REVOLUT HOME CHALLENGE

Pietro Alessandro Aluffi

October 13, 2020

Contents

1 Task 1

It is mandatory for all new Revolut users to be verified before an account can be opened. The verification process consists of clearing two checks. The first one is a document check whereas the second one is a facial check to make sure that the documents provided belong to the applicants. Each customer has a maximum of two attempts to get approved. These checks are carried out by a third party: Veritas. It has been brought to our attention that in the recent period the pass rate has been decreasing significantly.

In order to understand the reasons behind the decrease in approvals it is useful to visualize the data graphically. Figure 1 represents the *overall*, the *document* and *facial pass rates* over time. It is clear to see that what lowers the overall pass rate is the document check. Therefore, I aim to dig deeper into this in order to understand the causes.

Figure 2 and Figure 3 reports the pass rates for *gender* and *nationality.* It is clear that neither of these characteristics are a cause for failing documents check as both lines follow the overall trend.

Great indicators of a decrease in the pass rate for documents checks are *Face Detection*, *Document Quality*, *Sub Result.* Figure 4, 5, 6 report their pass rate, respectively. It can be seen that there is an increase in documents not clearing those tests which eventually lead to the decrease in the overall pass rate.

To conclude, the drop in pass rate is generated by failure of the document check. The causes for an increase in documents being rejected are documents with poor quality or those ones with problems matching the face on the documents. There is no change in the demographic of the user, hence the causes for this must rely in the service provided by Veritas. Therefore, with the data available it is possible to make a statistically significant conclusion.

In order to solve this issue it is necessary to gather more data in order to discover patterns that lead to the increase of the number of documents labeled as *caution* or *rejected* by the Veritas API. If the problem relies on the Veritas algorithm it is crucial that they improve the recognition algorithm and make sure it is not faulty. On the Revolut side, an effort should be spent providing a better User Experience as well as User Interface which helps the customer take clearer and better quality photos. In the event that the user fails the first attempt it is important that he is encouraged to take the second by improving the quality of the image.

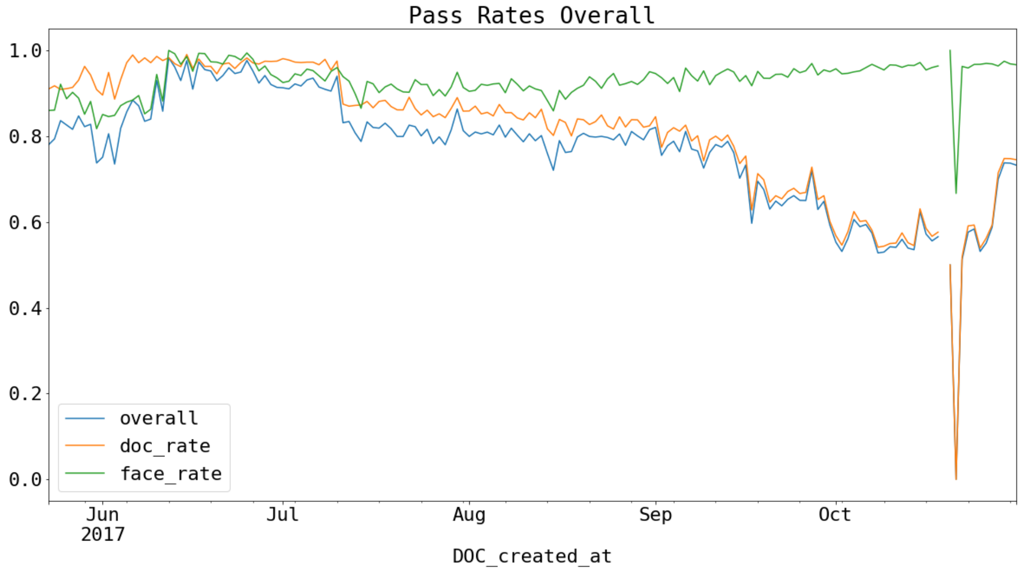

Figure 1

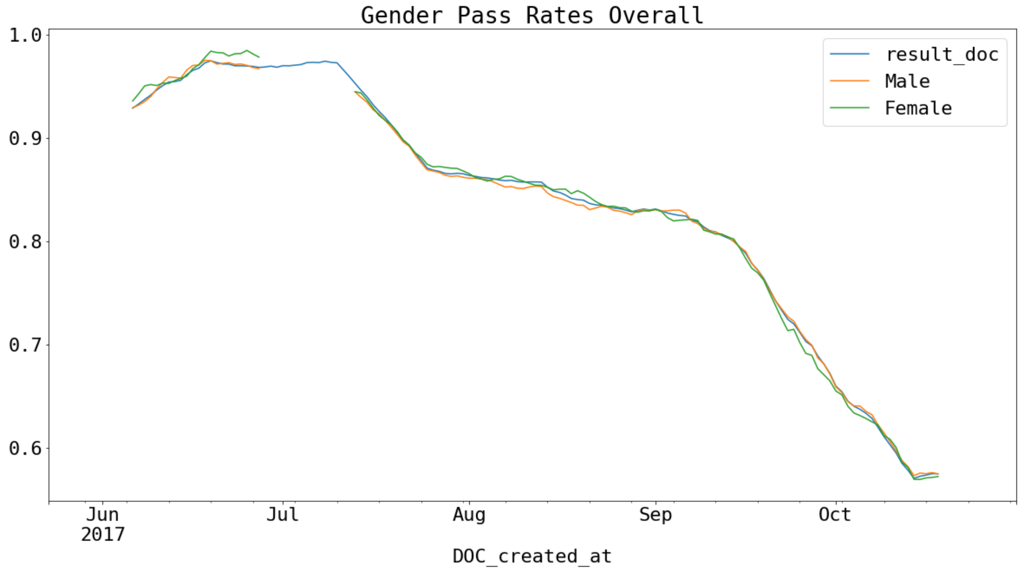

Figure 2

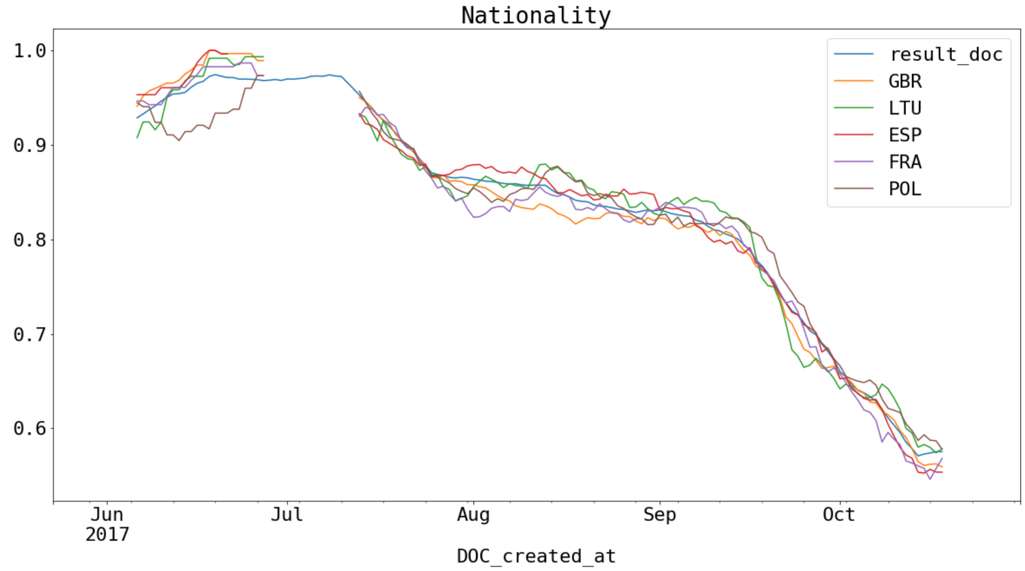

Figure 3

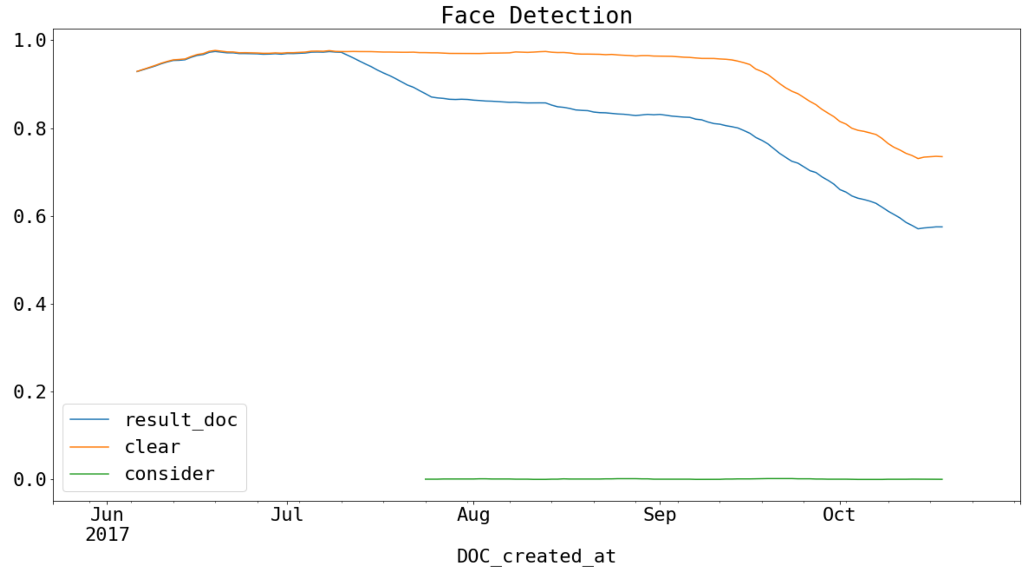

Figure 4

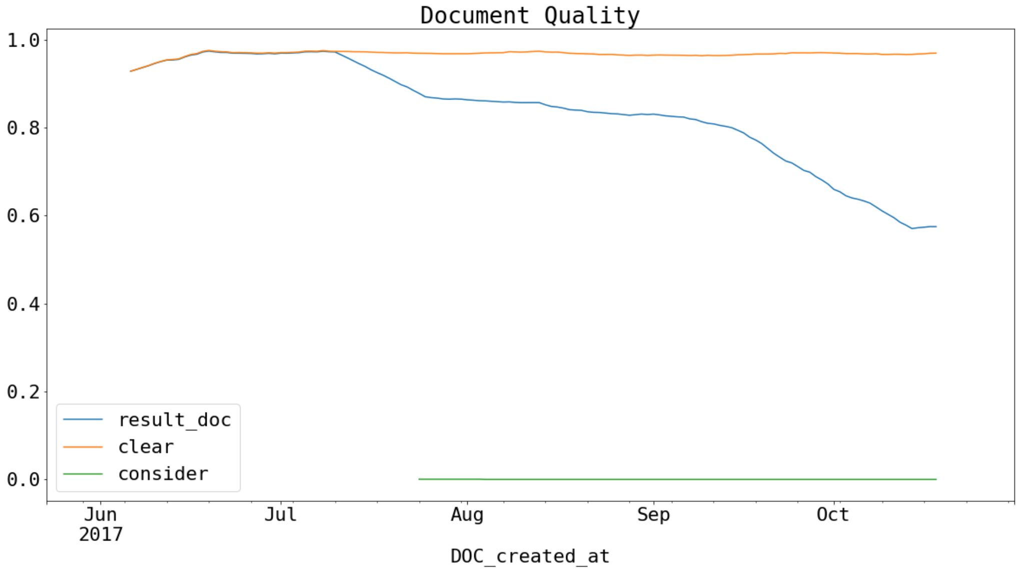

Figure 5

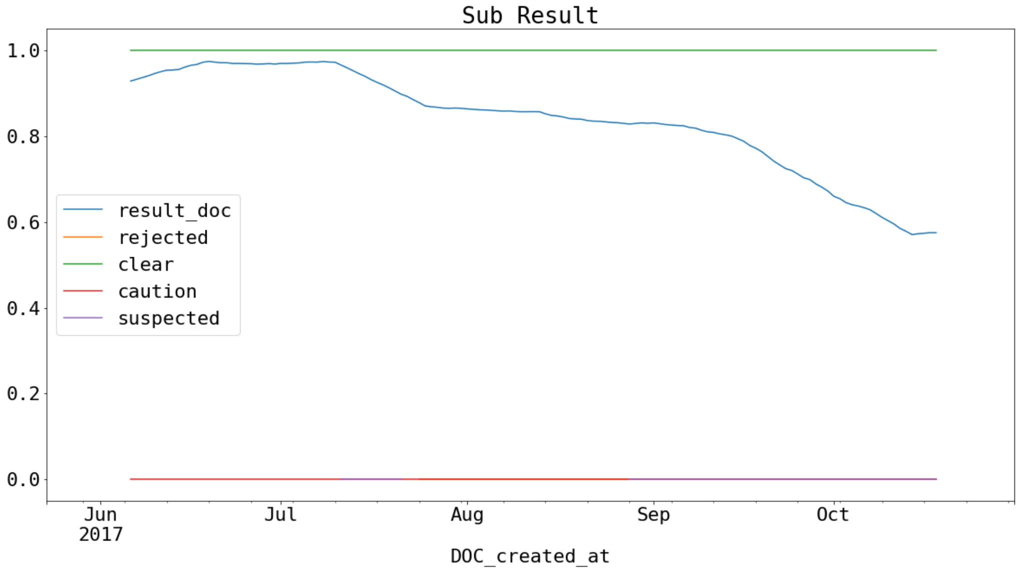

Figure 6

2 TASK 2

2.1 SQL QUERY EXAMINATION

```
WITH processed_users
    AS ( SELECT LEFT (u.phone_country , 2 ) AS short_phone_country , u.id
       FROM users u)
```

The WITH clause allows us to initiate a sub-query and assign it a temporary table alias name to use later on within the main SQL query.

Therefore, the first block of the query creates a temporary table called *processed users*. The sub-query preformed in this block selects two columns (*phone_country* and *id*) from the *users* table. The *phone_country* column is modified to only retrieve the first two characters from the data.

```
SELECT t.user_id ,
       t.merchant_country ,
       Sum (t.amount / fx.rate / Power ( 10 , cd.exponent)) AS amount

FROM transactions t
    JOIN fx_rates fx
     ON ( fx.ccy = t.currency
       AND fx.base_ccy = 'EUR' )
    JOIN currency_details cd
     ON cd.currency = t.currency --cd.currency should be cd.ccy
    JOIN processed_users pu
     ON pu.id = t.user_id

WHERE t.source = 'GAIA'
    AND pu.short_phone_country = t.merchant_country -- different format for
       country codes
GROUP BY t.user_id ,
    t.merchant_country
ORDER BY amount DESC ;
```

The following describes what the query means and what it should retrieve in theory. However some mistakes have been made which will be covered later on.

Commands used:

- *SELECT*: is used to retrieve the columns needed
- *FROM*: is used to indicate the table from which we are retriving the data. A letter next to the table name represents its abbreviation for the rest of the query
- *SUM*: is an aggregate function that adds together all the values selected. In this case is used to convert currency transactions
- *AS* rename the column or a table with an *alias*
- *JOIN* is used to combine rows from different tables base on the related columns between them. The condition and parameters of the JOIN let the user retrieve the information needed and are specified with ON and AND statements
- *WHERE* filters the result by limiting the result of the SELECT statement to what specified
- *GROUP BY* groups columns that have the same value
- *ORDER BY ... DESC* order the values in the column in descending order

The second block of the query selects the user id, merchant country the sum transaction per user in Euro by ordering in decreasing order. The users selected have the following characteristics: transaction source is *GAIA*. According to the table information this indicates ATM or credit card transactions and the country of the transaction is the same as the first recorded phone country for the user.

2.2 Errors

```
JOIN currency_details cd
     ON cd.currency = t.currency
```

This *JOIN* statement will not work as the currency column in the table transaction is ccy instead of currency. According to the table information file provided the table *currency_details* has a *ccy* column. For this reason the join will not work. However, the same column in the CSV file is names *currency*. If we consider this the join will work.

```
AND pu.short_phone_country = t.merchant_country -- different format country code
```

The above condition in the WHERE clause will not be satisfied by any record because the short phone country from processed users and merchant country from transaction have different formats, 2 and 3 characters respectively.

2.3 Query to identify users whose first transaction was a successful card payment over 10 USD

```
 WITH dates as ( with usid AS (
                             SELECT
                                 t.user_id,
                                 min(t.created_date) as mindate
                             FROM transactions t
                             GROUP BY 1
                             order by 1
                             )
               select usid.user_id, usid.mindate, t.amount,t.currency
               from usid
               INNER JOIN transactions t on (t.user_id = usid.user_id AND
                  t.created_date = usid.mindate)
               WHERE
               t.state = ’COMPLETED’ AND
               t.type = ’CARD_PAYMENT’),
amounts AS (
SELECT d.user_id,
    CASE WHEN d.currency!=’USD’
        THEN f.rate * (d.amount/ Power ( 10 , cd.exponent))
        ELSE 1 * (d.amount/ Power ( 10 , cd.exponent))
    END AS amount_usd
FROM dates d
LEFT JOIN fx_rates f ON (d.currency=f.ccy and f.base_ccy = ’USD’)
JOIN currency_details cd ON cd.ccy = d.currency
)

SELECT user_id
FROM amounts
WHERE amount_usd > 10;
```

2.4 Fraudsters

In order to identify new fraudsters it is useful to analyze behaviour and characteristics patterns of existing fraudsters. The Exploratory Data Analysis (EDA) is conducted on both a demographic and transactions levels. The former analyze the feature of the user and the latter the transactions.

From EDA is stands out that the majority of fraudsters are more likely to pass the Know Your Customer (KYC) test Figure 7 and they are from Sterling Pound 8 and, on average, they tend to be younger than non fraudsters Figure 9. The majority of fraudulent transaction occurs in Sterling Pound Figure 10 and they have a higher proportion decline rate Figure11. Ultimely the majority of these transaction take place at an ATM, POI or Supermarket Figure 12.

The 5 new possible fraudsters are identify through a Random Model Classifier. The original data was imbalanced. It has been re sampled using SMOTE. The five identified users are:

- **002ad534-53c5-4320-a199-45a2b0a9265a**
- **005380b0-d940-43c4-ac9f-d142d848aa03**
- **0031da48-f009-4fde-8288-e8ec96726b0b**
- **00681dec-2e83-4123-b529-13a6cee1356f**
- **006d7f41-a2e0-49ef-b43e-64a604ee4cf5**

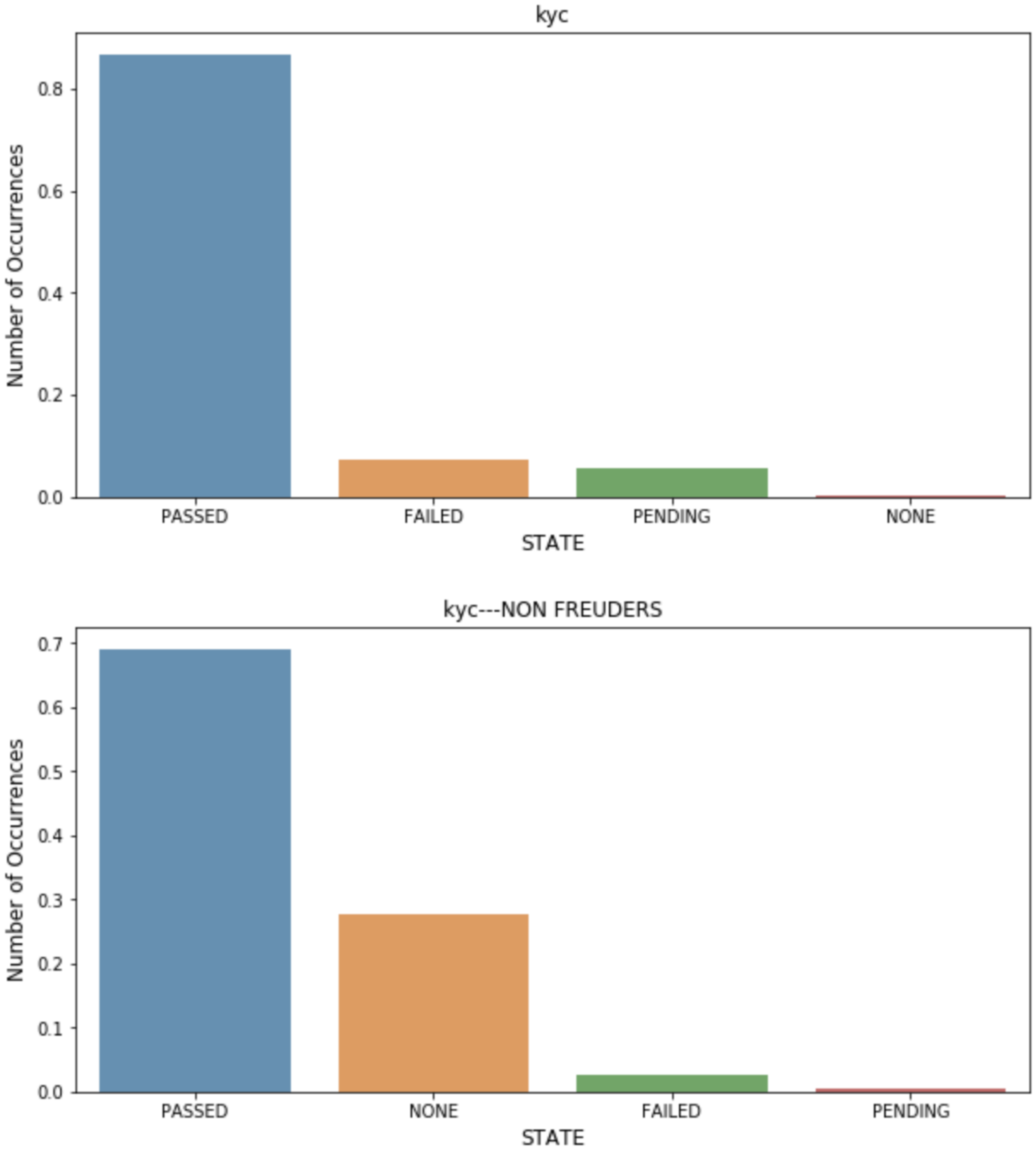
kyc
Number of Occurrences
0.0
0.2
0.4
0.6
0.8
PASSED
FAILED
PENDING
NONE
STATE
kyc---NON FREUDERS
Number of Occurrences
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
PASSED
NONE
FAILED
PENDING
STATE

Figure 7

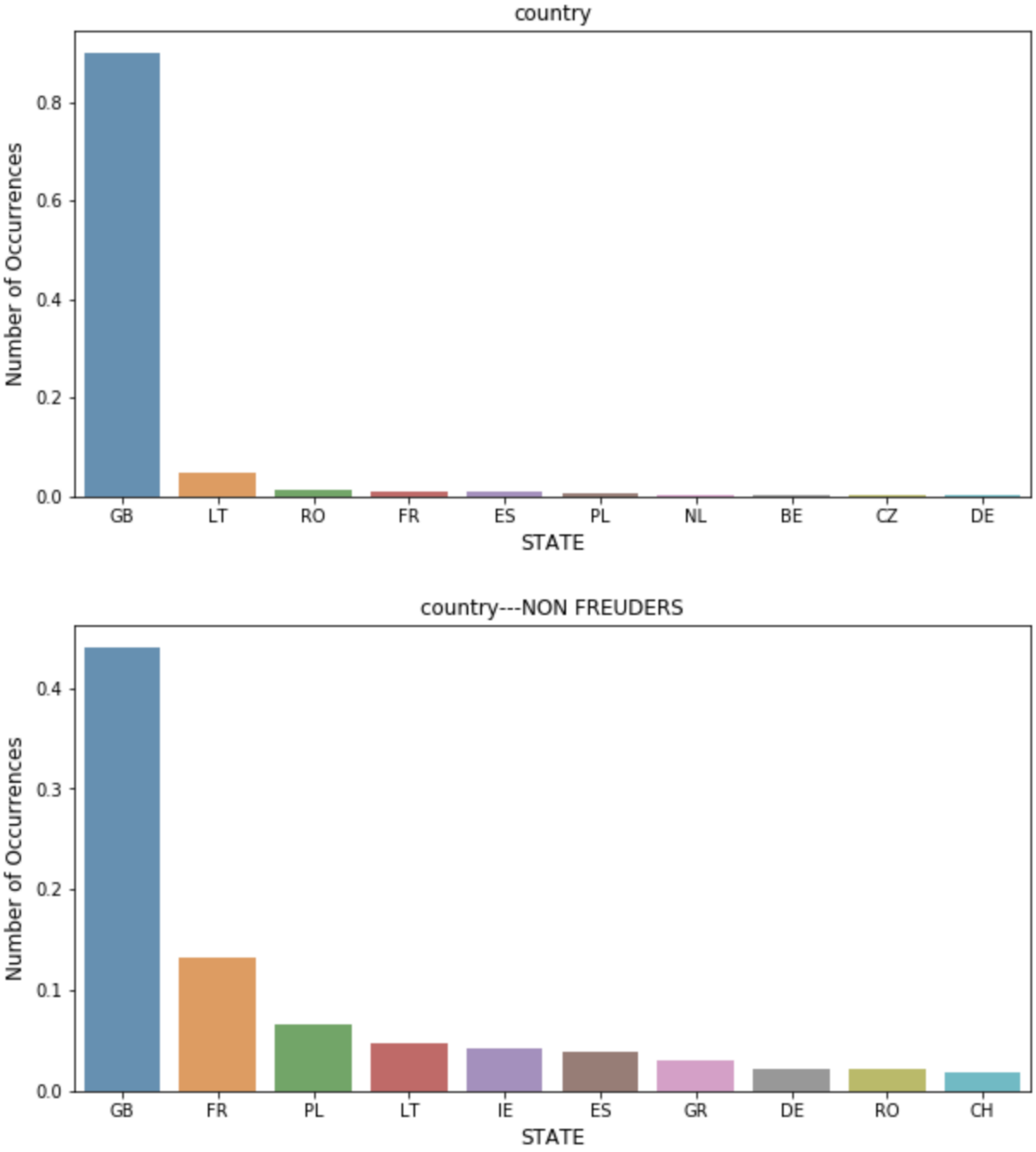
country
Number of Occurrences
0.0
0.2
0.4
0.6
0.8
GB
LT
RO
FR
ES
PL
NL
BE
CZ
DE
STATE
country---NON FREUDERS
Number of Occurrences
0.0
0.1
0.2
0.3
0.4
GB
FR
PL
LT
IE
ES
GR
DE
RO
CH
STATE

Figure 8

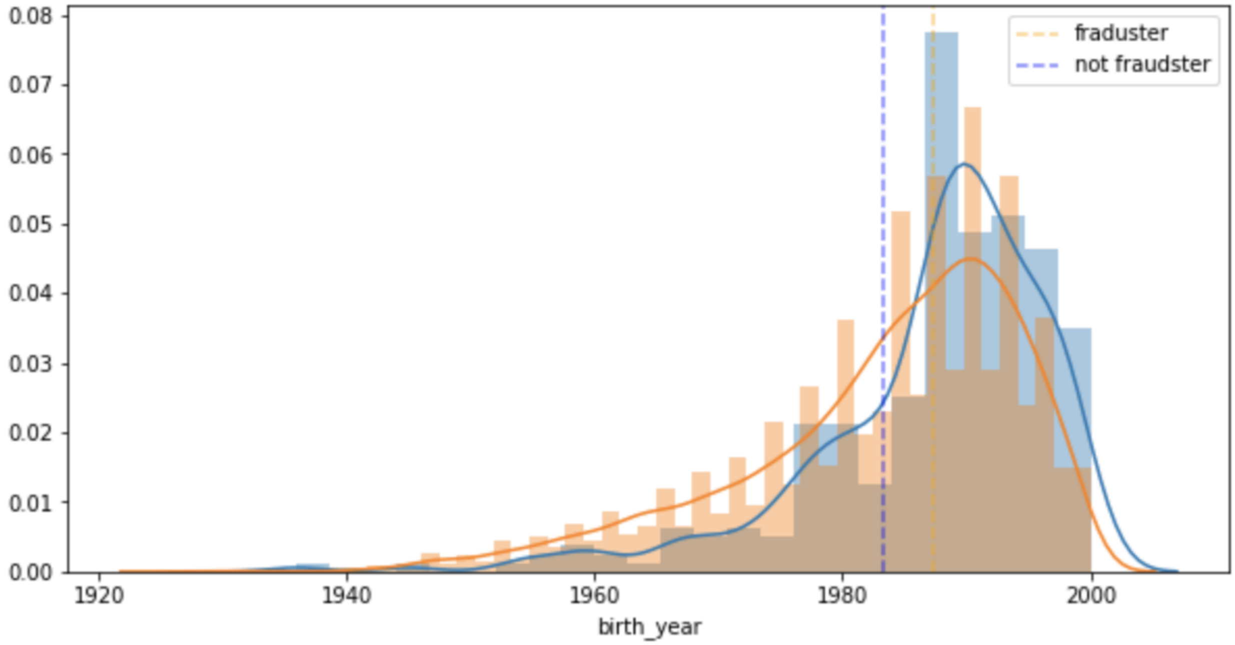
0.08
0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
1920
1940
1960
1980
2000
birth_year
fraduster
not fraudster

Figure 9

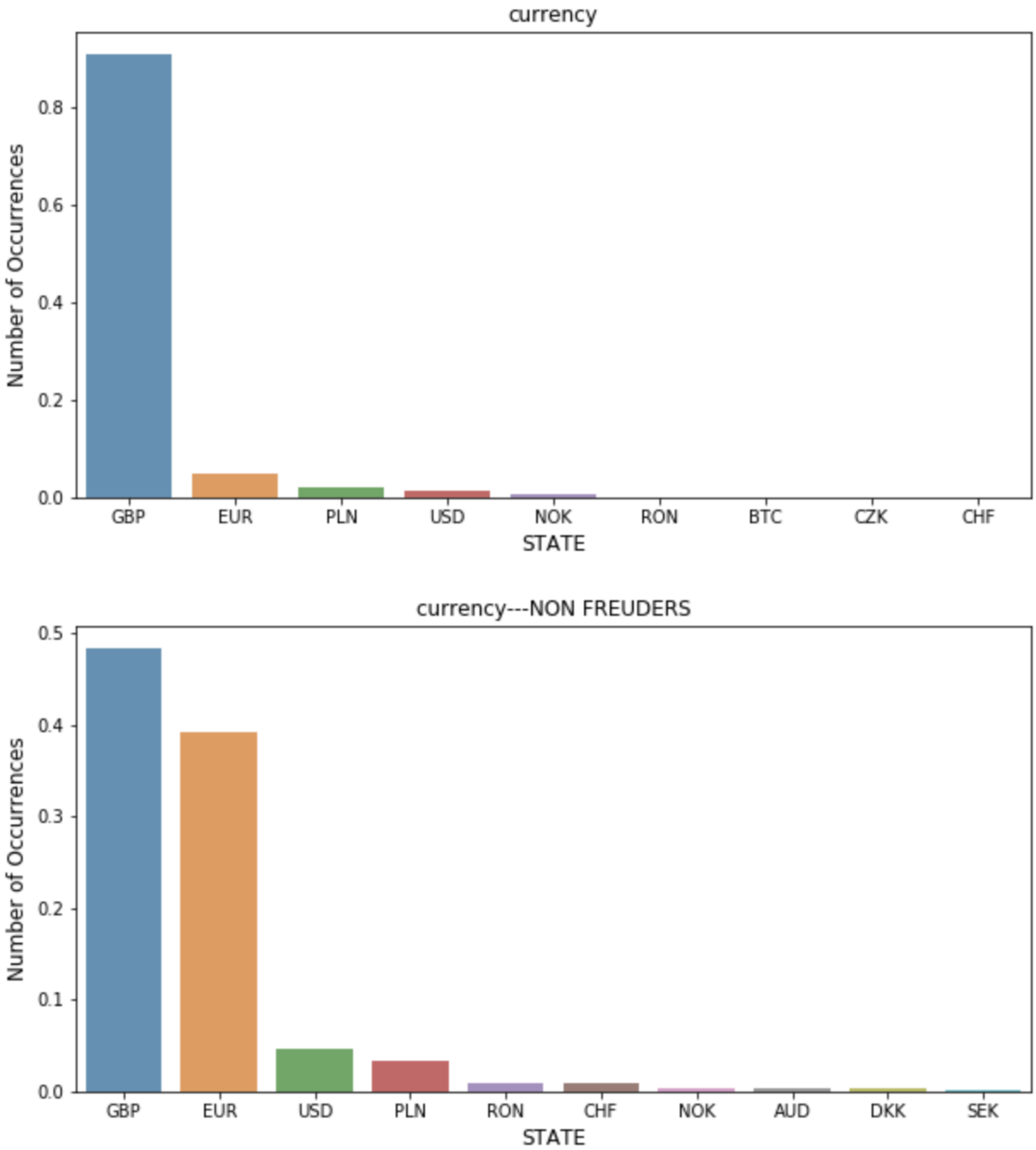
currency
Number of Occurrences
0.0
0.2
0.4
0.6
0.8
GBP
EUR
PLN
USD
NOK
RON
BTC
CZK
CHF
STATE
currency---NON FREUDERS
Number of Occurrences
0.0
0.1
0.2
0.3
0.4
0.5
GBP
EUR
USD
PLN
RON
CHF
NOK
AUD
DKK
SEK
STATE

Figure 10

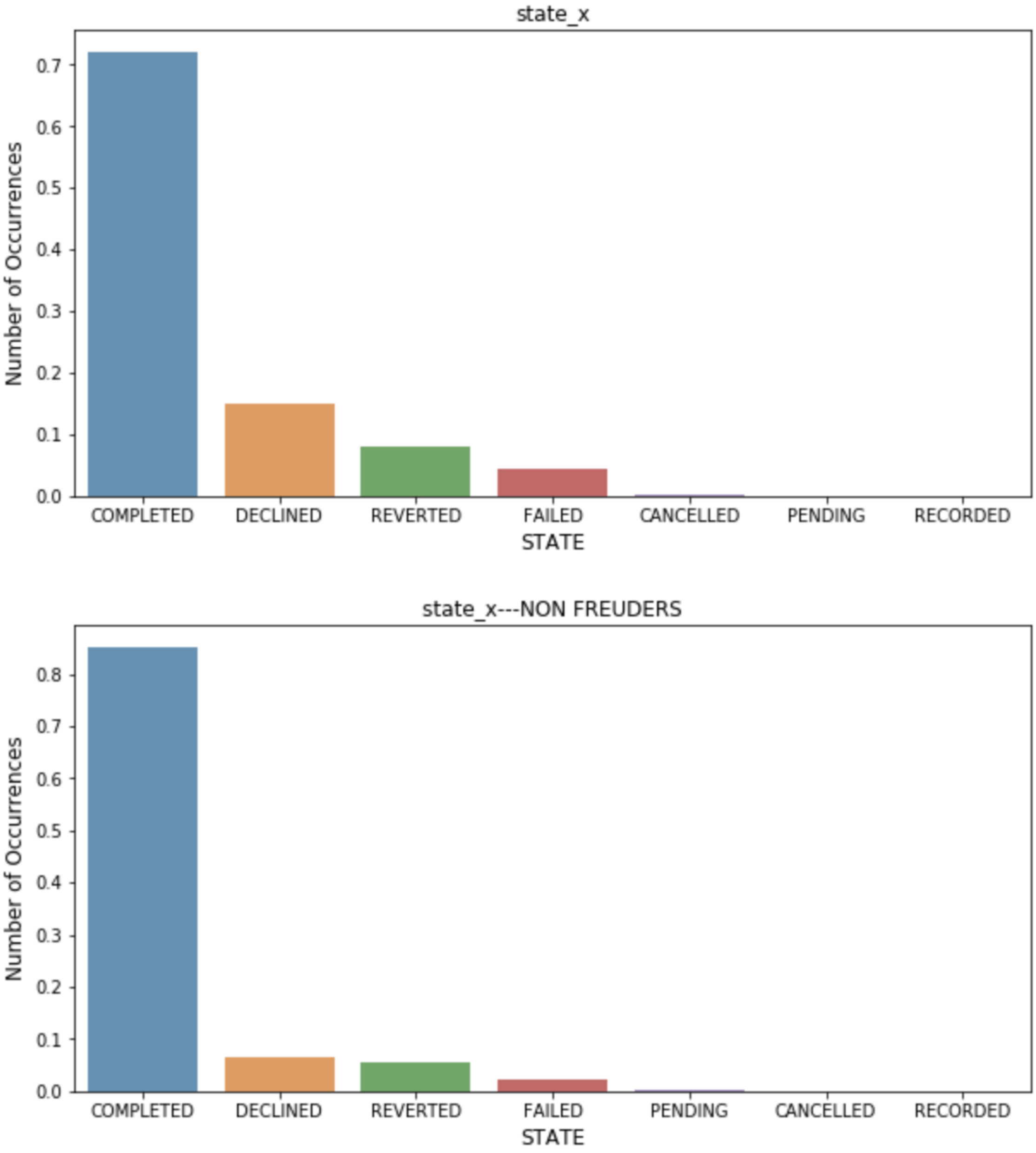
state_x
Number of Occurrences
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
COMPLETED
DECLINED
REVERTED
FAILED
CANCELLED
PENDING
RECORDED
STATE
state_x---NON FREUDERS
Number of Occurrences
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
COMPLETED
DECLINED
REVERTED
FAILED
PENDING
CANCELLED
RECORDED
STATE

Figure 11

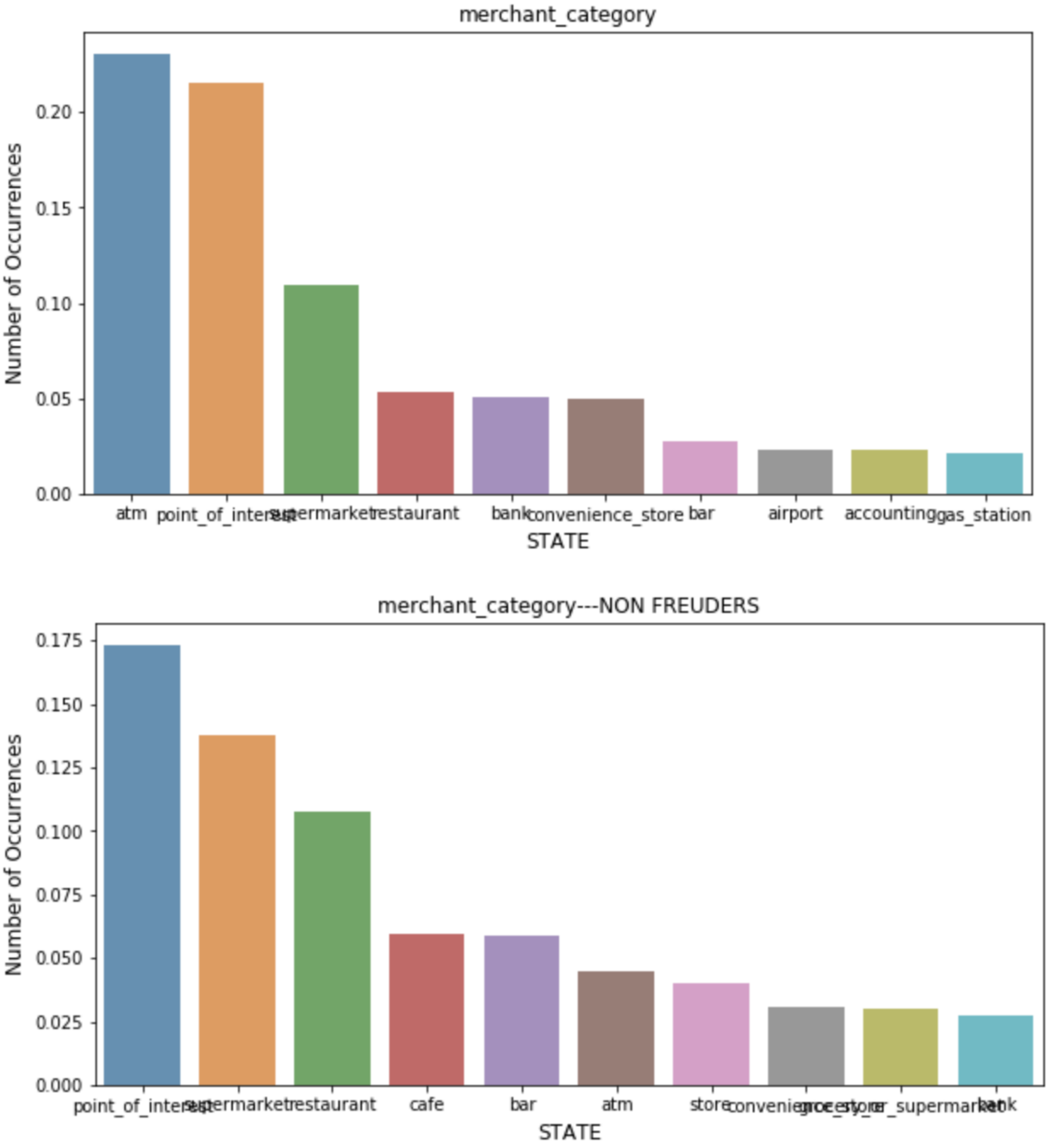
merchant_category
Number of Occurrences
0.00
0.05
0.10
0.15
0.20
atm
point_of_interest
supermarket
restaurant
bank
convenience_store
bar
airport
accounting
gas_station
STATE
merchant_category---NON FREUDERS
Number of Occurrences
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
point_of_interest
supermarket
restaurant
cafe
bar
atm
store
convenience_store
grocery_or_supermarket
bank
STATE

Figure 12