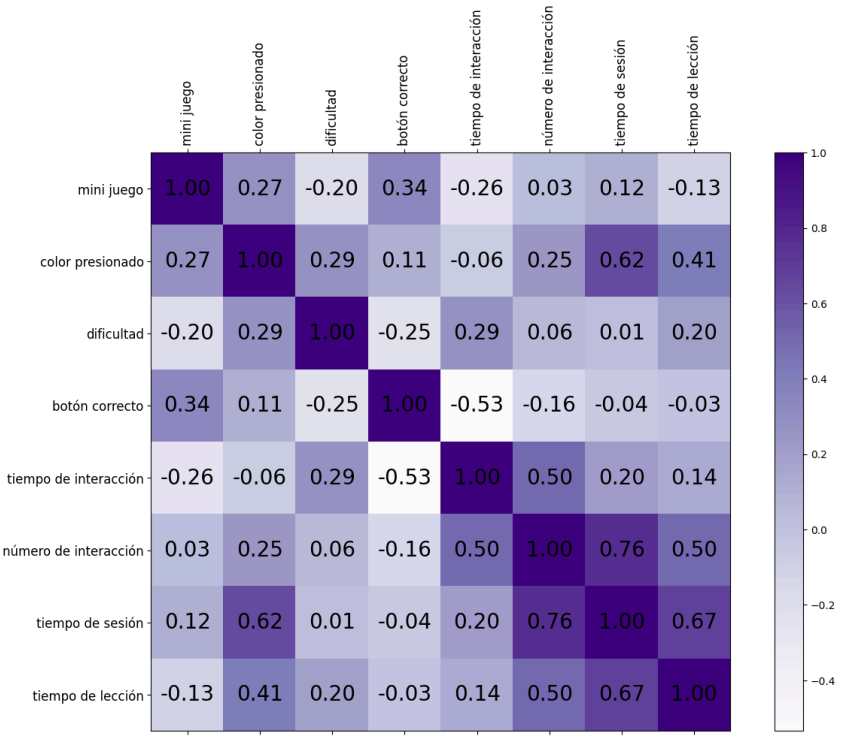

y-original
y-predecida
x
y

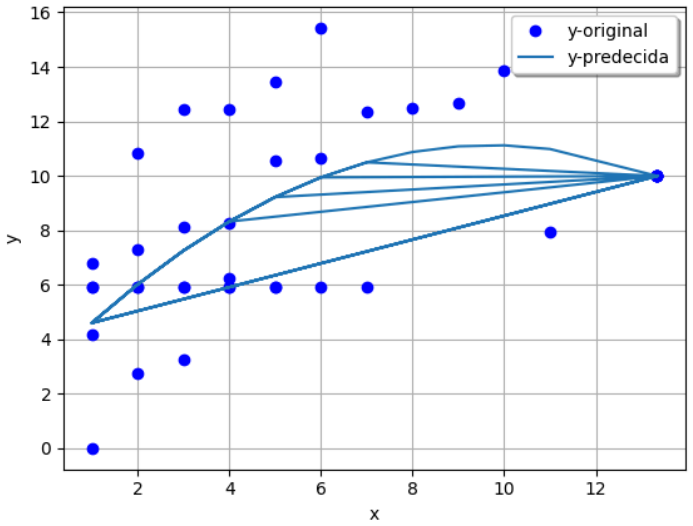

y-original
y-predecida
x
y

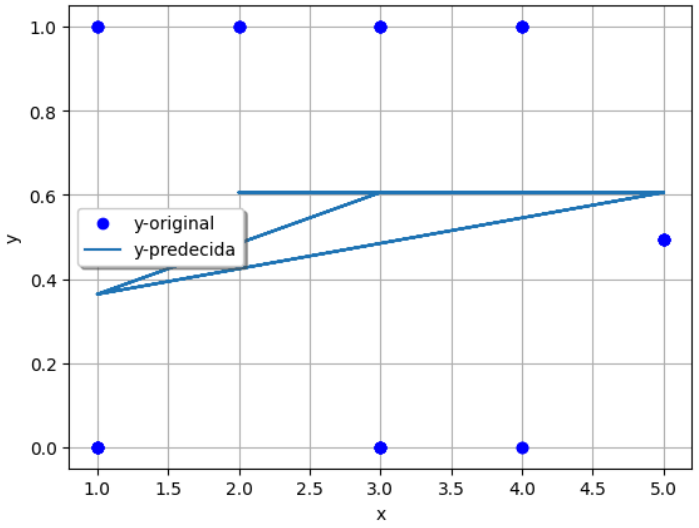

y-original
y-predecida
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
y
2
4
6
8
10
12
x

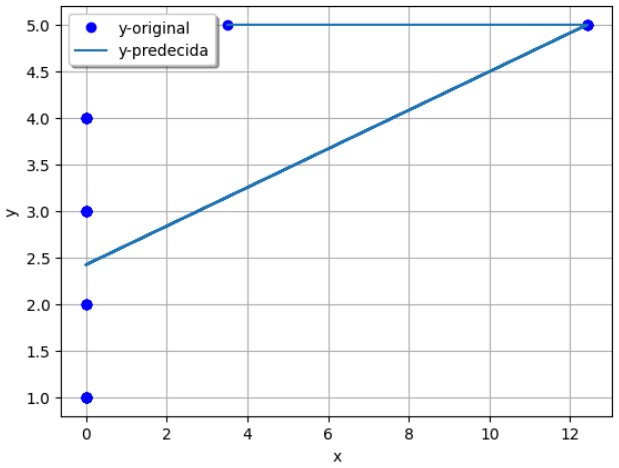

y-original
y-predecida
4.0
3.5
3.0
2.5
2.0
1.5
1.0
y
0
50
100
150
200
250
x

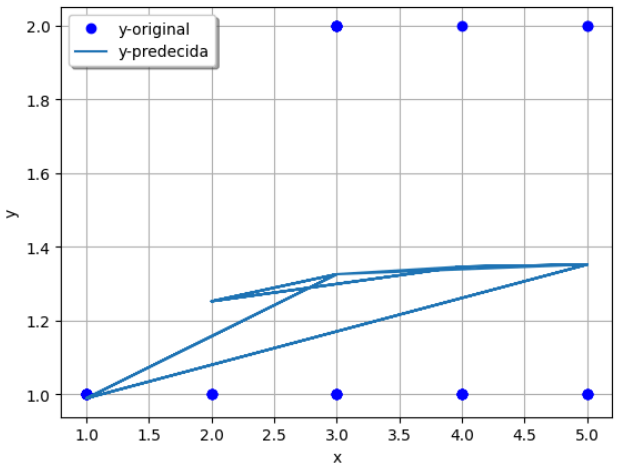
- **Carlos Abel**


mini juego
color presionado
dificultad
botón correcto
tiempo de interacción
número de interacción
tiempo de sesión
tiempo de lección
mini juego 1.00 0.27 -0.20 0.34 -0.26 0.03 0.12 -0.13
color presionado 0.27 1.00 0.29 0.11 -0.06 0.25 0.62 0.41
dificultad -0.20 0.29 1.00 -0.25 0.29 0.06 0.01 0.20
botón correcto 0.34 0.11 -0.25 1.00 -0.53 -0.16 -0.04 -0.03
tiempo de interacción -0.26 -0.06 0.29 -0.53 1.00 0.50 0.20 0.14
número de interacción 0.03 0.25 0.06 -0.16 0.50 1.00 0.76 0.50
tiempo de sesión 0.12 0.62 0.01 -0.04 0.20 0.76 1.00 0.67
tiempo de lección -0.13 0.41 0.20 -0.03 0.14 0.50 0.67 1.00
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

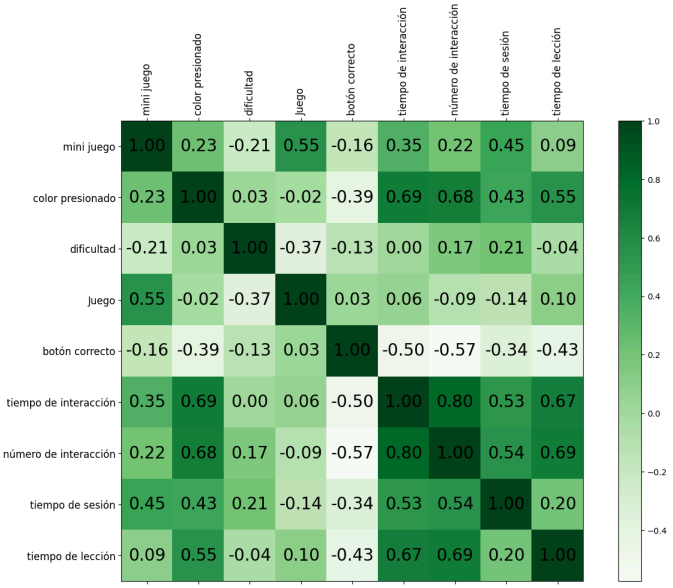

y-original
y-predecida
y
x
0
2
4
6
8
10
12
14
16

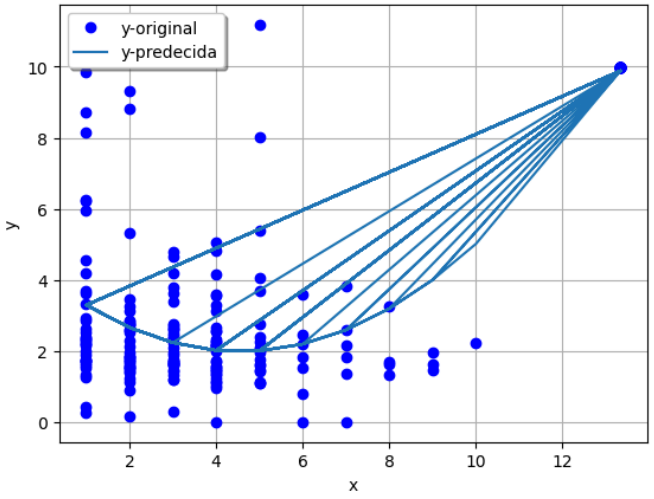


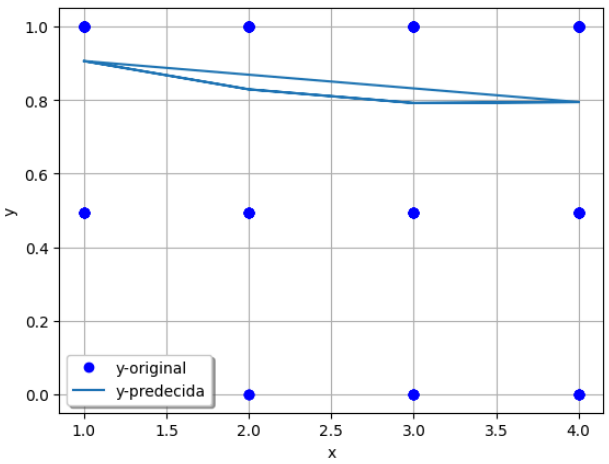


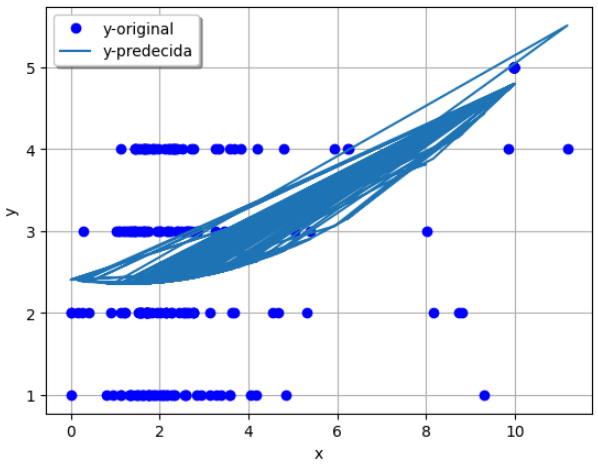


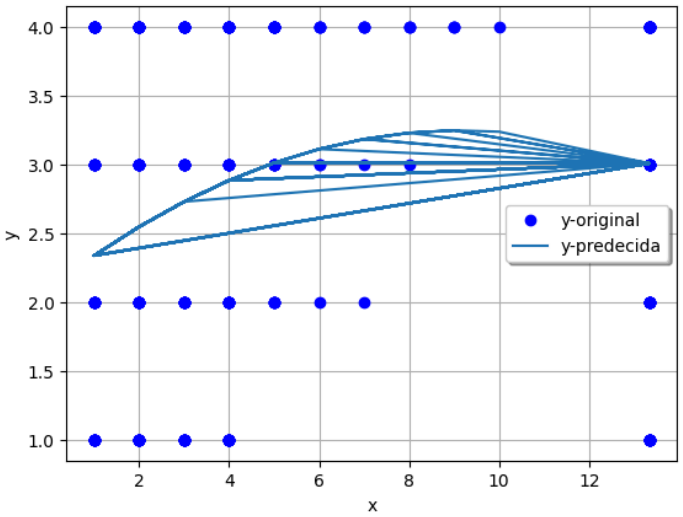
- **Carlos Enrique**


mini juego
color presionado
dificultad
Juego
botón correcto
tiempo de interacción
número de interacción
tiempo de sesión
tiempo de lección
mini juego 1.00 0.23 -0.21 0.55 -0.16 0.35 0.22 0.45 0.09
color presionado 0.23 1.00 0.03 -0.02 -0.39 0.69 0.68 0.43 0.55
dificultad -0.21 0.03 1.00 -0.37 -0.13 0.00 0.17 0.21 -0.04
Juego 0.55 -0.02 -0.37 1.00 0.03 0.06 -0.09 -0.14 0.10
botón correcto -0.16 -0.39 -0.13 0.03 1.00 -0.50 -0.57 -0.34 -0.43
tiempo de interacción 0.35 0.69 0.00 0.06 -0.50 1.00 0.80 0.53 0.67
número de interacción 0.22 0.68 0.17 -0.09 -0.57 0.80 1.00 0.54 0.69
tiempo de sesión 0.45 0.43 0.21 -0.14 -0.34 0.53 0.54 1.00 0.20
tiempo de lección 0.09 0.55 -0.04 0.10 -0.43 0.67 0.69 0.20 1.00
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

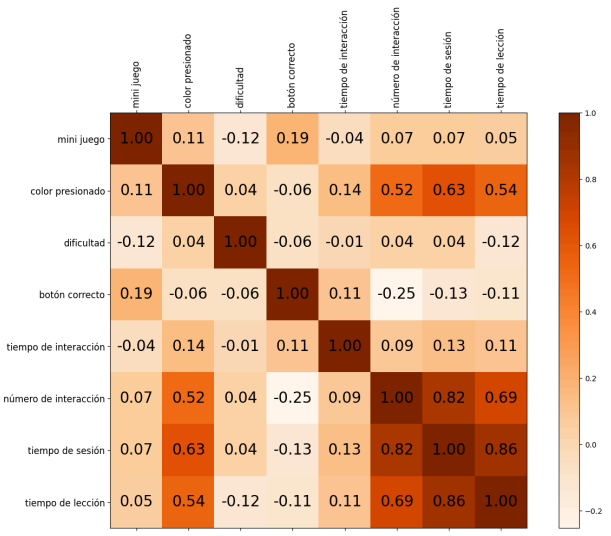

y-original
y-predecida
y
x
0
2
4
6
8
10
12

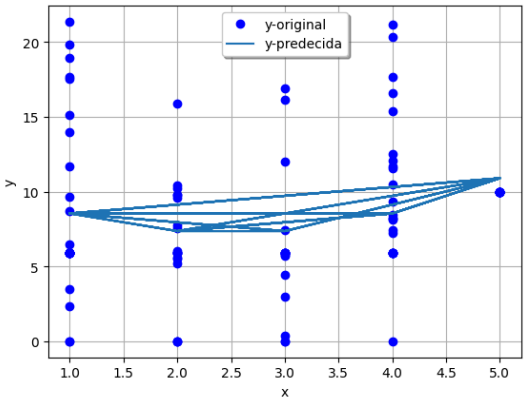

y-original
y-predecida
y
x
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0

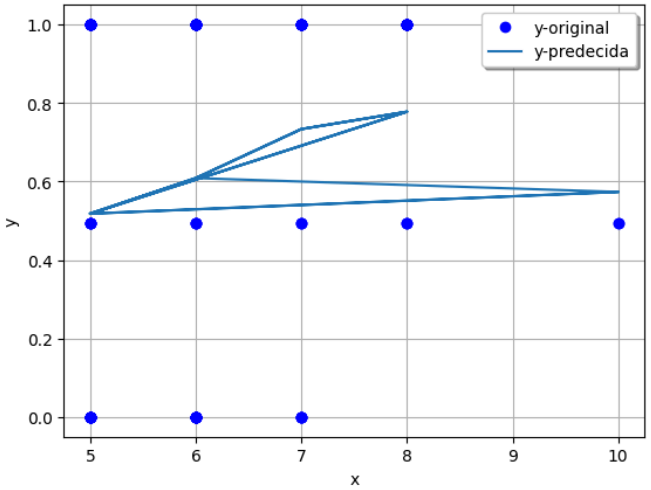

y-original
y-predecida
x
y

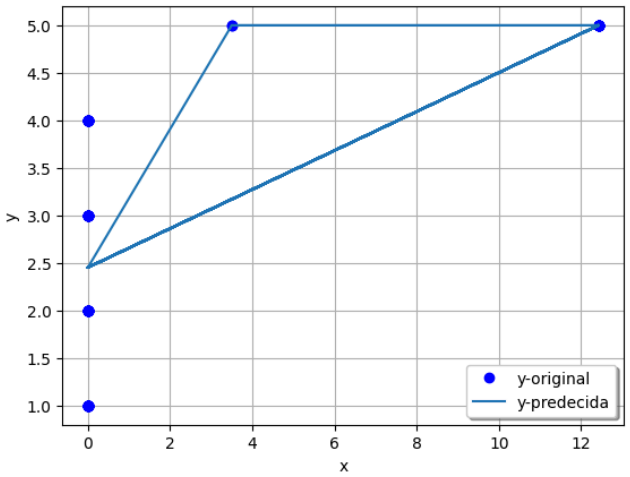

y-original
y-predecida
x
y

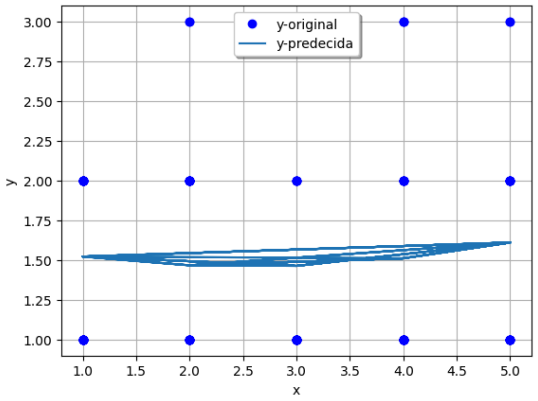
- **Concepción**


mini juego
color presionado
dificultad
botón correcto
tiempo de interacción
número de interacción
tiempo de sesión
tiempo de lección
mini juego 1.00 0.11 -0.12 0.19 -0.04 0.07 0.07 0.05
color presionado 0.11 1.00 0.04 -0.06 0.14 0.52 0.63 0.54
dificultad -0.12 0.04 1.00 -0.06 -0.01 0.04 0.04 -0.12
botón correcto 0.19 -0.06 -0.06 1.00 0.11 -0.25 -0.13 -0.11
tiempo de interacción -0.04 0.14 -0.01 0.11 1.00 0.09 0.13 0.11
número de interacción 0.07 0.52 0.04 -0.25 0.09 1.00 0.82 0.69
tiempo de sesión 0.07 0.63 0.04 -0.13 0.13 0.82 1.00 0.86
tiempo de lección 0.05 0.54 -0.12 -0.11 0.11 0.69 0.86 1.00

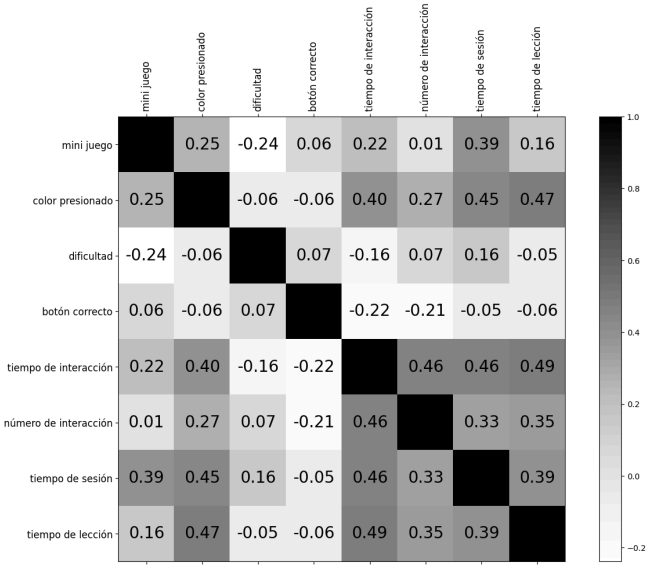

y-original
y-predecida
x
y

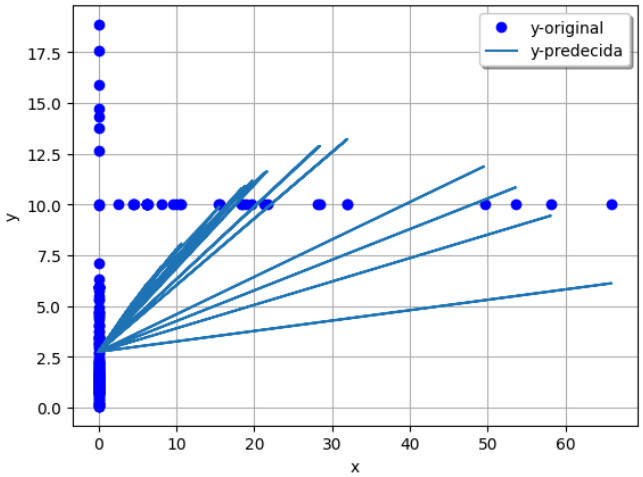

y-original
y-predecida
x
y

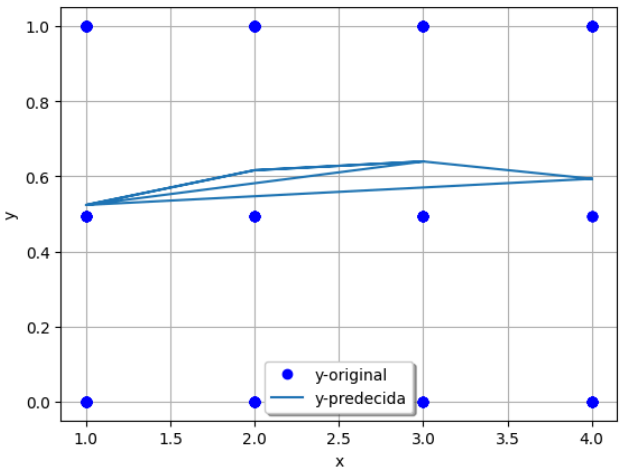

y-original
y-predecida
x
y

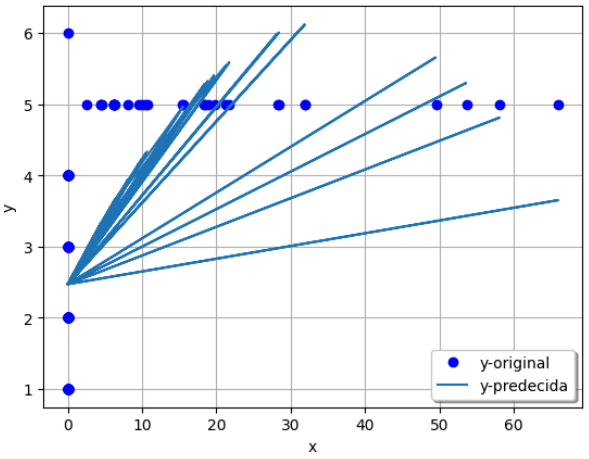

y-original
y-predecida
3.00
2.75
2.50
2.25
2.00
1.75
1.50
1.25
1.00
y
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
x

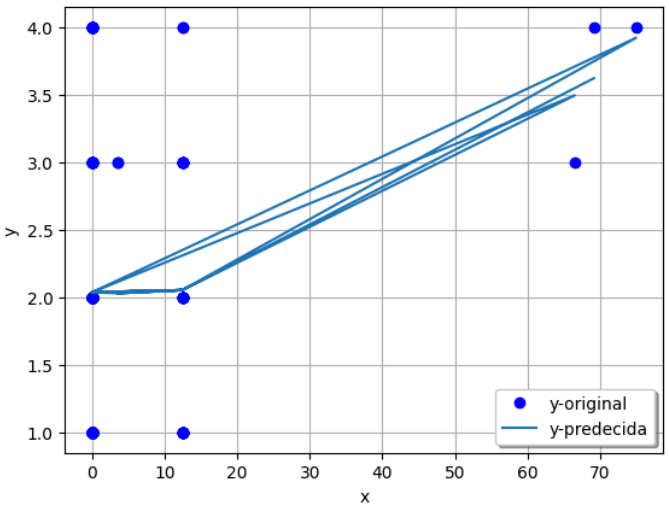
- **Denisse**


mini juego
color presionado
dificultad
botón correcto
tiempo de interacción
número de interacción
tiempo de sesión
tiempo de lección
mini juego 0.25 -0.24 0.06 0.22 0.01 0.39 0.16
color presionado 0.25 -0.06 -0.06 0.40 0.27 0.45 0.47
dificultad -0.24 -0.06 0.07 -0.16 0.07 0.16 -0.05
botón correcto 0.06 -0.06 0.07 -0.22 -0.21 -0.05 -0.06
tiempo de interacción 0.22 0.40 -0.16 -0.22 0.46 0.46 0.49
número de interacción 0.01 0.27 0.07 -0.21 0.46 0.33 0.35
tiempo de sesión 0.39 0.45 0.16 -0.05 0.46 0.33 0.39
tiempo de lección 0.16 0.47 -0.05 -0.06 0.49 0.35 0.39
1.0
0.8
0.6
0.4
0.2
0.0
−0.2


y-original
y-predecida
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
y
0
10
20
30
40
50
60
x


y-original
y-predecida
x
y


y-original
y-predecida
x
y


y-original
y-predecida
x
y