Task 1:Working with unclean data

• *For reference, Git Repository for the hometask is available at the following link :* https://github.com/abdul0214/TransferwiseDataScience

• *In order to see the repository, kindly send me an e-mail for making it public or send me GitHub username so that I add you the relevant person in it.*

• *Task1 has the corresponding Jupyter Notebook which contains detailed information along with detailed comments and figures.*

• *Kindly follow the Jupyter notebook along with this report as they contain code explanations and visualizations and valuable comments.*

Working with unclean data

The data seems to be records of money transfer transactions in the time period from **2016-01-01** to **2017-03-12**.

We have the following features available:

```
user_id_hashed
profile_type
user_create_date
user_language
age_years_bucket
user_country_code
transfer_submit_time
deposit_receive_time
transfer_amount_gbp
payment_status
payment_reference_classification
source_currency_code
target_currency_code
sum_of_this_user
deposit_receive_time_check
transfer_amount_gbp_check
submit_receive_Diff
```

With the following initial feature data types:

```
user_id_hashed                       object
profile_type                         object
user_create_date                     object
user_language                        object
age_years_bucket                     object
user_country_code                    int64
transfer_submit_time                 object
deposit_receive_time                 object
transfer_amount_gbp                  object
payment_status                       object
payment_reference_classification     object
source_currency_code                 int64
target_currency_code                 int64
dtype: object
```

However, we convert the feature types to more relevant datatypes and we have the following corrected data types:

```
user_id_hashed                                object
profile_type                                category
user_create_date                      datetime64[ns]
user_language                               category
age_years_bucket                            category
user_country_code                           category
transfer_submit_time                  datetime64[ns]
deposit_receive_time                  datetime64[ns]
transfer_amount_gbp                          float64
payment_status                              category
payment_reference_classification            category
source_currency_code                        category
target_currency_code                        category
dtype: object
```

Most frequent user of the service has the id '**ddbac55d04**' and has done **152** transfers in total. The user with highest total transfer amount is '**d71d16e6b5'** and has a total transfer sum of 18849552.0 GBP.

Top 10 Users By Total Transfer Sum:

Out[24]:

	user_id_hashed	sum_of_this_user
22371	d71d16e6b5	18849552.0
20501	3ca94ac42b	13432664.0
4500	c98ade9edb	10340256.0
19227	4b2f28326e	9896016.0
38317	3d7cc6fe80	6529974.0
23137	610b87481a	4366814.0
38363	e065dca6e7	4313022.0
11490	3ee46219a3	3994339.0
67202	9701e297dc	3583548.0
50963	37900a9e60	3568898.0

Missing Values

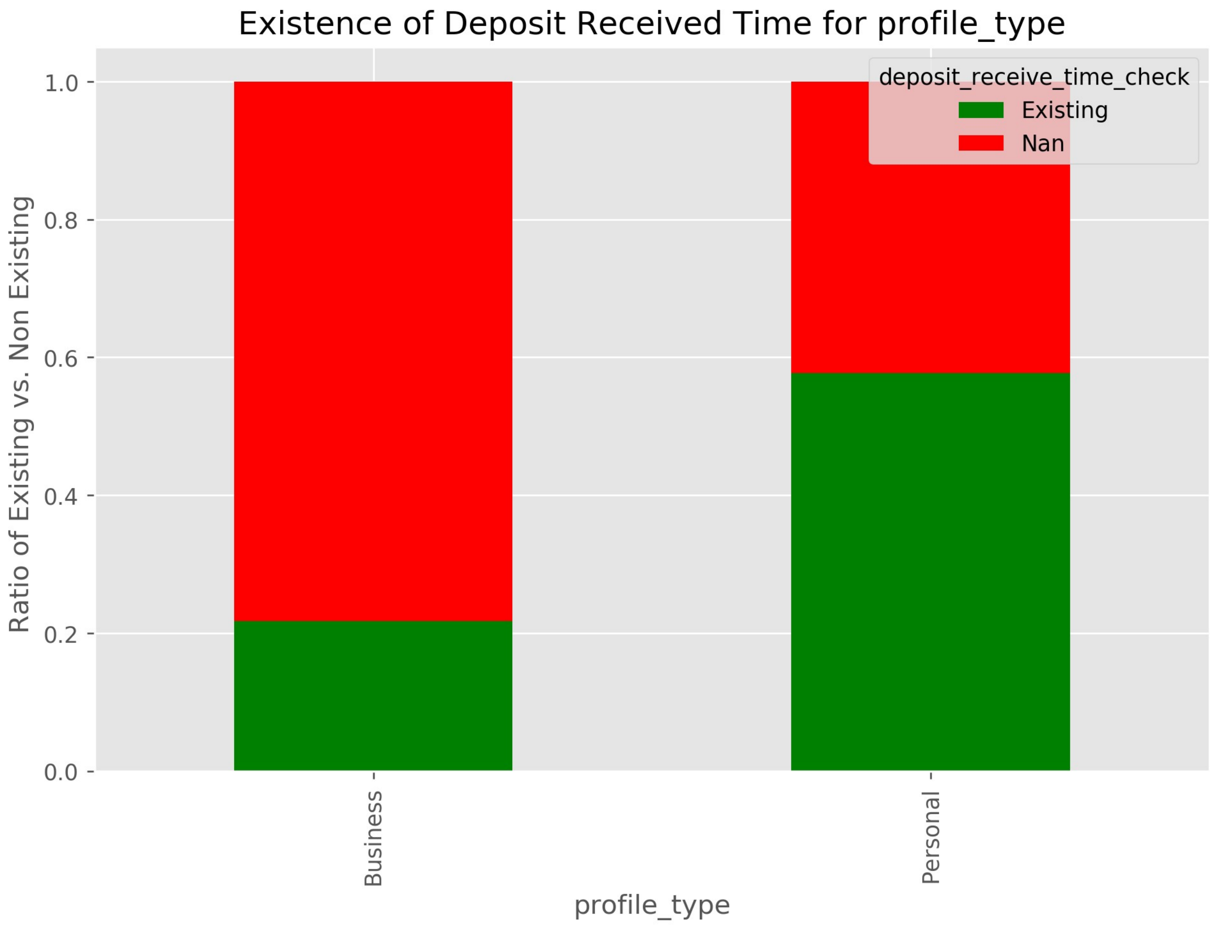
Most of the data was is in string format and hence needed to be converted to an appropriate format for further exploration.

```
In [7]: dataset.isnull().sum()

Out[7]: user_id_hashed                          0
        profile_type                            0
        user_create_date                        0
        user_language                           0
        age_years_bucket                        0
        user_country_code                       0
        transfer_submit_time                    0
        deposit_receive_time                30979
        transfer_amount_gbp                  4239
        payment_status                          0
        payment_reference_classification        0
        source_currency_code                    0
        target_currency_code                    0
        dtype: int64
```

As can be seen from above table, columns of '**deposit_receive_time**' and '**transfer_amount_gbp**' contain missing values. All other columns do not contain missing values.

Missing Deposit Received Time values



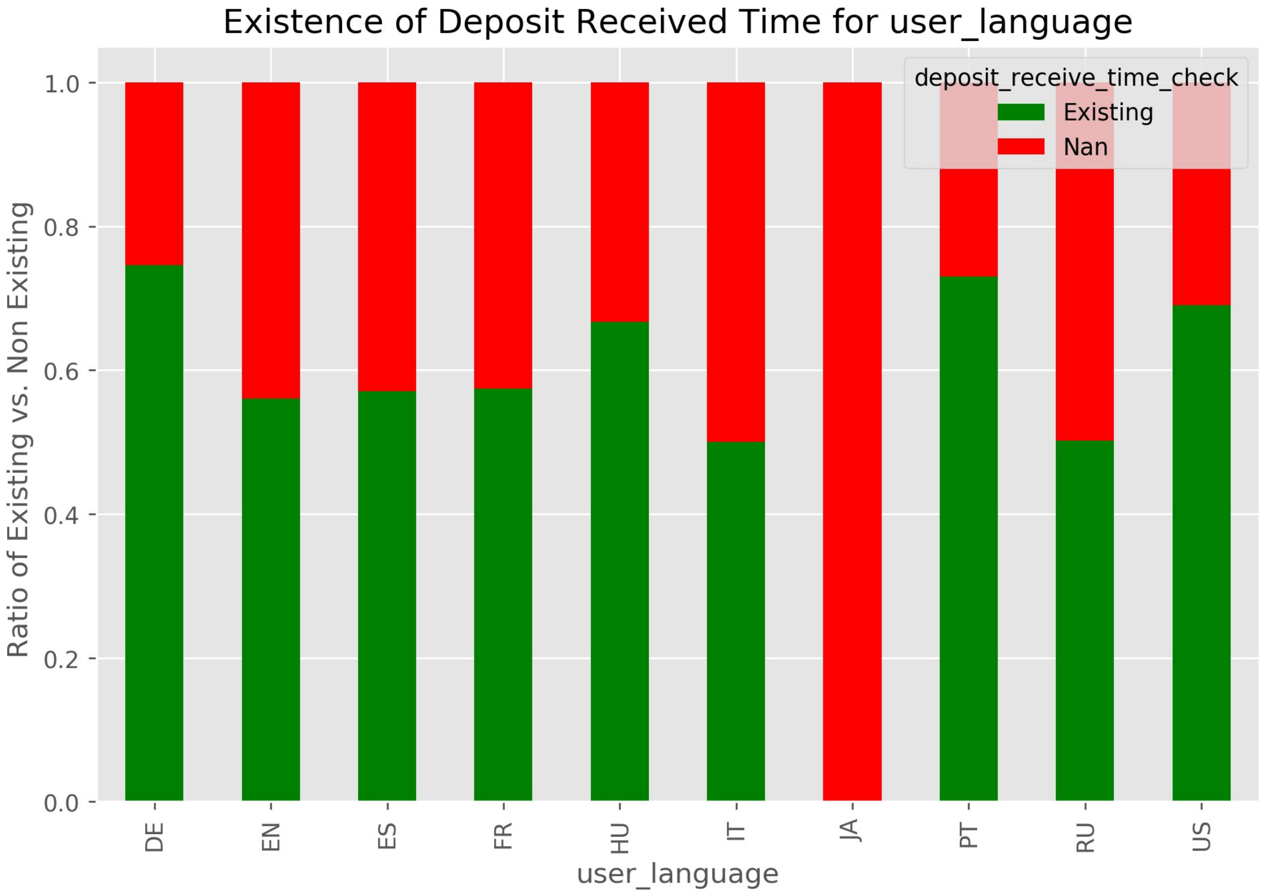
Records with the Profile Type category of ‘Business’ contain a high ratio of missing deposit received time values. Around 80% of 'Business' profile type records are missing deposit received time values.

However, most of these have Payment Status as 'Cancelled' as is shown below:

```
dataset[(dataset.profile_type == 'Business') &\
        (dataset.deposit_receive_time_check=="Nan")].describe(include=['category'])
```

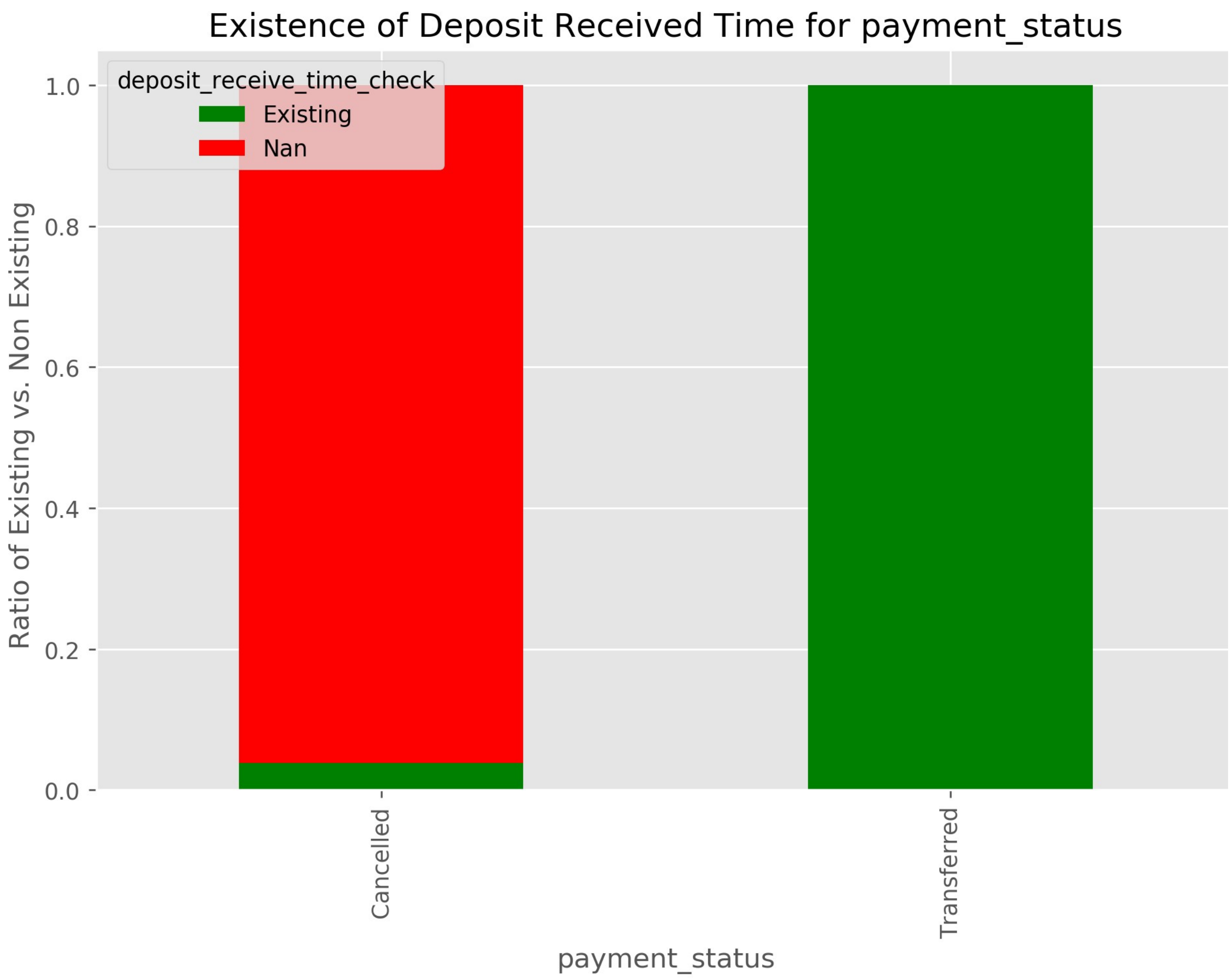
	profile_type	user_language	age_years_bucket	user_country_code	payment_status	payment_reference_classification	source_currency_code
count	852	852	852	852	852	852	852
unique	1	7	5	31	1	16	14
top	Business	EN	3. 26-34	134	Cancelled	blank	3
freq	852	794	371	494	852	534	314

Records with the User Language category of 'JA' are entirely missing deposit received time values. This is not the case with any other user language category.However number of records with missing values with missing deposit received time values for 'JA' language category are only 2 out of 71k. Hence, missing values cannot be attributed to the user language being 'JA'.



Records with the source curreny code of 11 and 35 and records with target currency code of 65 and 73 are entirely missing deposit received time values. This is not the case with any source/target curreny code .Records with user country code of 7,43,47,54,75,93,117,119,121,179,198,217,225 are entirely missing deposit received time values.

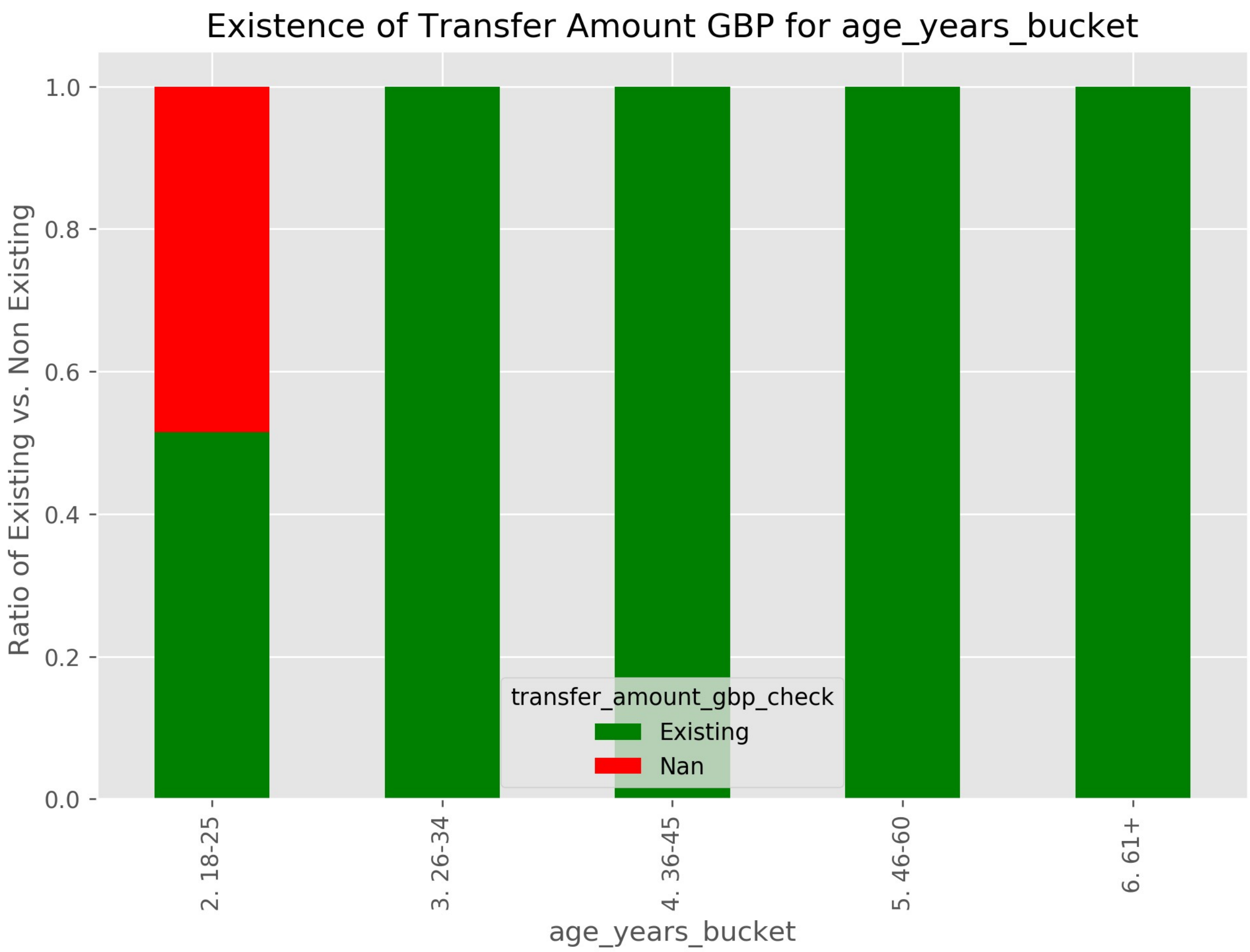
Despite the above facts almost all the records(>96%) with 'Cancelled' payment status are missing deposit received time values while none of the records with 'Transferred' payment status are missing deposit recived time values. Hence the mising of Deposit Received time values can be logically attributed to the 'Cancelled' payment status.



In addition, from the figure below we can see that records with payment status of ‘Cancelled’ make up for the (30977/30979 x 100) = 99.99% of the missing ‘Deposit Received time’ values. Hence, we can effectively conclude this as the cause of missing Deposit Missing time values. In this case, the missing values shall be given a new category. Imputation with some other values will necessarily mean introduction of false information into our data.

Missing Transfer Amount GBP values

Records with the Age Year bucket value of '2. 18-25' is the only age-years bucket value that is missing Transfer Amount values. All other age buckets are not missing Transger Amount GBP Values.



Records with the User country code of '134' is the only country code value that is missing Transfer Amount values. All other country code values are not missing Transger Amount GBP Values.

```
In [100]: dataset[(dataset.age_years_bucket == "2. 18-25") & \
                  (dataset.transfer_amount_gbp_check=="Nan")].describe(include='all')
```

Out[100]:

	user_id_hashed	profile_type	user_create_date	user_language	age_years_bucket	user_country_code	transfer_submit_time	deposit_receive_time	tra
count	4239	4239	4239	4239	4239	4239.0	4239	1918	
unique	1570	2	302	7	1	1.0	4239	1908	
top	5bc078ba04	Personal	2016-02-26 00:00:00	EN	2. 18-25	134.0	2016-10-09 15:53:02	2016-05-13 11:37:05	
freq	68	4065	73	3790	4239	4239.0	1	2	
first	NaN	NaN	2016-01-01 00:00:00	NaN	NaN	NaN	2016-01-01 17:11:05	2016-01-02 14:11:43	
last	NaN	NaN	2016-12-31 00:00:00	NaN	NaN	NaN	2017-03-11 20:54:10	2017-03-11 20:55:20	
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

All the transfer amount values are attributed to the age bucket of "2. 18-25". Further more as can be seen from the above results there is only 1 unique value of country code in this subset of age bucket and missing transfer amount values. This country code value is 134. Hence we can conclude users with country code of “134” and age bucket of "2. 18-25" are the source of these missing values.

Check for possible outliers outside 1.5 IQR

It seems like only features that can suitably be assessed for being outside the IQR are 'transfer_amount_gbp'.

```
In [32]: numeric_dataset = dataset.select_dtypes(include=['int','float'])
         numeric_dataset
```

Out[32]:

	transfer_amount_gbp
0	6056.0
1	1359.0
2	1571.0
3	8323.0
4	1571.0
...	...
72357	NaN
72358	NaN
72359	NaN
72360	NaN
72361	NaN

72362 rows × 1 columns

It seems like only features that can suitably be assessed for being outside the IQR are 'transfer_amount_gbp'.

```
In [33]: datacolumn=numeric_dataset.transfer_amount_gbp
         sorted(datacolumn)
         Q1,Q3 = np.nanpercentile(datacolumn , [25,75])
         IQR = Q3 - Q1
         lower_range = Q1 - (1.5 * IQR)
         upper_range = Q3 + (1.5 * IQR)
         print(lower_range,upper_range)

         -2184.0 4376.0

In [36]: data_outside_IQR=numeric_dataset[(numeric_dataset.transfer_amount_gbp < lower_range) \
                                         | (numeric_dataset.transfer_amount_gbp > upper_range)]

In [116]: print("\n\nPercentage of Non-Missing values outside 1.5 IQR : ",\
                round(len(data_outside_IQR)/dataset['transfer_amount_gbp'].notnull().sum(),4)*100)


          Percentage of Non-Missing values outside 1.5 IQR :  9.0
```

`Percentage of Non-Missing values outside 1.5 IQR is 9%.`

The data with values outside IQR covers most categories of other features. Hence, it is not advisable to remove the values that are outside IQR as by removing them we may lose valuable information.

In addition, just like wealth distribution in the world, where some small fraction of people have worlds most wealth, similarly, some users of Transferwise can be rich and have large transfer amounts.

Hence, it may not be reasonable to remove values outside IQR especially since we do not information on how the statistical model built on top it will react to these values.

Logically Incorrect Values

```
In [92]: dataset['submit_receive_Diff'] = dataset['deposit_receive_time'].sub\
                         (dataset['transfer_submit_time']).dt.days
         dataset[(dataset.submit_receive_Diff < 0) & (dataset.transfer_amount_gbp != np.nan)]\
                 [['user_id_hashed','transfer_submit_time','deposit_receive_time','submit_receive_Diff']]
```

Out[92]:

	user_id_hashed	transfer_submit_time	deposit_receive_time	submit_receive_Diff
8633	60132e3f0d	2016-11-03 16:29:55	2016-11-03 10:00:00	-1.0
8869	aeee61c1d5	2016-11-04 10:00:25	2016-11-04 10:00:00	-1.0
8870	19ef11b701	2016-11-04 10:02:19	2016-11-04 10:00:00	-1.0
8871	19ef11b701	2016-11-04 10:03:20	2016-11-04 10:00:00	-1.0
9005	d055fceaff	2016-11-04 14:58:47	2016-11-04 10:00:00	-1.0
...	...	...	...	...
27698	d329793a38	2017-01-16 13:38:32	2017-01-16 13:38:31	-1.0
30453	e704437807	2017-01-26 23:39:40	2017-01-26 21:42:39	-1.0
68684	f32892cf59	2016-11-07 14:40:49	2016-11-07 10:00:00	-1.0
68718	5ac3706a4d	2016-11-09 16:47:00	2016-11-09 10:00:00	-1.0
69227	f43f307b08	2016-12-07 10:31:31	2016-12-07 10:00:00	-1.0

205 rows × 4 columns

The above 205 rows logically incorrect values of 'transfer_submit_time' and 'deposit_receive_time' pair. This is because transfer_submit_time seems to be later than deposit_Receive_time which is logically incorrect.