Proyecto de Tripulaciones

Memoria

(Data Science)

El presente proyecto se abordó por parte del equipo de Data Science atendiendo las necesidades del cliente ‘Ekhilur’, el cual requería los siguientes aspectos, crear una página web que permita a los usuarios acceder a estadísticas personalizadas sobre sus transacciones. Para ello, se utilizará una técnica de web scraping que obtendrá la información directamente de las cuentas de los usuarios. Que cuente con accesos personalizados así como de indicadores e información personalizada.

En atención a dichos requerimientos el trabajo inició por realizar un Web Scraping, el cual permitió acceder a toda la información necesaria, teniendo en cuenta que la información dada por el cliente se encontraba cifrada, al realizar esta acción le permitió al equipo entender con mayor facilidad el tipo de información con la que se tenía que trabajar.

En paralelo se coordinaron tareas con el equipo de Full Stack para definir la información que se tendría que presentar al usuario. Estas constan de lo siguiente:

- Resumen de gastos e ingresos por mes.
- Resumen total de gastos por categorías.
- Total de transacciones del usuario por mes.
- Modelo predictor para de CashBack.
- Cash back emitido y generado para los Clientes Ekhilur.
- Compras por categoría segmentadas por mes y año.
- Comparativa entre ingresos y gastos.
- Listado total de transacciones.
- Filtro de mes, año y categoría.
- Listado de ventas por mes y año para comercio.
- Monedero con el saldo actual de cada una de las cuentas.

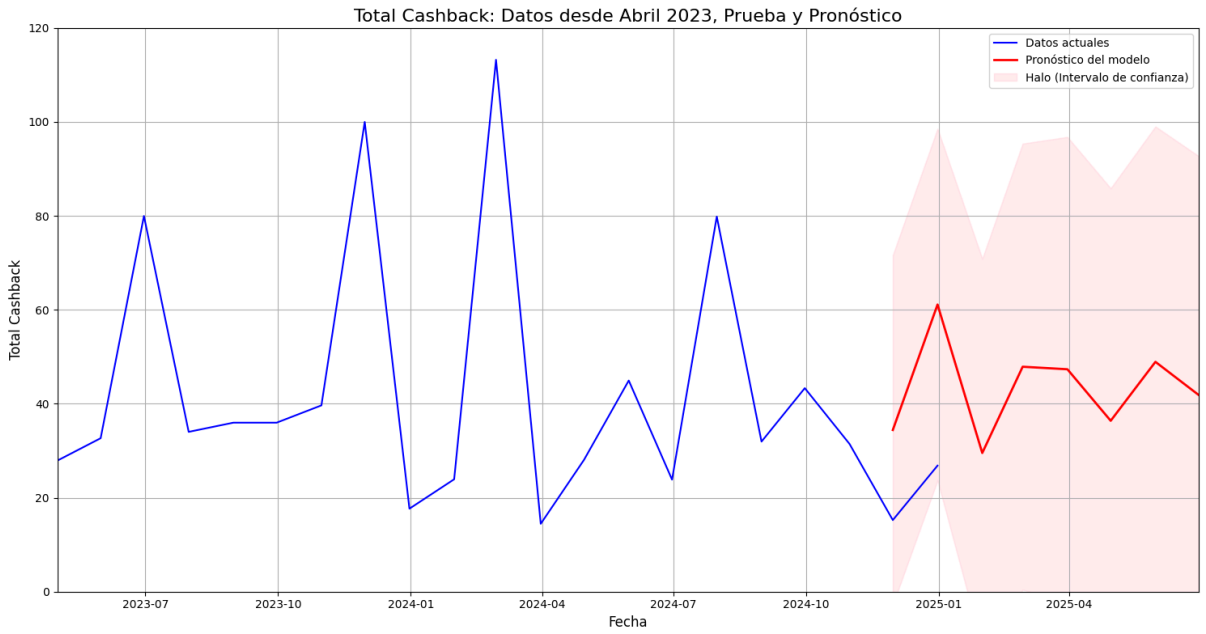
Este marco de acción exigió la creación de una base de datos en SQLite con los datos obtenidos del Web Scraping (parte fundamental del proceso) con lo cual se crearon las consultas necesarias en el programa mencionado anteriormente para después crear los diferentes ‘Endpoints’ con ‘Flask’ y realizar las pruebas pertinentes en la terminal para que dieran como resultado la visualización de los datos consultados.

Teniendo esta tarea realizada y confirmando con el equipo de Fullstack que era la información correcta, se remitieron los avances a dicho equipo para que pudieran iniciar con la visualización del proyecto.

Por su parte, el modelo predictor contó con múltiples desafíos; para desarrollar este modelo se tuvo en cuenta la distinción de dos tipos de usuarios, por un lado, uno que vive en Hernani y cuenta con dos tipos de CashBack, el otro que si bien usa la plataforma no reside en el Municipio y no cuenta con los mismos beneficios, por lo que la cantidad de data varía considerablemente. Es así que se tomó la decisión de usar un mismo modelo pero con un ajuste de hiperparametros diferentes para que se ajuste mejor a cada tipo de usuario.

El fin de este modelo consiste en que el cliente pueda ver el posible Cash Back del mes siguiente, este modelo tendrá en cuenta el historial de compra para poder predecir de forma acertada. En un primer acercamiento nos encontramos con modelos que no predecía de forma aceptable dado que la cantidad de datos con los que contamos no eran los suficientes, sin embargo pudimos tener logros sustanciales modificando los hiperparametros y usando un modelo Sarima (Seasonal AutoRegressive Integrated Moving Average), el cual incorpora componentes estacionales para modelar series temporales que presentan patrones estacionales. Si bien el modelo predice de manera razonable con una base de datos insuficiente, se proyecta que con el tiempo y la acumulacion de mas data el modelo trabaje de forma eficiente.

En la siguiente gráfica lo que se observa son, con color azul, los movimientos de Cash Back mes a mes. Por su parte, en rojo se observa cómo se comporta el modelo; bajo el parámetro de confianza (sombra en rojo) se muestra un margen muy amplio de error, lo que se traduce que el modelo, dado el volumen de datos, no se ajusta bien y su precisión es cuanto menos deficiente.

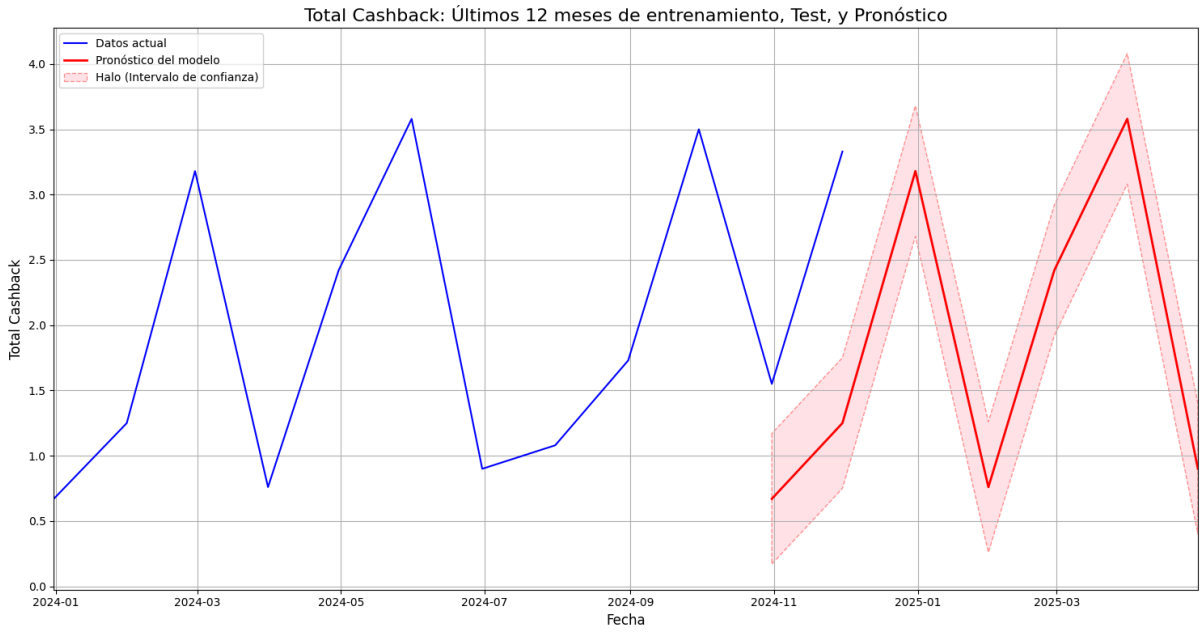


- Hiperparametros del usuario residente en Hernani.

```
# Define the best hyperparameters
order = (2, 1, 2)  # Best Order
seasonal_order = (0, 0, 0, 12)  # Best Seasonal Order

# Fit the SARIMA model
model = SARIMAX(
    train['Value'],
    order=order,
    seasonal_order=seasonal_order
)
results = model.fit(disp=False)  # Fit the model without displaying convergence messages
```

En línea con lo anterior, en el caso del usuario que habita fuera de Hernani vemos un comportamiento peculiar en el modelo. Si bien el intervalo de confianza se ajusta de manera óptima, se observa que el modelo no está prediciendo sino por el contrario está trasplantado datos y no prediciendo, debido al poco volumen de información con el que cuenta.



- Hiperparametros del usuario residente fuera de Hernani.

```
# Define the best hyperparameters
best_order = (2, 2, 0)
best_seasonal_order = (1, 0, 0, 12)

# Fit the SARIMAX model on the training data
model = SARIMAX(
    train['Total_Cashback'],
    order=best_order,
    seasonal_order=best_seasonal_order,
    enforce_stationarity=False,
    enforce_invertibility=False
)
results = model.fit(disp=False)
```

Recomendaciones
(Ciberseguridad)

Atendiendo a la importancia de la información que maneja Ekhilur y después de un análisis de riesgo, nuestro equipo se permite poner en su conocimiento las siguientes recomendaciones.

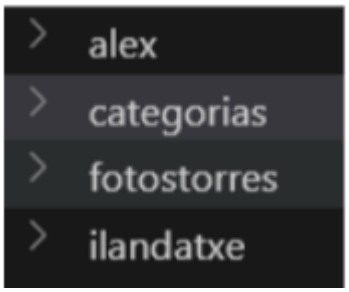
Tablas y nombres de usuarios

- Riesgo

1. Se ha encontrado que a lo largo de los ficheros, los nombres de las tablas que se emplean coinciden con los nombres de los usuarios, esto entraña el riesgo de poder realizar un ataque de fuerza bruta tras obtener los nombres de las tablas con herramientas como “sqlmap”.
2. Asociado a esto, se ha visto que los nombres de las tablas que emplea la aplicación están hardcodeadas directamente en el código, lo cual supone que la aplicación no es escalable.
3. En la BBDD, los nombres de los usuarios se presentan en texto plano, lo cual supone que cualquiera que pueda acceder a la BBDD puede conocer los nombres y apellidos de los usuarios que emplean la plataforma, esto supone una violación de la LOPDGDD (Ley Orgánica de Protección de Datos y Garantía de Derechos Digitales).
4. Las tablas intermedias o de información adicional se encuentran en la misma BBDD que las tablas con datos de los usuarios.

- Acciones sugeridas.

1. Emplear algún ID o clave para hacer referencia a las tablas con los datos de los usuarios sin que éstas lleven el propio nombre de usuario. Podría emplearse el propio hash (md5 por ejemplo) del nombre de usuario.



2. Emplear los usuarios de la BBDD de usuarios de Full Stack para crear las referencias a las tablas a medida que estas sean necesarias, de forma que a medida que se creen nuevos usuarios, se puedan crear nuevas tablas.

```
tablas_permitidas = {"ilandatxe", "fotostorres", "alex", "categorias"}
```

3. Implementar (con que quede desarrollada e indicado donde se aplicaría nos vale) una función que 'Hashee' los nombres de las personas que realizan transacciones en el sistema de Ekhilur. Implementado esto, debería crearse una tabla intermedia que relacione los hashes (o ids) con la persona en cuestión, esta deberá alimentarse a medida que se realicen 'scrapeos' y se vayan encontrando nuevos usuarios.

Cantidad	Concepto	Usuario Asociado	Saldo	Cuenta
15,00	Ekhilur kuota urtebete	Ekhilur S.Coop.	15,00	Ekhi Hernani
1,00	Froga	Alejandro Lopez Morgado	16,00	Ekhi
-1,00	froga	Alejandro Lopez Morgado	15,00	Ekhi
-1,00	froga	Herrilur Kontsumo Elkartea	14,00	Ekhi Hernani
1,00	NULL	Herrilur Kontsumo Elkartea	15,00	Ekhi

4. Las tablas de datos intermedias, tanto las de categorías como las de hashes o IDs de la que se ha hablado en el apartado anterior, deberían estar aisladas de la tabla de datos, de forma que no sea posible acceder a las tablas de datos en caso de comprometer las intermedias.

En conclusión, este trabajo por parte del equipo de Data Science logró a partir de la manipulación y visualización de los datos impulsar mejoras el rendimiento de la plataforma del cliente e integrar modelos predictivos que pueden llegar a ser de gran utilidad para los clientes en el futuro.