



THE
BOYS

PROPUESTA DE VALOR

MINIMIZAR COSTOS & MAXIMIZAR COBRANZA

NUESTRA MISIÓN



Nuestros objetivos a cumplir a lo largo de este evento fueron:

- Minimizar costo de comisiones, optimizando el número de envíos
- Maximizar la cobranza sin importar el costo

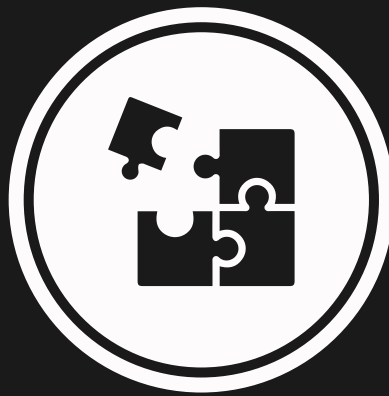


Identificar KPIs y puntos débiles dentro de la estructuración de la información.

Generar un modelo de cobranza que elimine el elemento humano.



OBJETIVOS



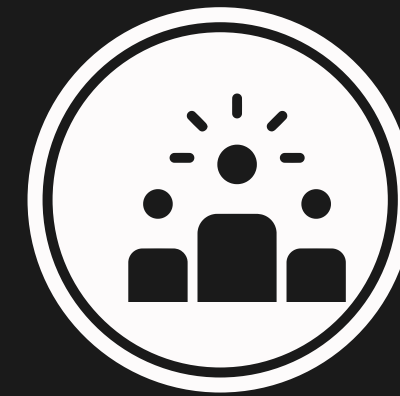
Rentabilidad

Buscamos maximizar la cantidad de cobranzas realizadas para potenciar la obtención de ROI en los créditos prestados.



Ganancias

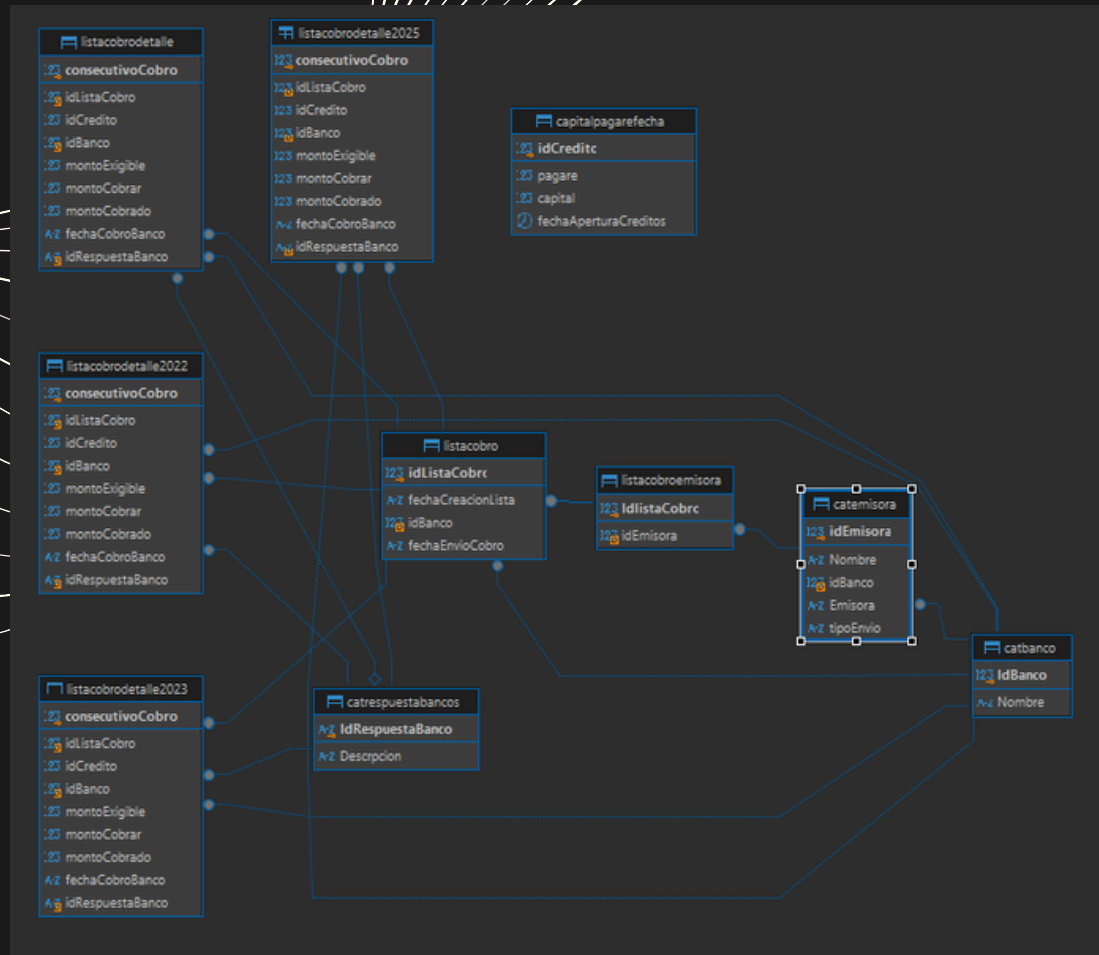
Minimizar la cantidad de intentos de cobro que generan un adeudo para la empresa, de esa manera afectando el margen de ganancias




Facilidad de uso


Un sistema similar al H2H, donde se subira un archivo de tipo csv donde se incluiran los datos clave para crear predicciones de los posibles casos de exito.

EL PROYECTO




 **Predicción de Cobros - Datathon**

Sube un archivo CSV para obtener predicciones de ahorro, ganancia y mejor emisora


 **Cargar archivo CSV para predicción**

Columnas requeridas: montoCobrar, montoCobrado, montoExigible, diaCobro, horaCobro

No file chosen

 **¡Comienza subiendo tu archivo CSV!**

El archivo debe contener las columnas: montoCobrar, montoCobrado, montoExigible, diaCobro, horaCobro

 **Notas importantes:**

- Asegúrate de que el backend Flask esté ejecutándose en puerto 5000
- Los modelos entrenados (h5 y pickle) deben estar en la carpeta backend/
- El archivo CSV no debe contener valores nulos

- Construir un modelo que recomiende la mejor emisora para realizar intentos de cobro bancario, considerando estrategias históricas, ahorro potencial y ganancias reales.
- Utilizar modelos de clasificación para segmentar a los clientes por comportamiento y entrenar redes neuronales para proyectar el costo, ahorro y emisora ideal.
- Predecir la emisora más efectiva para cada intento de cobro, basada en comisiones históricas, respuesta esperada y tipo de transacción.
- Entregar predicciones en formato CSV incluyendo idCredito, idEmisoraUsada, idEmisoraProyectada, pred_ahorro, pred_ganancia.

REFACORIZACION DE BD

- idCredito ordenado por ultima fecha de pago solamente insertando aquellos con un caso de exito

	123 idCredito	123 idBanco	A-2 nombre_banco	123 idEmisora	A-2 nombre_emisora	A-2 tipoEnvio	123 exitos	🕒 fecha_mas_reciente
1	606.411	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-11
2	599.306	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-11
3	507.710	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-11
4	459.208	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-11
5	525.580	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-11
6	530.806	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-11
7	538.436	44	SCOTIABANK	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
8	544.895	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
9	545.229	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
10	523.431	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
11	331.044	127	AZTECA	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
12	379.813	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
13	597.122	44	SCOTIABANK	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
14	600.248	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
15	410.057	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13
16	413.833	21	HSBC	20	BBVA CUENTA MATUTINO	MATUTINO	1	2022-01-13

	123 idCredito	123 intentos
1	484.058	100
2	490.604	100
3	508.588	100
4	579.193	100
5	484.958	100
6	600.033	100
7	566.380	100
8	111.844	100
9	351.181	100
10	467.517	100
11	120.843	100
12	532.719	100
13	506.794	100
14	594.541	100
15	519.118	100
16	476.721	100

- idCredito donde solamente insertan los que tienen cobros no validados

	123 idListaCobro	123 idCredito	123 consecutivoCobro	123 idBanco	123 montoExigible	123 montoCobrar	123 montoCobrado
1	79.678	139.713	23.794.379	44	851,01	851,01	0
2	79.678	161.287	23.794.380	44	205,63	205,63	0
3	79.678	67.455	23.794.389	21	735,73	735,73	0
4	79.678	70.077	23.794.390	21	313,37	313,37	0
5	79.678	70.845	23.794.391	21	708,48	708,48	0
6	79.678	75.623	23.794.392	21	211,19	211,19	0
7	79.678	80.296	23.794.393	21	170,31	170,31	0
8	79.678	112.266	23.794.394	21	350,05	350,05	0
9	79.678	116.619	23.794.395	21	463,24	463,24	0
10	79.678	117.474	23.794.396	21	592,67	592,67	0
11	79.678	118.984	23.794.397	21	395,12	395,12	0

- idCredito donde solamente se insertan aquellos con mas de 100 intentos de cobro fallidos

ESTADÍSTICAS

Decidimos implementar red neuronal de 2 capas por el motivo de la facilidad y una mayor accuracy.

Primera capa, fase logica de clasificación de datos convencional, logica de tiempo-respuesta-monto-fechaultimoPago-Pagare

Segunda capa, encargada de manejar los casos externos, encargada de una regresion lineal para encaje de datos.

```
Epoch 9/20
51/51 ————— 0s 5ms/step - accuracy: 0.8867 - loss: 0.2611 - val_accuracy: 0.8867 - val_loss: 0.2611
Epoch 10/20
51/51 ————— 0s 6ms/step - accuracy: 0.9038 - loss: 0.2418 - val_accuracy: 0.9038 - val_loss: 0.2418
Epoch 11/20
51/51 ————— 1s 6ms/step - accuracy: 0.9115 - loss: 0.2289 - val_accuracy: 0.9115 - val_loss: 0.2289
Epoch 12/20
51/51 ————— 1s 6ms/step - accuracy: 0.9149 - loss: 0.2145 - val_accuracy: 0.9149 - val_loss: 0.2145
Epoch 13/20
51/51 ————— 1s 6ms/step - accuracy: 0.9193 - loss: 0.2027 - val_accuracy: 0.9193 - val_loss: 0.2027
Epoch 14/20
51/51 ————— 0s 4ms/step - accuracy: 0.9221 - loss: 0.1984 - val_accuracy: 0.9221 - val_loss: 0.1984
Epoch 15/20
51/51 ————— 0s 4ms/step - accuracy: 0.9234 - loss: 0.1954 - val_accuracy: 0.9234 - val_loss: 0.1954
Epoch 16/20
51/51 ————— 0s 4ms/step - accuracy: 0.9284 - loss: 0.1824 - val_accuracy: 0.9284 - val_loss: 0.1824
Epoch 17/20
51/51 ————— 0s 4ms/step - accuracy: 0.9284 - loss: 0.1796 - val_accuracy: 0.9284 - val_loss: 0.1796
Epoch 18/20
51/51 ————— 0s 4ms/step - accuracy: 0.9296 - loss: 0.1756 - val_accuracy: 0.9296 - val_loss: 0.1756
Epoch 19/20
51/51 ————— 0s 4ms/step - accuracy: 0.9349 - loss: 0.1641 - val_accuracy: 0.9349 - val_loss: 0.1641
Epoch 20/20
51/51 ————— 0s 4ms/step - accuracy: 0.9344 - loss: 0.1620 - val_accuracy: 0.9344 - val_loss: 0.1620
<keras.src.callbacks.history.History at 0x79adfa18e5d0>
```

```
Epoch 7/20
56/56 ————— 0s 4ms/step - accuracy: 0.6216 - loss: 1.0874 - val_accuracy: 0.6192 - val_loss: 1.0874
Epoch 8/20
56/56 ————— 0s 4ms/step - accuracy: 0.6259 - loss: 1.0461 - val_accuracy: 0.6519 - val_loss: 1.0461
Epoch 9/20
56/56 ————— 0s 4ms/step - accuracy: 0.6585 - loss: 0.9717 - val_accuracy: 0.6621 - val_loss: 0.9717
Epoch 10/20
56/56 ————— 0s 4ms/step - accuracy: 0.6583 - loss: 0.9590 - val_accuracy: 0.6698 - val_loss: 0.9590
Epoch 11/20
56/56 ————— 0s 4ms/step - accuracy: 0.6755 - loss: 0.9164 - val_accuracy: 0.6783 - val_loss: 0.9164
Epoch 12/20
56/56 ————— 0s 4ms/step - accuracy: 0.6798 - loss: 0.8838 - val_accuracy: 0.6700 - val_loss: 0.8838
Epoch 13/20
56/56 ————— 0s 4ms/step - accuracy: 0.6812 - loss: 0.8555 - val_accuracy: 0.6908 - val_loss: 0.8555
Epoch 14/20
56/56 ————— 0s 4ms/step - accuracy: 0.6870 - loss: 0.8442 - val_accuracy: 0.6821 - val_loss: 0.8442
Epoch 15/20
56/56 ————— 0s 5ms/step - accuracy: 0.6854 - loss: 0.8300 - val_accuracy: 0.6864 - val_loss: 0.8300
Epoch 16/20
56/56 ————— 0s 4ms/step - accuracy: 0.6953 - loss: 0.8080 - val_accuracy: 0.6930 - val_loss: 0.8080
Epoch 17/20
```