Eduardo Trevelin Moreira da Silva

Sr. Data Analyst Case assessment

Digital Bank Mobile App Transaction events

The following is the case study of the Digital Bank Mobile App transactions regarding fraud activity and classification.

Index

1. Descriptive Analysis of Transaction Events........1
EDA, data manipulation, and overall view of data........1
Device age........5
Frequent use location........7
Device characteristics........7
Account_id and device_id........8
2. Fraud Risk Classification Rules:........9
High-Risk Transactions........9
Medium-Risk Transactions........10
Low-Risk Transactions........10
3. Evaluation of Classification Rules........10
New evaluation overview........10
Proposed flow assessment........11
Financial assessment........12
4. Recommendations and Conclusion........13
Annex........14
Jupyter Notebook........14
SQL snippets........15

1. Descriptive Analysis of Transaction Events

EDA, data manipulation, and overall view of data[1]

It was identified during the EDA process a few data inconsistencies, such as *null* values for the columns *'distance_to_frequent_location'*, *'is_emulator'*, *'has_root_permisions'*, *'app_is_tampered'* and '*transaction_id*'.

- *'distance_to_frequent_location'* , 1887 *null* values replaced with the **median** distance of all transactions made in the dataframe. The median value soothes outliers in distance:
    - `Mean: 14468.36`
    - `Median: 5.85`
- *is_emulator'*, *'has_root_permisions'*, *'app_is_tampered'* 68 *null* values in each column were replaced with False, as the value distribution in these columns is heavily favoring False. This can be revisited in the future in case we understand it affects future decision-making
- *'transaction_id'*
    - replaced ~139 rows with 0 values with dummy_ids by concatenating transaction_timestamp and account_id

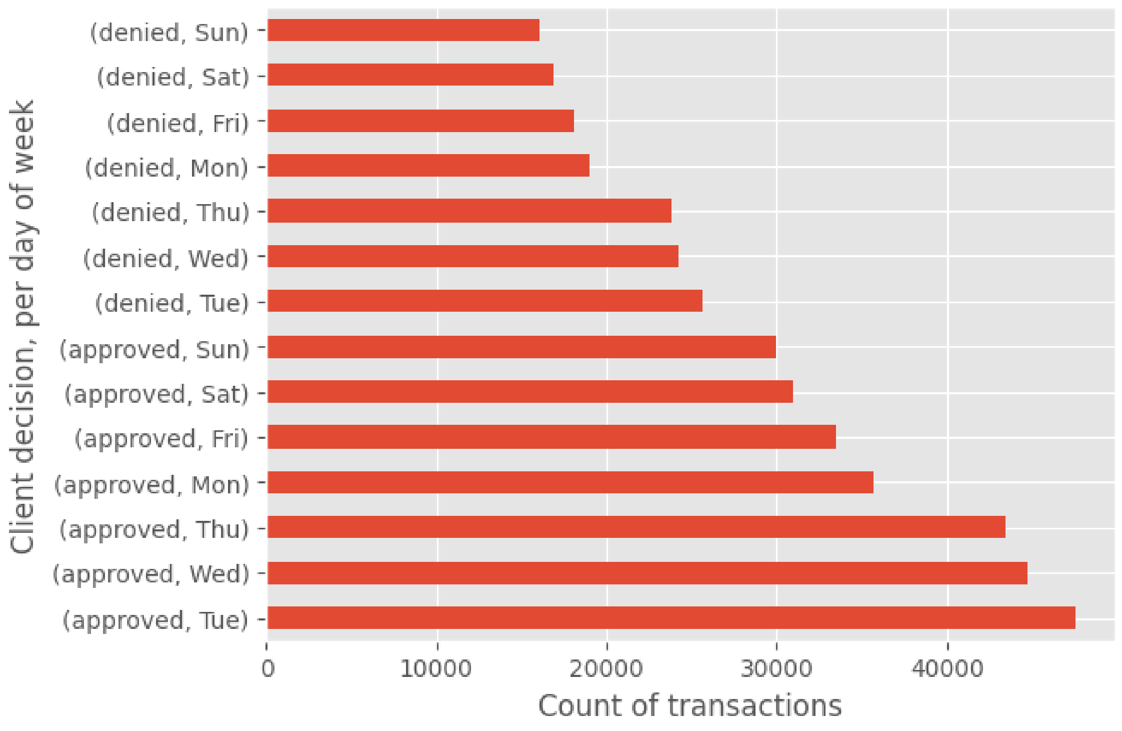
Columns were added in this process to leverage data and better understand the data pattern.

Added columns*:*

- '*approved_transaction_bool*' - True/False representation of the 'client_decision' column, boolean
- *'week_day'* - day of week acronym: (Mon, Wed, etc., varchar
- *'reference_date'* - YYYY-MM-DD representation of the event timestamp, varchar
- *'transaction_datetime'* - Date and hour of the event, timestamp
- '*transaction_hour_24'* - Hour of event, int

Added columns allows us to better understand data patterns, such as how many transactions are being approved or denied based on the weekday. It also allows visibility on possible current patterns not identified beforehand.

[1] EDA done in both Python and SQL (MySQL).



Mid-week (Tuesday, Wednesday, and Thursday) are the days with the most transactions made, with Tuesday the day with both the most approved and denied transactions.

Analyzing the hour of the days most transactions are made can provide insights on how 1. what is the user behavior on transactions and 2. when fraud activities are most being flagged currently regarding the hour of the day.

Table 1: Denied transactions per hour of the day, ordered by n_of_transactions

Client Decision	Hour of transaction	Nº of Transactions	Average transaction value, BRL
denied	23	9.683	R$ 219,54
denied	22	9.317	R$ 219,77
denied	0	9.083	R$ 211,74
denied	21	8.953	R$ 226,83
denied	18	8.710	R$ 225,14
denied	20	8.698	R$ 229,83
denied	19	8.606	R$ 229,85
denied	16	8.494	R$ 233,01
denied	17	8.487	R$ 238,67
denied	15	8.439	R$ 235,48
denied	1	7.969	R$ 218,63
denied	14	7.571	R$ 223,79
denied	13	6.392	R$ 202,76
denied	2	6.021	R$ 209,51
denied	12	5.262	R$ 189,95
denied	11	4.440	R$ 179,92
denied	3	4.341	R$ 204,01
denied	10	3.553	R$ 173,43
denied	4	2.600	R$ 207,49
denied	9	2.250	R$ 164,47
denied	5	1.646	R$ 211,54
denied	8	1.223	R$ 167,42
denied	6	1.087	R$ 213,22
denied	7	857	R$ 192,78

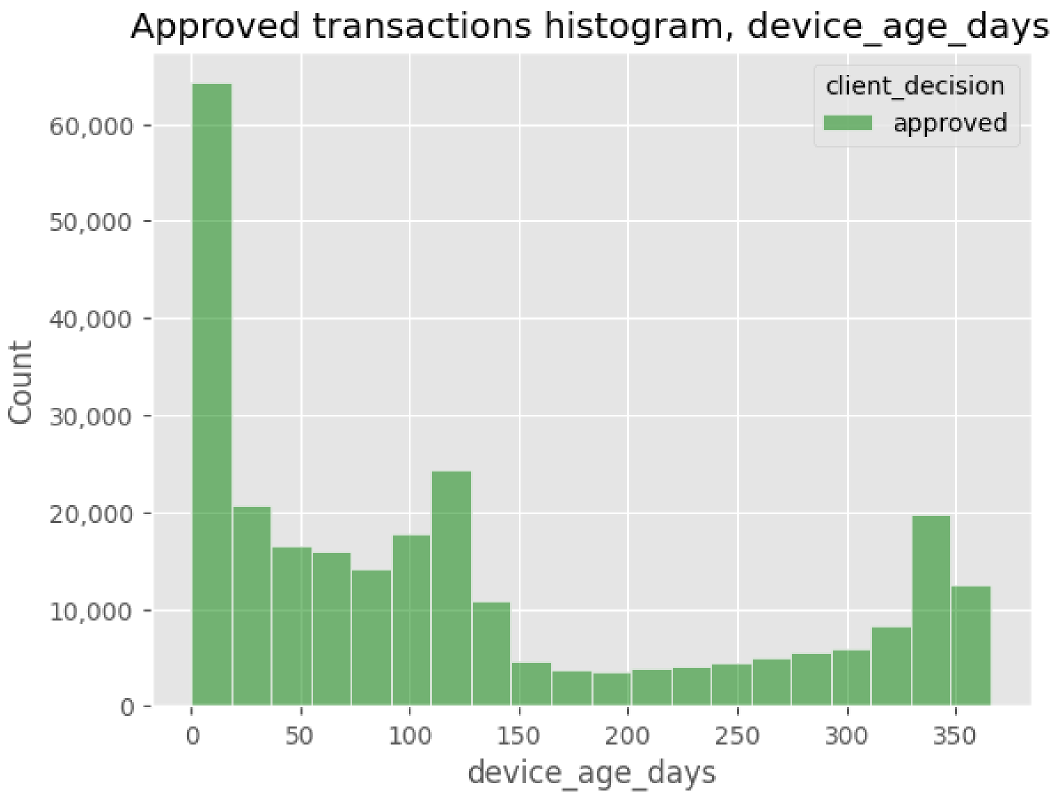
Table 1 shows that most of the denied transactions are flagged between 17h (5 PM) and 1h (1 AM), peaking at 23h. This indicates that most likely fraudster activity is currently being mapped with an emphasis on the time of transaction.

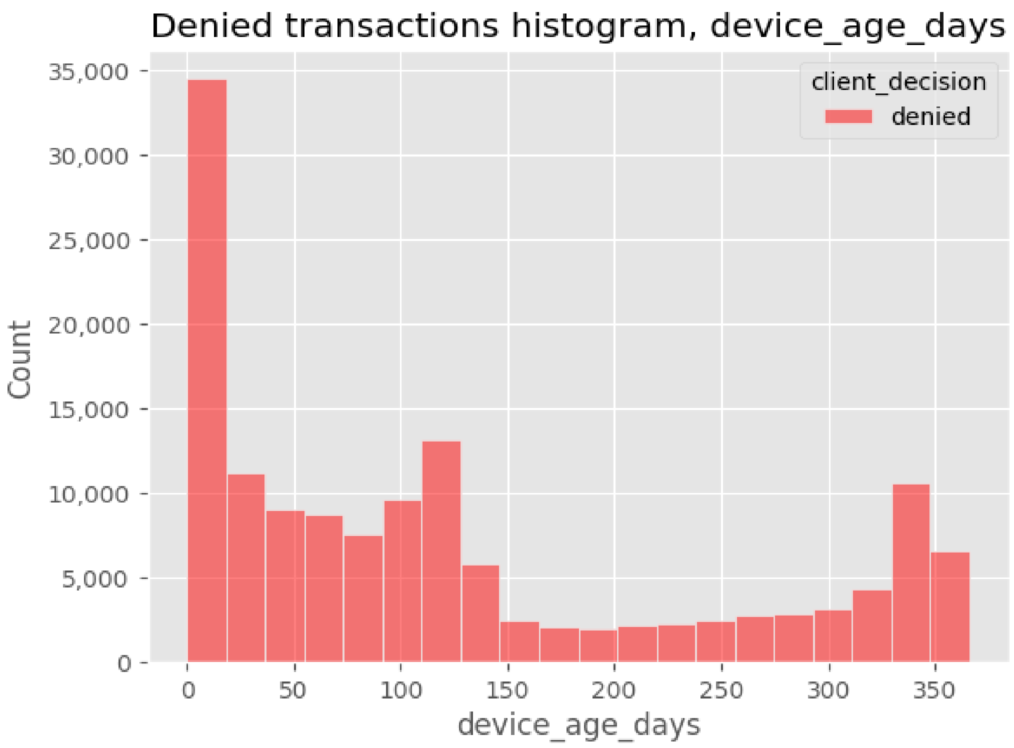
Table 2: transactions per hour of day, ordered by n_of_transactions

Hour of transaction	Nº of Transactions	Average transaction value, BRL
17	24.173	R$ 237,60
16	24.225	R$ 234,15
15	24.018	R$ 232,74
19	24.645	R$ 232,70
20	24.814	R$ 229,91
18	24.517	R$ 229,75
14	21.584	R$ 224,68
1	22.936	R$ 220,94
23	27.282	R$ 220,66
21	25.729	R$ 220,06
22	26.624	R$ 217,86
0	25.713	R$ 212,27
2	17.317	R$ 211,94
3	12.328	R$ 211,89
5	4.778	R$ 211,39
13	18.275	R$ 207,38
4	7.589	R$ 205,74
6	3.180	R$ 204,99
7	2.506	R$ 195,94
12	15.013	R$ 191,22
8	3.512	R$ 182,63
11	12.514	R$ 176,57
10	9.792	R$ 172,06
9	6.360	R$ 165,28

Device age

Understanding now the distribution of transactions, both approved and denied, by age of device, we can observe trends in usage by possible fraudsters activity.



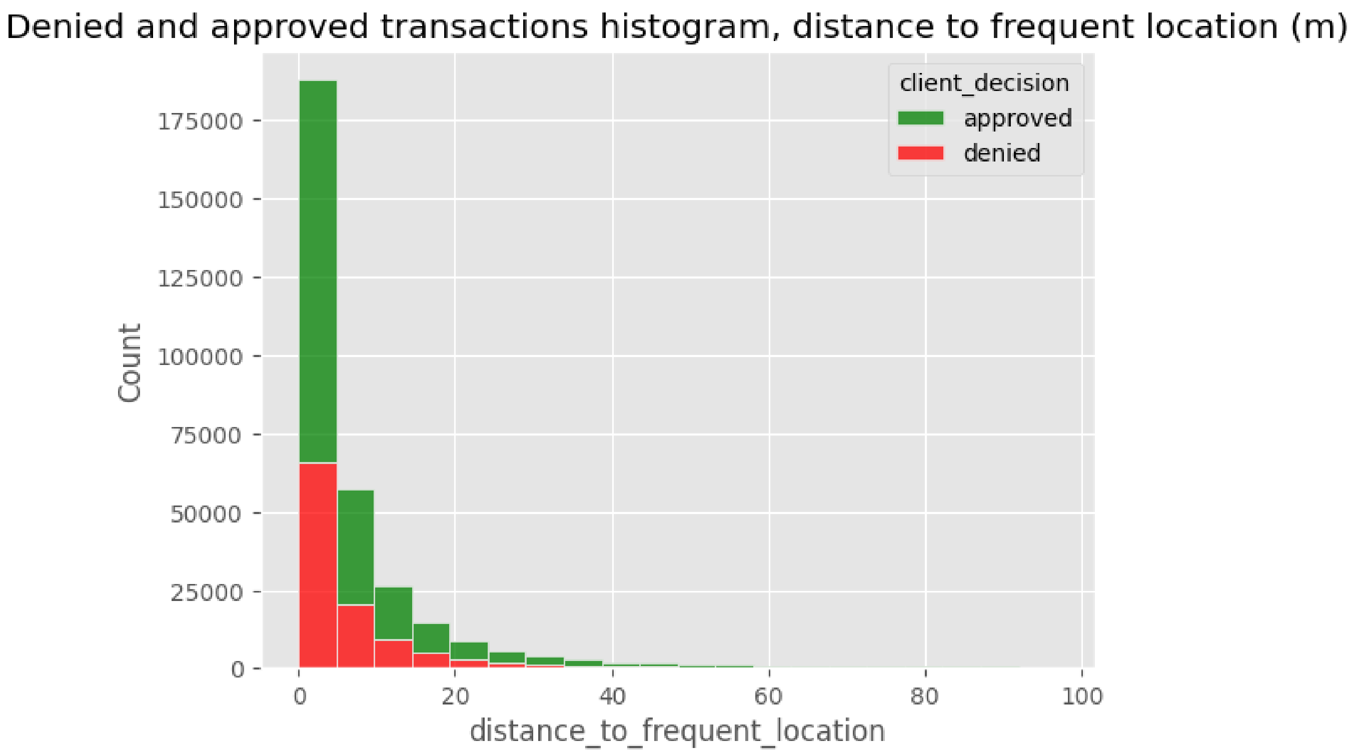


Although there is the prevalence of newer device usage overall, there is a pattern that the transactions are more prevalent in the first days of the device, then go up again around the 100th day up until the 150th day of age, and then spike again near the 350th day for both approved and denied transactions.

Frequent use location

In regards to distance to frequent, safe location in meters, we can identify that most transactions are done near the location. This can indicate that:

1. denied transactions are happening in non-fraudulent transactions, which can impact user experience - i.e. when a new user (device_age_days near to 0) is trying to do their first transactions.
2. fraudsters are often using new devices, which means that the most frequent location transactions will be closer to the fraud location.



Device characteristics

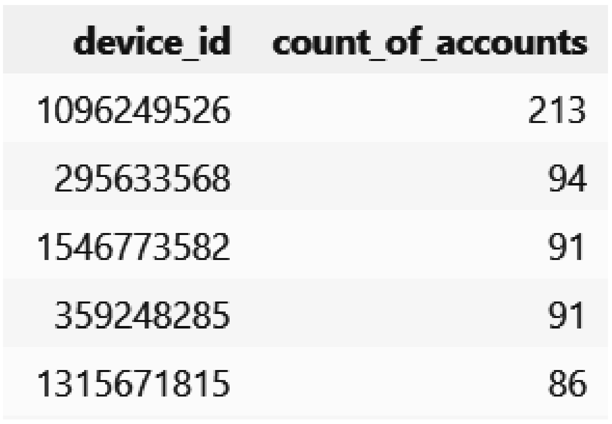
Analysing categorical value columns - ‘app_is_tampered’, ‘has_root_permissions’, ‘is_emulator’, and ‘has_fake_location’ -, it appears that these columns have currently little to no impact on denying or approving the transactions, as per the following table. This can mean that transactions that are fraudulent are considered normal transactions (Type 2 error), and non-fraudulent transactions are considered fraudulent (Type 1 error). Since both types of error have their own implications and trade-offs, we must better understand how to balance them and minimize risks while mitigating financial and user experience losses.

Table 3: categorical data analysis, ordered by count of transactions.

app_is_tampered	has_root_permissions	is_emulator	has_fake_location	client_decision	count
FALSE	FALSE	FALSE	FALSE	approved	265.241
FALSE	FALSE	FALSE	FALSE	denied	143.418
FALSE	TRUE	FALSE	FALSE	approved	260
FALSE	TRUE	FALSE	FALSE	denied	161
TRUE	FALSE	FALSE	FALSE	approved	91
FALSE	FALSE	TRUE	TRUE	approved	72
FALSE	FALSE	FALSE	TRUE	approved	55
TRUE	FALSE	FALSE	FALSE	denied	48
FALSE	FALSE	FALSE	TRUE	denied	25
FALSE	TRUE	TRUE	TRUE	approved	21
FALSE	FALSE	TRUE	TRUE	denied	20
FALSE	TRUE	TRUE	TRUE	denied	10
FALSE	TRUE	FALSE	TRUE	approved	1
TRUE	FALSE	FALSE	TRUE	approved	1

Account_id and device_id

When analyzing the distribution of account per devices, it is assumed that one personal device should be connected to one account for safety measures (Face ID, fingerprint verification, etc.). There are currently 36,606 devices with more than one account linked, with most being linked to two accounts 22,448. Extreme cases, such as device_id **1096249526**, linked to 213 accounts, resulted in 136 approved transactions and only 77 denied transactions.



device_id	count_of_accounts
1096249526	213
295633568	94
1546773582	91
359248285	91
1315671815	86

Ideally, in this case, it should be being flagged by the system as an emulator - nonetheless, no rows for ‘is_emulator’ were tagged as *True* for this device_id. This can be further investigated to understand if there are possibilities of creating a flag for when the device being used has more than one account linked to it or even adding this to a future risk classification method.

As for the count of *account_id* per *device_id*, only 30 accounts were flagged as having more than one *device_id* - this can be a case of device change during the observed period.

2. Fraud Risk Classification Rules:

Below is presented a set of rules that can be implemented to change how decision-making is being done today.

High-Risk Transactions

1. **Unusual Device Activity:**
   - If is_emulator is True, it's a high-risk transaction. Emulators are often used for fraudulent activities to disguise as physical devices.
   - If has_root_permissions is True, consider it high-risk. Rooted devices can pose a security threat as they possess elevated privileges to make system-level changes to reduce security levels, for example.
   - Flagging if the device has more than one account linked to it should raise a warning and tag the device as high-risk of being an emulator since as discussed, there are cases not being mapped by that column.
      i. After testing, it was decided to flag them as high-risk devices with more than 5 accounts linked.
      ii. This was due to concerns of a big impact on user experience. Scenarios:
         1. More than one account linked: **124,547** transactions flagged as **High-Risk**
         2. More than five accounts linked:**46,479** transactions flagged as **High-Risk**
2. **Suspicious Locations and Timestamp**

- If has_fake_location is True, mark it as high risk. Fake location data may indicate malicious intent. Also, as mentioned above, transactions made in unusual hours in the '*transaction_hour_24'* can be an indicator of fraud.

3. **App Tampering:**
    - If app_is_tampered is True, classify it as high risk. Tampered apps can lead to unauthorized access as they can be modified in their source code or jailbreak.
4. **Large Transaction Value:**
    - Setting a threshold for transaction_value, for example, BRL 4,999 (in Brazil, max limit, transactions above a certain amount can be flagged as high risk. Large transactions are more frequent in fraud activities.

Medium-Risk Transactions

1. **Device Age:**
    - If device_age_days is relatively new but not suspicious (determined by domain-specific criteria), classify it as medium risk. Newer devices may be more susceptible to fraud activities.
2. **Distance from Frequent Location:**
    - Set a threshold for distance_to_frequent_location above which transactions are considered medium risk. Unusually distant transactions can be moderately risky.
3. **Unusual device activity**
    - As mentioned above, we are setting the threshold for over than 5 account_id per device_id as High Risk. Devices with more than one and up to 5 accounts linked will be flagged as Medium Risk transactions

Low-Risk Transactions

All other transactions that are qualified as High and Medium risk transactions.

More in-depth logic for evaluation is available in the Annex section.

3. Evaluation of Classification Rules

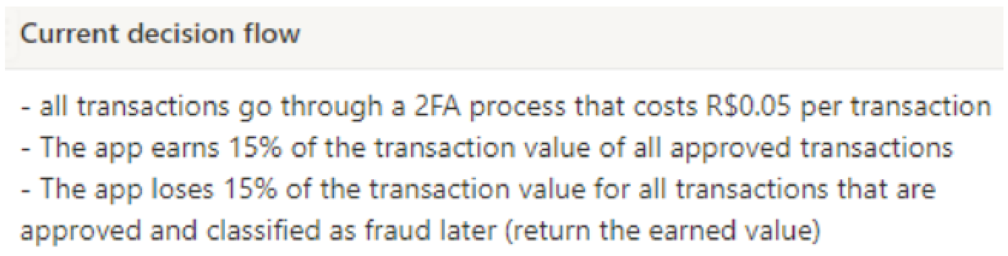
New evaluation overview

After running the proposed risk classification into the provided data, this is the identified distribution between Low, Medium, and High risk transactions:

Table 4: Updated classification

Classification	Nº of transactions	Percentage
Low Risk	279.150	68,18%
Medium Risk	83.795	20,47%
High Risk	46.479	11,35%

Based on the current decision flow below, we must minimize the **Type 2** (false negatives) errors to reduce financial costs (15% of the transaction value).



Current decision flow

- all transactions go through a 2FA process that costs R$0.05 per transaction
- The app earns 15% of the transaction value of all approved transactions
- The app loses 15% of the transaction value for all transactions that are approved and classified as fraud later (return the earned value)

At the same time, it is advised not to overestimate **Type 1** (false positives), as it both impacts the app's revenue stream (15% of approved transactions value) and the user experience - non-fraudulent activity being constantly flagged as fraud hinders the user relation and his/her trust to Digital Mobile. This Type 1 overestimation is most likely what is being observed in the current process, where 35.09% of the transactions are being denied at face value, instead of 11.35% in the proposed risk assessment classification.

Table 5: Summary of current decision-making

Client Decision	Nº of transactions	Percentage
approved	265.742	64,91%
denied	143.682	35,09%

Table 6: Current decision-making against fraud_feedback dataset

Fraud check	Client Decision	Nº of transactions	Evaluation
not fraud	approved	23,8308	Ok
not fraud	denied	128,715	False Positive
fraud	approved	27,434	False Negative
fraud	denied	14,967	Ok

Proposed flow assessment

The proposed solution would be to deny transactions marked as High-risk at first and approve Medium and Low-risk transactions - Medium-risk level transactions will be investigated in detail to understand if they possess fraudulent activity patterns and reinforce fraud algorithms.

The results of the new proposed classification against the fraud_transactions_feedback dataset, which includes only the actual fraudster *transaction_id* are presented in Table 7 below.

False Positive frauds went down by 85%, and False Negative by 47.4% compared to the decision-making rationale currently in place. This is significant in regards to user experience, as the drop in False Positives should improve the user journey on the Digital Banking Mobile App, by decreasing incorrectly denied transactions.

Table 7: Implemented decision-making against fraud database

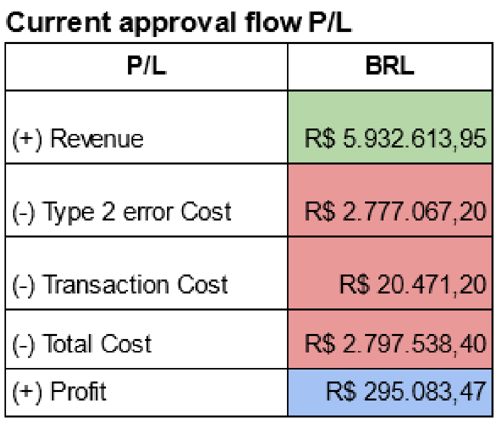
Fraud Check	Updated risk classification	Nº of transactions	Type
not fraud	High Risk	18.510	False Positive
not fraud	Low Risk	274.452	Ok
not fraud	Medium Risk	74.061	Ok
fraud	High Risk	27.969	Ok
fraud	Low Risk	4.698	False Negative
fraud	Medium Risk	9.734	False Negative

There is a room from improvement regarding False Negatives errors in Low and Medium-Risk transactions. Fine-tuning the parameters for risk classification should reduce these cases.

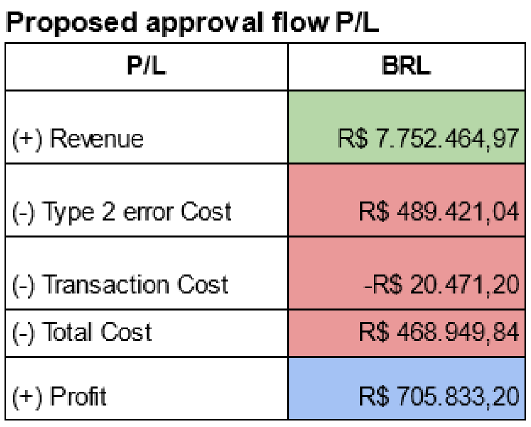
Financial assessment

Evaluating the current approval and the new, proposed approval flow against the ***fraud_feedback*** dataset.

R: Revenue
TC: Total Cost
TrC: Transaction Cost
FnC: Type 2 error Cost (False Negative)
P: Profit
R = Sum of correct approved transactions X 0.015
TC = TrC + FnC
FnC = Nº of transactions X 0.05
FnC = Sum of Type 2 transactions X 0.015
P = R - TC


Current approval flow P/L

P/L	BRL
(+) Revenue	R$ 5.932.613,95
(-) Type 2 error Cost	R$ 2.777.067,20
(-) Transaction Cost	R$ 20.471,20
(-) Total Cost	R$ 2.797.538,40
(+) Profit	R$ 295.083,47


Proposed approval flow P/L

P/L	BRL
(+) Revenue	R$ 7.752.464,97
(-) Type 2 error Cost	R$ 489.421,04
(-) Transaction Cost	-R$ 20.471,20
(-) Total Cost	R$ 468.949,84
(+) Profit	R$ 705.833,20

The proposed approval flow could increase the profit by **R$ 410.749,73** (139% increase) if applied in the evaluated month. This results from a reduction in both False Positives (Type 1) and False Negatives (Type 2) number of transactions, simultaneously driving revenue up and reducing costs.

4. Conclusion and Next Steps

Overall, the data shows us that the current decision flow has two main issues:

1. A serious user experience impact as it is denying most non-fraudulent transactions from users. This may be happening as the current decision flow from approval relies heavily on the hour-of-day transaction, as shown in Table 1. This also impacts the revenue stream, since the Digital Bank Mobile App has zero revenue if a transaction is a Type 1 error.
2. Even with the zealous approach of denying a considerable number of transactions, there is still a good number of False Negatives, which in turn impacts negatively the results as it drives reimbursement costs up.

Given the presented data, the recommended course of action is to start applying the proposed approval flow of denying all High-Risk transactions and approving all Medium and Low-risk transactions.

In a second moment, we should aim for a fine-tuning process of the parameters associated with each risk category by applying state-of-the-art techniques for fraud detection and raising the correct flags for each fraud detection column. This fine-tuning is aimed at decreasing even further the amount of False Negative values, thus reducing costs and driving profits for the Digital Bank Mobile App.

Annex

Jupyter Notebook

Python EDA step available on GitHub.

SQL snippets

MySQL format. Tables uploaded locally via Python (sqlalchemy and Pandas libraries).

Creating transaction_hour_24 column

```
alter table incognia_database.digital_banking_data
add column transaction_hour_24 int;
update incognia_database.digital_banking_data
set transaction_hour_24 = hour(transaction_datetime);
```

Table 1 snippet

```
select
client_decision
,transaction_hour_24
,count(transaction_id) as n_of_transactions
,round(avg(transaction_value),2) as average_transaction_value
from incognia_database.digital_banking_data
where client_decision = 'denied'
group by 1,2
order by 3 desc;
```

Table 2 snippet

```
select
transaction_hour_24
,count(transaction_id) as n_of_transactions
,round(avg(transaction_value),2) as average_transaction_value
from incognia_database.digital_banking_data
group by 1
order by 3 desc;
```

Number of accounts per device_id threshold definition

More than 1

```
select sum(d.transaction_id_count) as sum_of_transactions
        from (
         select
         device_id,
        count(distinct transaction_id) as transaction_id_count,
         count(account_id)
         from incognia_database.digital_banking_data
         group by device_id
         having count(account_id)>1) d;
```

More than 5

```
select sum(d.transaction_id_count) as sum_of_transactions
          from (
           select
           device_id,
          count(distinct transaction_id) as transaction_id_count,
           count(account_id)
           from incognia_database.digital_banking_data
           group by device_id
           having count(account_id)>5) d;
```

New proposed risk classification overview

```
select
count(db.transaction_id)
,case
   -- categorical values filters
   when (
           db.is_emulator = true OR
           db.has_root_permissions = true OR
           db.has_fake_location = true OR
           db.app_is_tampered = true) then 'High Risk'
          -- transaction value treshold based on the transaction hour -> high risk hours with
less transaction_value tolerance
          when
           (db.transaction_hour_24 BETWEEN 4 AND 20 AND db.transaction_value > 10000)
OR
           (db.transaction_hour_24 < 4 AND db.transaction_value > 1000) OR
           (db.transaction_hour_24 > 20 AND db.transaction_value > 1000) then 'High Risk'

          -- treshold for distance_from frequent_location - 50km, paired with high
transaction_value
          when
           (db.transaction_value > 1000 and db.distance_to_frequent_location > 50000) then
'High Risk'
```

```
        -- n of accounts per device greater than one. Subquery to extract the device_ids
        -- for less impact to user experience, we're defining riskier transactions with devices
with more than 5 account_ids linked
        when
         db.device_id in (select d.device_id
         from (
                select
                device_id,
                count(account_id)
                from incognia_database.digital_banking_data
                group by device_id
                having count(account_id)>5) d) then 'High Risk'
        when
         -- low device age paired with high transaction value as Medium risk
        (db.device_age_days <10 and db.transaction_value > 1000) then 'Medium Risk'
        when
         -- greater values for distance_to_frequent_location
         db.distance_to_frequent_location > 50000 then 'Medium Risk'
   when
         db.device_id in (select d.device_id
         from (
                select
                device_id,
                count(account_id)
                from incognia_database.digital_banking_data
                group by device_id
                having count(account_id)>1  and count(account_id)<=5) d) then 'Medium
Risk'
        else
        'Low Risk'
         end as updated_risk_classification
from incognia_database.digital_banking_data db
group by 2
order by 1 desc
limit 100;
```

Addin updated_risk_classification column to table

```
alter table incognia_database.digital_banking_data
add column updated_risk_classification varchar(255);

update incognia_database.digital_banking_data
set updated_risk_classification = hour(transaction_datetime);

update incognia_database.digital_banking_data
```

```
set updated_risk_classification = (case
   -- categorical values filters
   when (is_emulator = true OR
          has_root_permissions = true OR
          has_fake_location = true OR
          app_is_tampered = true) then 'High Risk'
         -- transaction value threshold based on the transaction hour -> high risk hours with
less transaction_value tolerance
         when
          (transaction_hour_24 BETWEEN 4 AND 20 AND transaction_value > 10000) OR
          (transaction_hour_24 < 4 AND transaction_value > 1000) OR
          (transaction_hour_24 > 20 AND transaction_value > 1000) then 'High Risk'

         -- threshold for distance_from frequent_location - 50km, paired with high
transaction_value
         when
          (transaction_value > 1000 and distance_to_frequent_location > 50000) then 'High
Risk'
         -- Number of accounts per device greater than one. Subquery to extract the
device_ids
         -- for less impact on user experience, we're defining riskier transactions with devices
with more than 5 account_ids linked
         when
          device_id in (select d.device_id
          from (
                  select
                  device_id,
                  count(account_id)
                  from incognia_database.digital_banking_data
                  group by device_id
                  having count(account_id) > 5) d) then 'High Risk'
         when
          -- Low device age paired with high transaction value as Medium risk
         (device_age_days < 10 and transaction_value > 1000) then 'Medium Risk'
         when
          -- Greater values for distance_to_frequent_location
          distance_to_frequent_location > 50000 then 'Medium Risk'
   when
          device_id in (select d.device_id
          from (
                  select
                  device_id,
                  count(account_id)
                  from incognia_database.digital_banking_data
                  group by device_id
```

```
                    having count(account_id) > 1  and count(account_id) <= 5) d) then 'Medium
Risk'
          else
          'Low Risk'
           end);
```

Evaluating of risk assessment classification against fraud dataset

```
-- not fraud selection
(select
'not fraud' as fraud,
updated_risk_classification,
count(distinct db.transaction_id) as count_of_transactions
from incognia_database.digital_banking_data db
left join incognia_database.fraud_transactions_feedback ft on db.transaction_id =
ft.transaction_id
where ft.transaction_id is null
group by 1,2)
union
-- fraud selection
 (
select
'fraud' as fraud,
updated_risk_classification,
count(distinct db.transaction_id) as count_of_transactions
from incognia_database.digital_banking_data db
left join incognia_database.fraud_transactions_feedback ft on db.transaction_id =
ft.transaction_id
where ft.transaction_id is not null
group by 1,2);
```

Current decision flow P/L calculation

```
-- current decision flow financial impact

with a as (
select
          -- no fraud, revenue calculation
          round((select
          sum(transaction_value)*0.15
          from incognia_database.digital_banking_data db
```

```
        left join incognia_database.fraud_transactions_feedback ft on db.transaction_id =
        ft.transaction_id
        where ft.transaction_id is null
        and db.client_decision='Approved'),2) as revenue
-- type 2 error cost calculation
        ,round((
        select
        sum(transaction_value)*0.15
        from incognia_database.digital_banking_data db
        left join incognia_database.fraud_transactions_feedback ft on db.transaction_id =
        ft.transaction_id
        where ft.transaction_id is not null
        and db.client_decision ='Approved'),2) as type_2_error_cost
-- transaction cost calculation
,count(transaction_id)*0.05 as transaction_cost
from incognia_database.digital_banking_data)
select *,
round(revenue - type_2_error_cost - transaction_cost,2) as profit
from a
;
```

Proposed flow P/L calculation

```
-- proposed flow P/L
with a as (
select
-- no fraud, revenue calculation
round((select
sum(transaction_value)*0.15
from incognia_database.digital_banking_data db
left join incognia_database.fraud_transactions_feedback ft on db.transaction_id =
ft.transaction_id
where ft.transaction_id is null
and db.updated_risk_classification <> 'High Risk'),2) as revenue
-- type 2 error cost calculation
,round((
select
sum(transaction_value)*0.15
from incognia_database.digital_banking_data db
left join incognia_database.fraud_transactions_feedback ft on db.transaction_id =
ft.transaction_id
where ft.transaction_id is not null
-- if False Negative, risk classification must be either Low or Medium risk
and db.updated_risk_classification <> 'High Risk'),2) as type_2_error_cost
,count(transaction_id)*0.05 as transaction_cost
from incognia_database.digital_banking_data)
```

```
select *,
round(revenue - type_2_error_cost - transaction_cost,2) as profit
from a
;
```