

E_Slot_Assign

November 10, 2018

0.1 CASE STUDY: LATE DELIVERY ROOT CAUSE

READING THE DATA The below code reads the data from excel file using python package pandas

```
In [169]: import pandas as pd
```

```
data=pd.read_excel(r"D:\files\5th Sem\MS4110-Introduction to Data Analytics\assignment1.xlsx")
```

```
In [170]: data.head()
```

```
Out[170]:
```

	order_id	country	shipping_method	units_per_order	facility
0	E00000001	UNITED KINGDOM	Ground	1	OXFORD
1	E00000002	FRANCE	Ground	1	ANTWERP
2	E00000003	FRANCE	Ground	1	ANTWERP
3	E00000004	FRANCE	Ground	1	ANTWERP
4	E00000005	UNITED KINGDOM	Next Day	1	OXFORD

	product_category	on_sale	transit_days	datetime_ordered
0	ACCESSORIES	Y	2	2016-07-03 03:07:29
1	JACKETS & VESTS	N	3	2016-07-03 00:08:43
2	TOPS	Y	3	2016-07-03 00:36:00
3	JACKETS & VESTS	Y	5	2016-07-03 00:47:45
4	JACKETS & VESTS	Y	1	2016-07-03 03:52:13

	datetime_sourced	datetime_product_ready	datetime_planned
0	2016-07-03 04:09:49	2016-07-06 00:59:42	2016-07-08
1	2016-07-03 02:16:24	2016-07-03 07:17:04	2016-07-06
2	2016-07-03 02:16:18	2016-07-03 06:16:57	2016-07-06
3	2016-07-03 02:16:18	2016-07-03 06:16:55	2016-07-07
4	2016-07-03 07:56:33	2016-07-05 08:53:19	2016-07-06

	deadline_source	deadline_make	deadline_deliver	delivered_to_plan
0	1612.0	38	3	PASS
1	531.0	8	3	PASS
2	504.0	8	3	PASS
3	492.0	8	5	PASS
4	1567.0	6	1	PASS

```
In [171]: data.describe(include='all')
```

#we can see the number of unique categories in the categorical data

```
Out[171]:
```

	order_id	country	shipping_method	units_per_order	facility	\
count	79999	79999	79999	79999.000000	79999	
unique	79999	5	4	NaN	6	
top	E00005764	GERMANY	Ground	NaN	AUGSBURG	
freq	1	46259	55494	NaN	19080	
first	NaN	NaN	NaN	NaN	NaN	
last	NaN	NaN	NaN	NaN	NaN	
mean	NaN	NaN	NaN	1.040013	NaN	
std	NaN	NaN	NaN	0.466923	NaN	
min	NaN	NaN	NaN	1.000000	NaN	
25%	NaN	NaN	NaN	1.000000	NaN	
50%	NaN	NaN	NaN	1.000000	NaN	
75%	NaN	NaN	NaN	1.000000	NaN	
max	NaN	NaN	NaN	50.000000	NaN	

	product_category	on_sale	transit_days	datetime_ordered	\
count	79999	79999	79999.000000	79999	
unique	3	2	NaN	78681	
top	JACKETS & VESTS	Y	NaN	2016-08-20 17:03:18	
freq	48523	41168	NaN	8	
first	NaN	NaN	NaN	2016-07-03 00:08:43	
last	NaN	NaN	NaN	2016-10-21 17:55:30	
mean	NaN	NaN	2.338067	NaN	
std	NaN	NaN	0.992345	NaN	
min	NaN	NaN	1.000000	NaN	
25%	NaN	NaN	2.000000	NaN	
50%	NaN	NaN	2.000000	NaN	
75%	NaN	NaN	3.000000	NaN	
max	NaN	NaN	8.000000	NaN	

	datetime_sourced	datetime_product_ready	datetime_planned	\
count	79999	79999	79999	
unique	72611	77461	127	
top	2016-09-16 08:39:53	2016-07-11 11:38:10	2016-09-30 00:00:00	
freq	6	4	1539	
first	2016-07-03 02:16:18	2016-07-03 06:16:55	2016-07-05 00:00:00	
last	2016-11-10 00:33:50	2016-11-10 19:30:18	2016-11-17 00:00:00	
mean	NaN	NaN	NaN	
std	NaN	NaN	NaN	
min	NaN	NaN	NaN	
25%	NaN	NaN	NaN	
50%	NaN	NaN	NaN	
75%	NaN	NaN	NaN	
max	NaN	NaN	NaN	

	deadline_source	deadline_make	deadline_deliver	delivered_to_plan
count	79464.000000	79999.000000	79999.000000	79999
unique	NaN	NaN	NaN	2
top	NaN	NaN	NaN	PASS
freq	NaN	NaN	NaN	70978
first	NaN	NaN	NaN	NaN
last	NaN	NaN	NaN	NaN
mean	1349.708975	41.173127	3.450756	NaN
std	1104.229891	25.686936	1.617486	NaN
min	1.000000	1.000000	1.000000	NaN
25%	484.000000	23.000000	2.000000	NaN
50%	1115.000000	33.000000	3.000000	NaN
75%	1600.250000	55.000000	5.000000	NaN
max	6392.000000	168.000000	13.000000	NaN

```
In [172]: for i in data:
           print(i,data[i].dtype,sep='\t\t\t',end='\n')
           #confirming the data type of the date time variables as datetime64[ns] which can
           # be specially operated to find time intervals
```

order_id	object
country	object
shipping_method	object
units_per_order	int64
facility	object
product_category	object
on_sale	object
transit_days	int64
datetime_ordered	datetime64[ns]
datetime_sourced	datetime64[ns]
datetime_product_ready	datetime64[ns]
datetime_planned	datetime64[ns]
deadline_source	float64
deadline_make	int64
deadline_deliver	int64
delivered_to_plan	object

CREATING NEW COLUMNS THAT RECORD THE TIME INTERVAL BETWEEN DATES

We create new columns to record the time intervals between the carious dates and the time ordered as this is easier to work with and captures the necessary information from this data

```
In [173]: #cell pending deletion;was used to confirm the units of the three columns
```

```
from datetime import timedelta
# for i,j in enumerate((data.datetime_sourced-data.datetime_ordered)):
#     print(j.seconds/60,data.deadline_source[i])
```

```

print(((data.datetime_sourced-data.datetime_ordered).map(lambda x: x.seconds/(60))<d
      (((data.datetime_product_ready-data.datetime_ordered).map(lambda x: int(x.secon
      (((data.datetime_product_ready-data.datetime_ordered).map(lambda x: int(x.days,
print(((data.datetime_sourced-data.datetime_ordered).map(lambda x: x.seconds/(60))>d
      (((data.datetime_product_ready-data.datetime_ordered).map(lambda x: int(x.secon
      (((data.datetime_product_ready-data.datetime_ordered).map(lambda x: int(x.days,

```

```

76275 68090 71902
3189 9736 2760

```

In [174]: *#converting the given dates to time intervals which ccan be worked with. check the l
function for conversion to seconds minutes or days*

```

data['time_sourced']=(data.datetime_sourced-data.datetime_ordered).map(lambda x:int(x.seconds/(60))<d
data['time_make']=(data.datetime_product_ready-data.datetime_ordered).map(lambda x: int(x.seconds/(60))>d
data['time_ready']=(data.datetime_product_ready-data.datetime_ordered).map(lambda x: int(x.seconds/(60))>d

```

IMPUTING NULL VALUES

In [175]: *#checking for null values in columns*

```

for i in data:
    print(i,data[i].isnull().values.any(),sep='\t\t\t')

```

order_id	False
country	False
shipping_method	False
units_per_order	False
facility	False
product_category	False
on_sale	False
transit_days	False
datetime_ordered	False
datetime_sourced	False
datetime_product_ready	False
datetime_planned	False
deadline_source	True
deadline_make	False
deadline_deliver	False
delivered_to_plan	False
time_sourced	False
time_make	False
time_ready	False

In [176]: *#surveying the null values. Below the dataset we can see 535 null values*

```

data[data.isnull().values]

```

```

Out[176]:
    order_id    country shipping_method  units_per_order  facility \
25    E00000026    FRANCE      Ground              1    ANTWERP
34    E00000035    FRANCE      Ground              1    ANTWERP
37    E00000038    FRANCE      Ground              1    ANTWERP
39    E00000040    FRANCE      Ground              1    ANTWERP
40    E00000041    FRANCE      Ground              1    ANTWERP
41    E00000042    FRANCE      Ground              1    ANTWERP
42    E00000043    FRANCE      Ground              1    ANTWERP
43    E00000044    FRANCE      Ground              1    ANTWERP
44    E00000045    FRANCE      Ground              1    ANTWERP
45    E00000046    FRANCE      Ground              1    ANTWERP
47    E00000048    FRANCE      Ground              1    ANTWERP
48    E00000049    FRANCE      Ground              1    ANTWERP
49    E00000050    FRANCE      Ground              1    ANTWERP
52    E00000053    FRANCE      Ground              1    ANTWERP
53    E00000054    FRANCE      Ground              1    ANTWERP
54    E00000055    FRANCE      Ground              1    ANTWERP
55    E00000056    FRANCE      Ground              1    ANTWERP
58    E00000059    FRANCE      Ground              1    ANTWERP
60    E00000061    FRANCE      Ground              1    ANTWERP
65    E00000066    FRANCE      Ground              1    ANTWERP
70    E00000071    FRANCE      Ground              1    ANTWERP
77    E00000078    FRANCE      Ground              1    ANTWERP
78    E00000079    FRANCE      Ground              1    ANTWERP
83    E00000084    FRANCE      Ground              1    ANTWERP
97    E00000098    FRANCE      Ground              1    ANTWERP
104   E00000105    FRANCE      Ground              1    ANTWERP
111   E00000112    GERMANY      Ground              1    EINDHOVEN
115   E00000116    FRANCE      Ground              1    ANTWERP
116   E00000117    FRANCE      Ground              1    ANTWERP
118   E00000119    UNITED KINGDOM  Next Day            1    OXFORD
...   ...         ...         ...         ...         ...
75449 E00075531    UNITED KINGDOM  Ground              1    MANCHESTER
75464 E00075546    UNITED KINGDOM  Ground              1    MANCHESTER
75465 E00075547    UNITED KINGDOM  Ground              1    MANCHESTER
75471 E00075553    UNITED KINGDOM  Ground              1    OXFORD
75476 E00075558    UNITED KINGDOM  Ground              1    OXFORD
75479 E00075561    UNITED KINGDOM  Ground              1    OXFORD
75484 E00075566    UNITED KINGDOM  Ground              1    OXFORD
75488 E00075570    UNITED KINGDOM  Ground              1    MANCHESTER
75498 E00075580    UNITED KINGDOM  Ground              1    MANCHESTER
75500 E00075582    UNITED KINGDOM  Ground              1    OXFORD
75502 E00075584    UNITED KINGDOM  Ground              1    OXFORD
75503 E00075585    UNITED KINGDOM  Ground              1    MANCHESTER
75507 E00075589    UNITED KINGDOM  Ground              1    OXFORD
75511 E00075593    UNITED KINGDOM  Ground              2    OXFORD
75513 E00075595    UNITED KINGDOM  Ground              1    MANCHESTER
75526 E00075608    UNITED KINGDOM  Ground              1    OXFORD

```

75529	E00075611	UNITED KINGDOM	Ground	1	OXFORD
75544	E00075626	UNITED KINGDOM	Ground	1	OXFORD
75545	E00075627	UNITED KINGDOM	Ground	1	OXFORD
75550	E00075632	UNITED KINGDOM	Ground	1	OXFORD
75552	E00075634	UNITED KINGDOM	Ground	1	OXFORD
75554	E00075636	UNITED KINGDOM	Ground	1	MANCHESTER
75556	E00075638	UNITED KINGDOM	Ground	2	MANCHESTER
75557	E00075639	UNITED KINGDOM	Ground	1	OXFORD
75572	E00075654	UNITED KINGDOM	Ground	1	OXFORD
75583	E00075665	UNITED KINGDOM	Ground	1	OXFORD
75606	E00075688	UNITED KINGDOM	Ground	2	MANCHESTER
75620	E00075702	UNITED KINGDOM	Ground	1	OXFORD
75678	E00075760	UNITED KINGDOM	Ground	1	MANCHESTER
79998	E00080081	UNITED KINGDOM	Ground	1	OXFORD

	product_category	on_sale	transit_days	datetime_ordered	\
25	TOPS	Y	4	2016-07-03 04:52:12	
34	JACKETS & VESTS	N	3	2016-07-03 05:42:00	
37	JACKETS & VESTS	Y	3	2016-07-03 06:01:54	
39	TOPS	N	3	2016-07-03 06:38:27	
40	TOPS	Y	3	2016-07-03 06:39:40	
41	TOPS	N	4	2016-07-03 07:16:03	
42	JACKETS & VESTS	Y	3	2016-07-03 07:33:24	
43	JACKETS & VESTS	Y	3	2016-07-03 07:49:19	
44	JACKETS & VESTS	N	4	2016-07-03 07:55:02	
45	TOPS	Y	4	2016-07-03 07:59:58	
47	ACCESSORIES	Y	2	2016-07-03 08:16:49	
48	JACKETS & VESTS	N	3	2016-07-03 08:22:15	
49	JACKETS & VESTS	N	3	2016-07-03 08:25:17	
52	JACKETS & VESTS	Y	4	2016-07-03 08:51:53	
53	JACKETS & VESTS	Y	3	2016-07-03 08:52:33	
54	TOPS	N	4	2016-07-03 08:55:24	
55	JACKETS & VESTS	Y	3	2016-07-03 08:56:31	
58	TOPS	Y	3	2016-07-03 09:41:02	
60	TOPS	Y	3	2016-07-03 09:42:17	
65	JACKETS & VESTS	Y	4	2016-07-03 10:18:41	
70	TOPS	Y	3	2016-07-03 10:36:41	
77	TOPS	Y	3	2016-07-03 11:27:04	
78	JACKETS & VESTS	Y	3	2016-07-03 11:29:29	
83	ACCESSORIES	Y	3	2016-07-03 11:36:10	
97	TOPS	Y	2	2016-07-03 12:28:30	
104	JACKETS & VESTS	N	5	2016-07-03 13:03:22	
111	ACCESSORIES	N	2	2016-07-03 15:19:55	
115	JACKETS & VESTS	Y	3	2016-07-03 13:30:37	
116	TOPS	Y	3	2016-07-03 13:34:35	
118	JACKETS & VESTS	N	1	2016-07-03 16:38:00	
...	
75449	TOPS	Y	1	2016-10-14 17:59:01	

75464	TOPS	Y	1	2016-10-14	18:08:15
75465	ACCESSORIES	N	3	2016-10-14	18:10:40
75471	JACKETS & VESTS	Y	2	2016-10-14	18:18:18
75476	JACKETS & VESTS	Y	1	2016-10-14	18:26:35
75479	JACKETS & VESTS	N	2	2016-10-14	18:27:03
75484	JACKETS & VESTS	Y	1	2016-10-14	18:37:41
75488	TOPS	Y	2	2016-10-14	18:44:12
75498	TOPS	N	2	2016-10-14	18:57:49
75500	JACKETS & VESTS	N	2	2016-10-14	18:58:52
75502	JACKETS & VESTS	Y	1	2016-10-14	19:02:06
75503	TOPS	N	1	2016-10-14	19:03:39
75507	JACKETS & VESTS	Y	1	2016-10-14	19:14:17
75511	JACKETS & VESTS	Y	1	2016-10-14	19:17:36
75513	TOPS	Y	3	2016-10-14	19:19:50
75526	JACKETS & VESTS	Y	1	2016-10-14	19:34:48
75529	JACKETS & VESTS	Y	3	2016-10-14	19:36:43
75544	JACKETS & VESTS	N	1	2016-10-14	19:50:58
75545	JACKETS & VESTS	Y	1	2016-10-14	19:53:14
75550	JACKETS & VESTS	Y	1	2016-10-14	20:00:25
75552	JACKETS & VESTS	Y	2	2016-10-14	20:01:00
75554	TOPS	Y	1	2016-10-14	20:02:16
75556	ACCESSORIES	N	1	2016-10-14	20:03:24
75557	JACKETS & VESTS	Y	2	2016-10-14	20:03:43
75572	JACKETS & VESTS	N	2	2016-10-14	20:24:07
75583	JACKETS & VESTS	Y	1	2016-10-14	20:40:16
75606	ACCESSORIES	N	1	2016-10-14	21:04:55
75620	JACKETS & VESTS	N	1	2016-10-14	21:28:55
75678	TOPS	N	2	2016-10-14	22:46:46
79998	JACKETS & VESTS	N	3	2016-10-21	12:15:57

	datetime_sourced	datetime_product_ready	datetime_planned	\
25	2016-07-03 07:17:09	2016-07-03 09:17:13	2016-07-06	
34	2016-07-03 07:16:36	2016-07-03 10:16:49	2016-07-05	
37	2016-07-03 08:16:37	2016-07-03 10:17:00	2016-07-05	
39	2016-07-03 08:17:11	2016-07-04 08:17:50	2016-07-05	
40	2016-07-03 08:16:57	2016-07-04 03:16:16	2016-07-06	
41	2016-07-03 09:16:34	2016-07-04 04:16:23	2016-07-06	
42	2016-07-03 09:17:23	2016-07-04 04:16:47	2016-07-05	
43	2016-07-03 09:17:20	2016-07-04 07:18:13	2016-07-06	
44	2016-07-03 10:16:45	2016-07-04 05:17:01	2016-07-06	
45	2016-07-03 10:16:52	2016-07-04 07:18:00	2016-07-07	
47	2016-07-03 10:17:01	2016-07-04 04:18:33	2016-07-06	
48	2016-07-03 10:16:57	2016-07-04 04:17:25	2016-07-06	
49	2016-07-03 10:16:57	2016-07-04 07:19:29	2016-07-06	
52	2016-07-03 10:16:24	2016-07-04 04:17:49	2016-07-07	
53	2016-07-03 11:16:29	2016-07-04 07:17:06	2016-07-06	
54	2016-07-03 11:16:22	2016-07-04 07:16:16	2016-07-08	
55	2016-07-03 11:16:37	2016-07-04 04:17:52	2016-07-06	

58	2016-07-04 03:16:31	2016-07-04 07:19:21	2016-07-06
60	2016-07-03 11:16:26	2016-07-04 03:17:04	2016-07-06
65	2016-07-03 12:16:29	2016-07-04 06:16:18	2016-07-09
70	2016-07-03 13:16:30	2016-07-04 06:16:27	2016-07-06
77	2016-07-03 13:16:44	2016-07-04 04:17:35	2016-07-07
78	2016-07-03 13:16:38	2016-07-04 04:17:25	2016-07-07
83	2016-07-03 13:16:50	2016-07-04 04:16:16	2016-07-07
97	2016-07-03 14:16:38	2016-07-04 08:18:23	2016-07-06
104	2016-07-03 15:16:40	2016-07-04 05:16:35	2016-07-13
111	2016-07-03 15:56:23	2016-07-06 11:58:22	2016-07-07
115	2016-07-03 15:16:17	2016-07-04 04:18:16	2016-07-07
116	2016-07-03 16:16:21	2016-07-04 04:17:31	2016-07-07
118	2016-07-03 17:41:49	2016-07-05 06:19:13	2016-07-06
...
75449	2016-10-14 19:09:18	2016-10-18 19:09:56	2016-10-19
75464	2016-10-14 19:09:22	2016-10-18 19:29:07	2016-10-19
75465	2016-10-14 19:29:24	2016-10-18 15:49:13	2016-10-21
75471	2016-10-14 19:28:20	2016-10-18 19:18:48	2016-10-20
75476	2016-10-14 19:28:36	2016-10-18 10:49:21	2016-10-19
75479	2016-10-14 19:28:32	2016-10-17 10:05:15	2016-10-20
75484	2016-10-14 19:28:38	2016-10-17 10:20:22	2016-10-19
75488	2016-10-14 19:50:33	2016-10-18 19:23:35	2016-10-20
75498	2016-10-14 20:03:00	2016-10-18 19:27:22	2016-10-20
75500	2016-10-14 19:51:31	2016-10-18 14:12:39	2016-10-20
75502	2016-10-14 19:55:39	2016-10-18 14:12:43	2016-10-19
75503	2016-10-14 20:03:22	2016-10-18 19:23:18	2016-10-19
75507	2016-10-14 20:31:47	2016-10-17 10:22:37	2016-10-19
75511	2016-10-14 20:33:45	2016-10-18 18:25:25	2016-10-19
75513	2016-10-14 20:32:29	2016-10-18 19:31:01	2016-10-21
75526	2016-10-14 20:33:56	2016-10-19 18:05:18	2016-10-19
75529	2016-10-14 20:21:51	2016-10-18 14:10:35	2016-10-21
75544	2016-10-14 21:10:36	2016-10-17 10:21:10	2016-10-21
75545	2016-10-14 21:10:32	2016-10-18 10:52:38	2016-10-19
75550	2016-10-14 21:10:31	2016-10-18 19:28:21	2016-10-19
75552	2016-10-14 20:56:40	2016-10-18 14:30:53	2016-10-20
75554	2016-10-14 21:10:21	2016-10-18 19:20:27	2016-10-19
75556	2016-10-14 20:59:27	2016-10-18 19:20:58	2016-10-19
75557	2016-10-14 21:10:37	2016-10-18 18:25:24	2016-10-20
75572	2016-10-14 21:35:47	2016-10-17 10:22:46	2016-10-20
75583	2016-10-14 21:57:33	2016-10-18 14:36:15	2016-10-19
75606	2016-10-14 21:59:30	2016-10-18 19:20:21	2016-10-19
75620	2016-10-14 22:21:58	2016-10-18 14:57:24	2016-10-19
75678	2016-10-15 00:04:43	2016-10-17 16:08:02	2016-10-19
79998	2016-10-21 13:19:49	2016-10-24 17:20:54	2016-10-31

	deadline_source	deadline_make	deadline_deliver	delivered_to_plan	\
25	NaN	3	4	PASS	
34	NaN	3	3	PASS	

37	NaN	2	3	PASS
39	NaN	2	3	PASS
40	NaN	2	3	PASS
41	NaN	25	4	PASS
42	NaN	25	3	FAIL
43	NaN	25	3	PASS
44	NaN	24	4	FAIL
45	NaN	24	4	PASS
47	NaN	24	2	PASS
48	NaN	24	3	PASS
49	NaN	24	3	PASS
52	NaN	24	4	PASS
53	NaN	23	3	PASS
54	NaN	23	4	PASS
55	NaN	23	3	PASS
58	NaN	7	3	PASS
60	NaN	23	3	PASS
65	NaN	22	4	PASS
70	NaN	21	3	PASS
77	NaN	21	3	PASS
78	NaN	21	3	PASS
83	NaN	21	3	PASS
97	NaN	20	2	PASS
104	NaN	19	5	PASS
111	NaN	83	2	FAIL
115	NaN	19	3	PASS
116	NaN	18	3	PASS
118	NaN	20	1	PASS
...
75449	NaN	143	2	FAIL
75464	NaN	143	2	PASS
75465	NaN	143	3	PASS
75471	NaN	143	3	PASS
75476	NaN	143	1	PASS
75479	NaN	143	2	PASS
75484	NaN	143	1	PASS
75488	NaN	142	3	PASS
75498	NaN	142	3	FAIL
75500	NaN	142	2	FAIL
75502	NaN	142	1	PASS
75503	NaN	142	2	PASS
75507	NaN	141	1	PASS
75511	NaN	141	1	PASS
75513	NaN	141	6	FAIL
75526	NaN	141	1	FAIL
75529	NaN	142	3	PASS
75544	NaN	141	1	FAIL
75545	NaN	141	1	PASS

75550	NaN	141	2	PASS
75552	NaN	141	2	PASS
75554	NaN	141	2	PASS
75556	NaN	141	2	PASS
75557	NaN	141	2	PASS
75572	NaN	140	2	PASS
75583	NaN	140	1	PASS
75606	NaN	140	2	FAIL
75620	NaN	140	1	PASS
75678	NaN	134	3	PASS
79998	NaN	77	3	PASS

	time_sourced	time_make	time_ready
25	144	4	0
34	94	4	0
37	134	4	0
39	98	1	1
40	97	20	0
41	120	21	0
42	103	20	0
43	88	23	0
44	141	21	0
45	136	23	0
47	120	20	0
48	114	19	0
49	111	22	0
52	84	19	0
53	143	22	0
54	140	22	0
55	140	19	0
58	1055	21	0
60	94	17	0
65	117	19	0
70	159	19	0
77	109	16	0
78	107	16	0
83	100	16	0
97	108	19	0
104	133	16	0
111	36	20	2
115	105	14	0
116	161	14	0
118	63	13	1
...
75449	70	1	4
75464	61	1	4
75465	78	21	3
75471	70	1	4

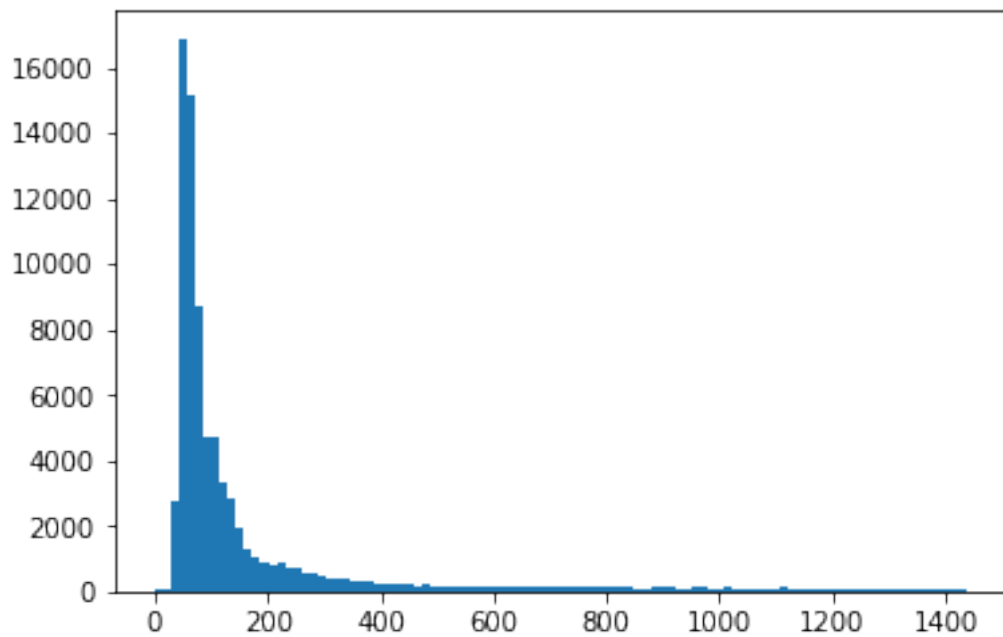
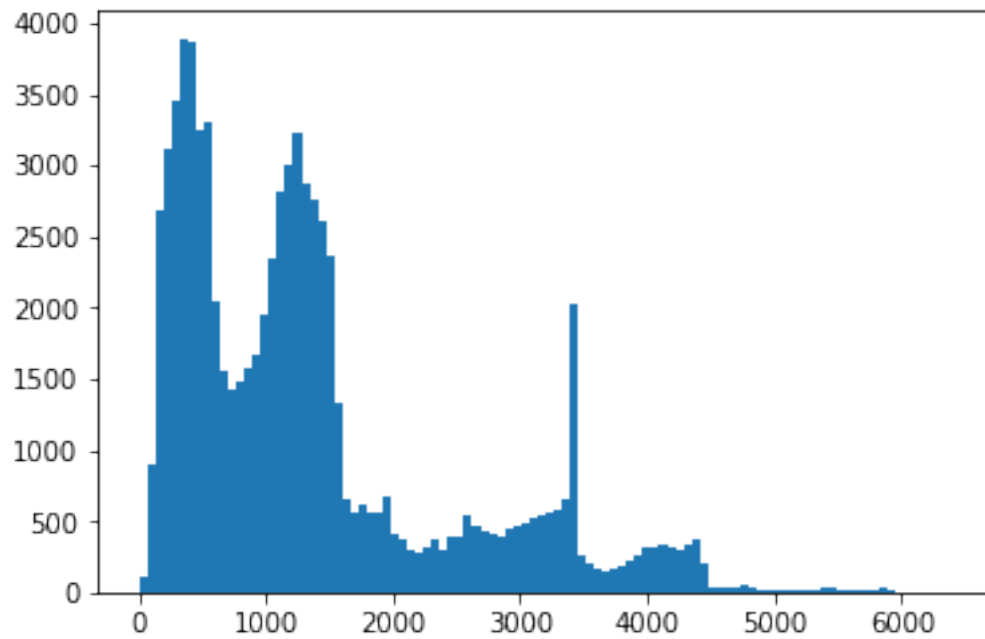
75476	62	16	3
75479	61	15	2
75484	50	15	2
75488	66	0	4
75498	65	0	4
75500	52	19	3
75502	53	19	3
75503	59	0	4
75507	77	15	2
75511	76	23	3
75513	72	0	4
75526	59	22	4
75529	45	18	3
75544	79	14	2
75545	77	14	3
75550	70	23	3
75552	55	18	3
75554	68	23	3
75556	56	23	3
75557	66	22	3
75572	71	13	2
75583	77	17	3
75606	54	22	3
75620	53	17	3
75678	77	17	2
79998	63	5	3

[535 rows x 19 columns]

In [177]: *#pending deletion. to check the plot of the missing na values. Can keep/ expand for*

```
import matplotlib.pyplot as plt
from tqdm import tqdm_notebook as tqdm
import seaborn as sns
%matplotlib inline

plt.hist(data.deadline_source.dropna(),bins=100)
plt.show()
plt.hist(data.time_sourced.dropna(),bins=100)
plt.show()
```



```
In [178]: #creating a new variable data_corr with which we can work on our dataset
          # and drop the columns irrelevant to the problem
```

```
data_corr=data;
data_corr=data_corr.drop(['order_id','delivered_to_plan','datetime_ordered','datetime_delivered'])
data_corr.head()
```

```
Out[178]:
```

		country	shipping_method	units_per_order	facility	product_category	\
0	UNITED KINGDOM		Ground	1	OXFORD	ACCESSORIES	
1	FRANCE		Ground	1	ANTWERP	JACKETS & VESTS	
2	FRANCE		Ground	1	ANTWERP	TOPS	
3	FRANCE		Ground	1	ANTWERP	JACKETS & VESTS	
4	UNITED KINGDOM		Next Day	1	OXFORD	JACKETS & VESTS	

	on_sale	transit_days	deadline_source	deadline_make	deadline_deliver	\
0	Y	2	1612.0	38	3	
1	N	3	531.0	8	3	
2	Y	3	504.0	8	3	
3	Y	5	492.0	8	5	
4	Y	1	1567.0	6	1	

	time_sourced	time_make	time_ready
0	62	21	2
1	127	7	0
2	100	5	0
3	88	5	0
4	244	5	2

```
In [264]: #creating a dictionary so that we can map the categorical values. can create the map
```

```
col=['country','shipping_method','facility','product_category','on_sale']#categorical
tmp=[];
for i,j in enumerate(col):
    a=data_corr[j].unique()
    tmp.append(dict(zip(a,list(range(0,len(a))))))
    print(tmp[i])
    data_corr[j]=data_corr[j].map(tmp[i])
data_corr.head()
```

```
{0: 0, 1: 1, 2: 2, 3: 3, 4: 4}
{0: 0, 1: 1, 2: 2, 3: 3}
{0: 0, 1: 1, 2: 2, 3: 3, 4: 4, 5: 5}
{0: 0, 1: 1, 2: 2}
{0: 0, 1: 1}
```

```
Out[264]:
```

	deadline_source	country	shipping_method	units_per_order	facility	\
0	1612.0	0	0	1	0	
1	531.0	1	0	1	1	
2	504.0	1	0	1	1	
3	492.0	1	0	1	1	
4	1567.0	0	1	1	0	