In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [2]:

```
data = pd.read_csv(r"C:\Users\KIIT\Desktop\Highradius Internship Training\Project\dataset.c
data
```

Out[2]:

	business_code	cust_number	name_customer	clear_date	buisness_year	doc_id
0	U001	0200769623	WAL-MAR corp	2020-02-11 00:00:00	2020.0	1.930438e+09
1	U001	0200980828	BEN E	2019-08-08 00:00:00	2019.0	1.929646e+09
2	U001	0200792734	MDV/ trust	2019-12-30 00:00:00	2019.0	1.929874e+09
3	CA02	0140105686	SYSC llc	NaN	2020.0	2.960623e+09
4	U001	0200769623	WAL-MAR foundation	2019-11-25 00:00:00	2019.0	1.930148e+09
...	...	...	...	...	...	...
49995	U001	0200561861	CO corporation	NaN	2020.0	1.930797e+09
49996	U001	0200769623	WAL-MAR co	2019-09-03 00:00:00	2019.0	1.929744e+09
49997	U001	0200772595	SAFEW associates	2020-03-05 00:00:00	2020.0	1.930537e+09
49998	U001	0200726979	BJ'S llc	2019-12-12 00:00:00	2019.0	1.930199e+09
49999	U001	0200020431	DEC corp	2019-01-15 00:00:00	2019.0	1.928576e+09

50000 rows × 19 columns

In [4]:

```
data.head()
```

Out[4]:

	business_code	cust_number	name_customer	clear_date	buisness_year	doc_id	pos
0	U001	0200769623	WAL-MAR corp	2020-02-11 00:00:00	2020.0	1.930438e+09	20
1	U001	0200980828	BEN E	2019-08-08 00:00:00	2019.0	1.929646e+09	20
2	U001	0200792734	MDV/ trust	2019-12-30 00:00:00	2019.0	1.929874e+09	20
3	CA02	0140105686	SYSC llc	NaN	2020.0	2.960623e+09	20
4	U001	0200769623	WAL-MAR foundation	2019-11-25 00:00:00	2019.0	1.930148e+09	20

In [5]:

```
data.shape
```

Out[5]:

```
(50000, 19)
```

In [6]:

```
data.columns
```

Out[6]:

```
Index(['business_code', 'cust_number', 'name_customer', 'clear_date',
       'buisness_year', 'doc_id', 'posting_date', 'document_create_date',
       'document_create_date.1', 'due_in_date', 'invoice_currency',
       'document type', 'posting_id', 'area_business', 'total_open_amount',
       'baseline_create_date', 'cust_payment_terms', 'invoice_id', 'isOpe
n'],
      dtype='object')
```

In [7]:

```
rows = len(data.axes[0])
rows
```

Out[7]:

```
50000
```

In [8]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 19 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   business_code           50000 non-null  object
 1   cust_number             50000 non-null  object
 2   name_customer           50000 non-null  object
 3   clear_date              40000 non-null  object
 4   buisness_year           50000 non-null  float64
 5   doc_id                  50000 non-null  float64
 6   posting_date            50000 non-null  object
 7   document_create_date    50000 non-null  int64
 8   document_create_date.1  50000 non-null  int64
 9   due_in_date             50000 non-null  float64
 10  invoice_currency        50000 non-null  object
 11  document type           50000 non-null  object
 12  posting_id              50000 non-null  float64
 13  area_business           0 non-null      float64
 14  total_open_amount       50000 non-null  float64
 15  baseline_create_date    50000 non-null  float64
 16  cust_payment_terms      50000 non-null  object
 17  invoice_id              49994 non-null  float64
 18  isOpen                  50000 non-null  int64
dtypes: float64(8), int64(3), object(8)
memory usage: 7.2+ MB
```

In [9]:

```
data.describe()
```

Out[9]:

	buisness_year	doc_id	document_create_date	document_create_date.1	due_in_da
count	50000.000000	5.000000e+04	5.000000e+04	5.000000e+04	5.000000e+
mean	2019.305700	2.012238e+09	2.019351e+07	2.019354e+07	2.019368e+
std	0.460708	2.885235e+08	4.496041e+03	4.482134e+03	4.470614e+
min	2019.000000	1.928502e+09	2.018123e+07	2.018123e+07	2.018122e+
25%	2019.000000	1.929342e+09	2.019050e+07	2.019051e+07	2.019052e+
50%	2019.000000	1.929964e+09	2.019091e+07	2.019091e+07	2.019093e+
75%	2020.000000	1.930619e+09	2.020013e+07	2.020013e+07	2.020022e+
max	2020.000000	9.500000e+09	2.020052e+07	2.020052e+07	2.020071e+

In [10]:

```
total = data.isnull().sum().sort_values(ascending=False)
percent = (data.isnull().mean()*100).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
missing_data
```

Out[10]:

	Total	Percent
area_business	50000	100.000
clear_date	10000	20.000
invoice_id	6	0.012
business_code	0	0.000
invoice_currency	0	0.000
cust_payment_terms	0	0.000
baseline_create_date	0	0.000
total_open_amount	0	0.000
posting_id	0	0.000
document type	0	0.000
due_in_date	0	0.000
cust_number	0	0.000
document_create_date.1	0	0.000
document_create_date	0	0.000
posting_date	0	0.000
doc_id	0	0.000
buisness_year	0	0.000
name_customer	0	0.000
isOpen	0	0.000

In [11]:

```
data.duplicated()
```

Out[11]:

```
0        False
1        False
2        False
3        False
4        False
         ...
49995    False
49996    False
49997    False
49998    False
49999    False
Length: 50000, dtype: bool
```

In [12]:

```
count_uniques = data.nunique(axis=0)
count_uniques
```

Out[12]:

```
business_code                 6
cust_number                1425
name_customer              4197
clear_date                  403
buisness_year                 2
doc_id                    48839
posting_date                506
document_create_date        507
document_create_date.1      506
due_in_date                 547
invoice_currency              2
document type                 2
posting_id                    1
area_business                 0
total_open_amount         44349
baseline_create_date        506
cust_payment_terms           74
invoice_id                48833
isOpen                        2
dtype: int64
```

In [13]:

```
data.drop(columns='area_business', inplace=True)
data
```

Out[13]:

lear_date	buisness_year	doc_id	posting_date	document_create_date	document_create_dat
2020-02-11 00:00:00	2020.0	1.930438e+09	2020-01-26	20200125	202001
2019-08-08 00:00:00	2019.0	1.929646e+09	2019-07-22	20190722	201907
2019-12-30 00:00:00	2019.0	1.929874e+09	2019-09-14	20190914	201909
NaN	2020.0	2.960623e+09	2020-03-30	20200330	202003
2019-11-25 00:00:00	2019.0	1.930148e+09	2019-11-13	20191113	201911
...	...	...	...	...	
NaN	2020.0	1.930797e+09	2020-04-21	20200417	202004
2019-09-03 00:00:00	2019.0	1.929744e+09	2019-08-15	20190814	201908
2020-03-05 00:00:00	2020.0	1.930537e+09	2020-02-19	20200218	202002
2019-12-12 00:00:00	2019.0	1.930199e+09	2019-11-27	20191126	201911
2019-01-15 00:00:00	2019.0	1.928576e+09	2019-01-05	20190105	201901

In [14]:

```
data[['clear_date', 'posting_date', 'document_create_date', 'document_create_date.1', 'due_
```

Out[14]:

	clear_date	posting_date	document_create_date	document_create_date.1	due_in_date	b
0	2020-02-11 00:00:00	2020-01-26	20200125	20200126	20200210.0	
1	2019-08-08 00:00:00	2019-07-22	20190722	20190722	20190811.0	
2	2019-12-30 00:00:00	2019-09-14	20190914	20190914	20190929.0	
3	NaN	2020-03-30	20200330	20200330	20200410.0	
4	2019-11-25 00:00:00	2019-11-13	20191113	20191113	20191128.0	
...	...	...	...	...	...	
49995	NaN	2020-04-21	20200417	20200421	20200506.0	
49996	2019-09-03 00:00:00	2019-08-15	20190814	20190815	20190830.0	
49997	2020-03-05 00:00:00	2020-02-19	20200218	20200219	20200305.0	
49998	2019-12-12 00:00:00	2019-11-27	20191126	20191127	20191212.0	
49999	2019-01-15 00:00:00	2019-01-05	20190105	20190105	20190124.0	

50000 rows × 6 columns

In [15]:

```
data['clear_date'] = pd.to_datetime(data['clear_date'], format = '%Y%m%d', infer_datetime_f
data['posting_date'] = pd.to_datetime(data['posting_date'], format = '%Y%m%d', infer_dateti
data['document_create_date'] = pd.to_datetime(data['document_create_date'], format = '%Y%m%
data['document_create_date.1'] = pd.to_datetime(data['document_create_date.1'], format = '%
data['due_in_date'] = pd.to_datetime(data['due_in_date'], format = '%Y%m%d', infer_datetime
data['baseline_create_date'] = pd.to_datetime(data['baseline_create_date'], format = '%Y%m%
```

In [16]:

```
data[['clear_date', 'posting_date', 'document_create_date', 'document_create_date.1', 'due_
```

Out[16]:

	clear_date	posting_date	document_create_date	document_create_date.1	due_in_date	baseline_create
0	2020-02-11	2020-01-26	2020-01-25	2020-01-26	2020-02-10	2020-
1	2019-08-08	2019-07-22	2019-07-22	2019-07-22	2019-08-11	2019-
2	2019-12-30	2019-09-14	2019-09-14	2019-09-14	2019-09-29	2019-
3	NaT	2020-03-30	2020-03-30	2020-03-30	2020-04-10	2020-
4	2019-11-25	2019-11-13	2019-11-13	2019-11-13	2019-11-28	2019
...	...	...	...	...	...	
49995	NaT	2020-04-21	2020-04-17	2020-04-21	2020-05-06	2020-

In [18]:

```
(data['document_create_date'] > data['baseline_create_date']).sum()
```

Out[18]:

```
5710
```

In [18]:

```
(data['document_create_date'] > data['due_in_date']).sum()
```

Out[18]:

```
179
```

In [19]:

```
(data['document_create_date'] > data['posting_date']).sum()
```

Out[19]:

```
3526
```

In [20]:

```
(data['document_create_date'] > data['clear_date']).sum()
```

Out[20]:

```
1
```

In [21]:
```
(data['document_create_date.1'] > data['baseline_create_date']).sum()
```

Out[21]:
```
2225
```

In [22]:
```
(data['document_create_date.1'] > data['due_in_date']).sum()
```

Out[22]:
```
131
```

In [23]:
```
(data['document_create_date.1'] > data['posting_date']).sum()
```

Out[23]:
```
0
```

In [24]:
```
(data['document_create_date.1'] > data['clear_date']).sum()
```

Out[24]:
```
0
```

In [26]:
```
(data['baseline_create_date'] > data['due_in_date']).sum()
```

Out[26]:
```
0
```

In [27]:
```
(data['baseline_create_date'] > data['posting_date']).sum()
```

Out[27]:
```
3198
```

In [28]:
```
(data['baseline_create_date'] > data['clear_date']).sum()
```

Out[28]:
```
2
```

In [29]:

```
(data['posting_date'] > data['due_in_date']).sum()
```

Out[29]:

```
137
```

In [30]:

```
(data['posting_date'] > data['clear_date']).sum()
```

Out[30]:

```
0
```

In [22]:

```
data['isOpen'].sort_values(ascending = False)
```

Out[22]:

```
38809    1
7656     1
22811    1
34764    1
22809    1
        ..
18436    0
18437    0
18438    0
18439    0
49999    0
Name: isOpen, Length: 50000, dtype: int64
```

In [25]:

```
data['isOpen'].value_counts()
```

Out[25]:

```
0    40000
1    10000
Name: isOpen, dtype: int64
```

In [51]:

```
train, test = [x for _, x in data.groupby(data['isOpen'] > 0)]
```

In [52]:

```
train
```

Out[52]:

	business_code	cust_number	name_customer	clear_date	buisness_year	doc_id
0	U001	0200769623	WAL-MAR corp	2020-02-11	2020.0	1.930438e+09
1	U001	0200980828	BEN E	2019-08-08	2019.0	1.929646e+09
2	U001	0200792734	MDV/ trust	2019-12-30	2019.0	1.929874e+09
4	U001	0200769623	WAL-MAR foundation	2019-11-25	2019.0	1.930148e+09
5	CA02	0140106181	THE corporation	2019-12-04	2019.0	2.960581e+09
...	...	...	...	...	...	...
49994	U001	0200762301	C&S WH trust	2019-07-25	2019.0	1.929601e+09
49996	U001	0200769623	WAL-MAR co	2019-09-03	2019.0	1.929744e+09
49997	U001	0200772595	SAFEW associates	2020-03-05	2020.0	1.930537e+09
49998	U001	0200726979	BJ'S llc	2019-12-12	2019.0	1.930199e+09
49999	U001	0200020431	DEC corp	2019-01-15	2019.0	1.928576e+09

40000 rows × 18 columns

In [53]:

```
test
```

Out[53]:

	business_code	cust_number	name_customer	clear_date	buisness_year	doc_id
3	CA02	0140105686	SYSC llc	NaT	2020.0	2.960623e+09
7	U001	0200744019	TARG us	NaT	2020.0	1.930659e+09
10	U001	0200418007	AM	NaT	2020.0	1.930611e+09
14	U001	0200739534	OK systems	NaT	2020.0	1.930788e+09
15	U001	0200353024	DECA corporation	NaT	2020.0	1.930817e+09
...	...	...	...	...	...	...
49975	U001	0200769623	WAL-MAR in	NaT	2020.0	1.930625e+09
49980	U001	0200769623	WAL-MAR corporation	NaT	2020.0	1.930851e+09
49982	U001	0200148860	DOLLA co	NaT	2020.0	1.930638e+09
49992	U001	0200900909	SYSCO co	NaT	2020.0	1.930702e+09
49995	U001	0200561861	CO corporation	NaT	2020.0	1.930797e+09

10000 rows × 18 columns

In [32]:

```
train.to_csv(r"C:\Users\KIIT\Desktop\Highradius Internship Training\Project\train.csv")
```

In [36]:

```
test.to_csv(r"C:\Users\KIIT\Desktop\Highradius Internship Training\Project\test.csv")
```

In [37]:

```
pip install fast_ml
```

```
Collecting fast_ml
  Downloading fast_ml-3.68-py3-none-any.whl (42 kB)Note: you may need to res
tart the kernel to use updated packages.
Installing collected packages: fast-ml

Successfully installed fast-ml-3.68


  WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None,
status=None)) after connection broken by 'NewConnectionError('<pip._vendor.u
rllib3.connection.HTTPSConnection object at 0x0000022F9F96D700>: Failed to e
stablish a new connection: [Errno 11001] getaddrinfo failed')': /packages/2
f/c1/ff0d486b163cc98a0ed85be0bb1e50ad72a286befe78f90dc36572228a44/fast_ml-3.
68-py3-none-any.whl
```

In [54]:

```
from fast_ml.feature_selection import get_constant_features

constant_features = get_constant_features(train)
constant_features.head(10)
```

Out[54]:

	Desc	Var	Value	Perc
0	Constant	posting_id	1.0	100.000
1	Constant	isOpen	0	100.000
2	Quasi Constant	document type	RV	99.985

In [55]:

```
constant_features_list = constant_features.query("Desc=='Constant'")['Var'].to_list()
print(constant_features_list)
```

```
['posting_id', 'isOpen']
```

In [56]:

```
quasi_constant_features_list = constant_features.query("Desc=='Quasi Constant'")['Var'].to_
print(quasi_constant_features_list)
```

```
['document type']
```

In [57]:

```
train.drop(columns = constant_features_list, inplace=True)
train.shape
```

Out[57]:

```
(40000, 16)
```

In [58]:

```
train.drop(columns = quasi_constant_features_list, inplace=True)
train.shape
```

Out[58]:

```
(40000, 15)
```

In [59]:

```
from fast_ml.feature_selection import get_constant_features

constant_features = get_constant_features(test)
constant_features.head(10)
```

Out[59]:

	Desc	Var	Value	Perc
0	Constant	clear_date	NaN	100.0
1	Constant	buisness_year	2020.0	100.0
2	Constant	document type	RV	100.0
3	Constant	posting_id	1.0	100.0
4	Constant	isOpen	1	100.0

In [60]:

```
constant_features_list = constant_features.query("Desc=='Constant'")['Var'].to_list()
print(constant_features_list)
```

```
['clear_date', 'buisness_year', 'document type', 'posting_id', 'isOpen']
```

In [62]:

```
test.drop(columns = constant_features_list, inplace=True)
test.shape
```

Out[62]:

```
(10000, 13)
```

In [64]:

```
missing_valuesintrain = train.isnull().sum().sort_values(ascending=False)
missing_valuesintrain
```

Out[64]:

```
invoice_id                6
business_code             0
cust_number               0
name_customer             0
clear_date                0
buisness_year             0
doc_id                    0
posting_date              0
document_create_date      0
document_create_date.1    0
due_in_date               0
invoice_currency          0
total_open_amount         0
baseline_create_date      0
cust_payment_terms        0
dtype: int64
```

In [66]:

```
train['invoice_id'].fillna(train['invoice_id'].median(), inplace=True)
```

In [65]:

```
missing_valuesintest = test.isnull().sum().sort_values(ascending=False)
missing_valuesintest
```

Out[65]:

```
business_code             0
cust_number               0
name_customer             0
doc_id                    0
posting_date              0
document_create_date      0
document_create_date.1    0
due_in_date               0
invoice_currency          0
total_open_amount         0
baseline_create_date      0
cust_payment_terms        0
invoice_id                0
dtype: int64
```

In [ ]:

```

```

In [ ]:

```

```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: