Fraud Detection Report

January 31, 2021

1 Part One

1.1 Introduction

We start by loading the data and running it through the pipelines to impute the missing values we can easily determine

[2]:

	date	orders	circumstance	drivers_available	month	weekday
0	2012-01-02	24899.0	dry	4223.0	1	0
1	2012-01-03	24774.0	dry	5450.0	1	1
3	2012-01-05	25083.0	dry	4691.0	1	3
4	2012-01-06	24676.0	dry	5308.0	1	4
5	2012-01-07	24905.0	dry	5905.0	1	5

As we can see we have 4 features:

* date: datetime
* orders: float
* circumstance: str
* drivers_available: float

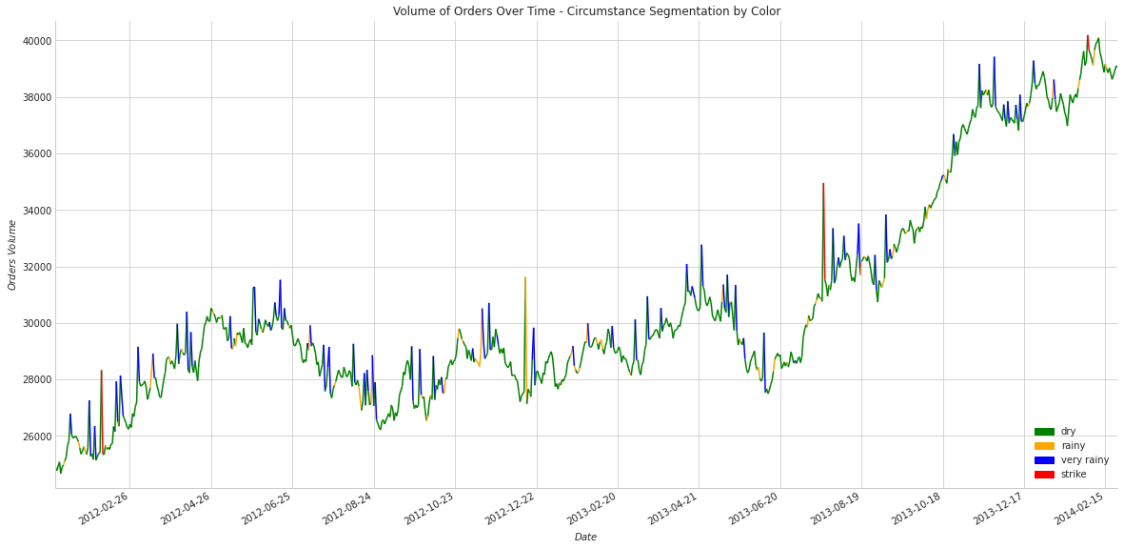
* month: int from 0 to 12
* weekday: int from 0 to 6

This dataset gives us daily data on number of orders and drivers available across 785 days between 2012-01-02 and 2014-02-24. There are 16 missing values from the 'orders' column and 8 from the 'drivers_available'. We will be able to use an ML model to impute those values

1.2 Exploratory Analysis

1.2.1 Time Series entire period graph

The graph below plots the entire time series, coloring the segments according to their corresponding circumstance



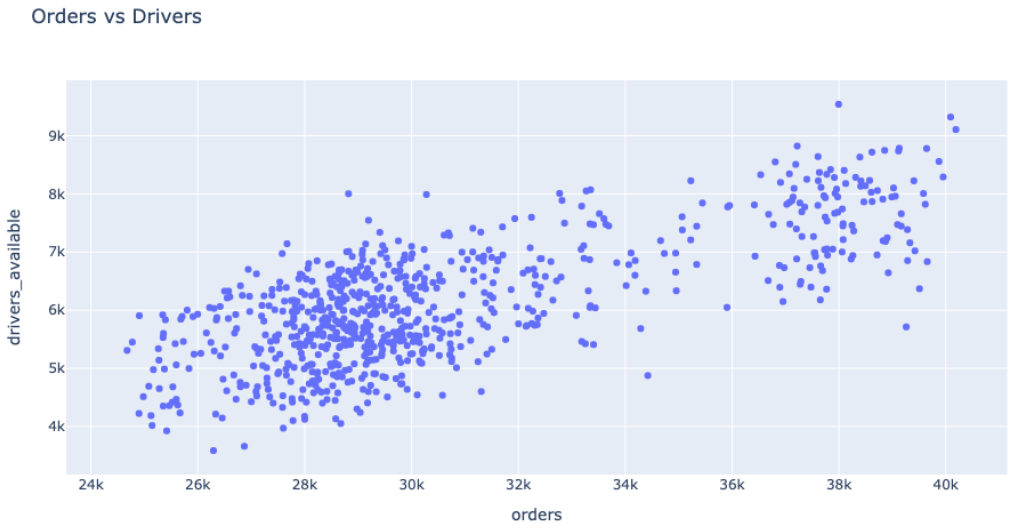
- "Very rainy" days (in blue) often come with a spike in orders
- The three "strike" days (in red) also come with a spike

but they are too few to make any kind of statistical conclusion out of it

1.2.2 Relationship between the two numerical variables (Scatter Plot)

Below is a scatter plot that hopes to visualize the correlation between 'drivers_available' and 'orders'

[4]:



The elliptical shape of the points shows a clear positive correlation between the two variables. This is not surprising as the company likely makes more drivers available to meet demand.

Although highly predictive, using this variable as a feature in a model that predicts the number of orders might not be a good idea.

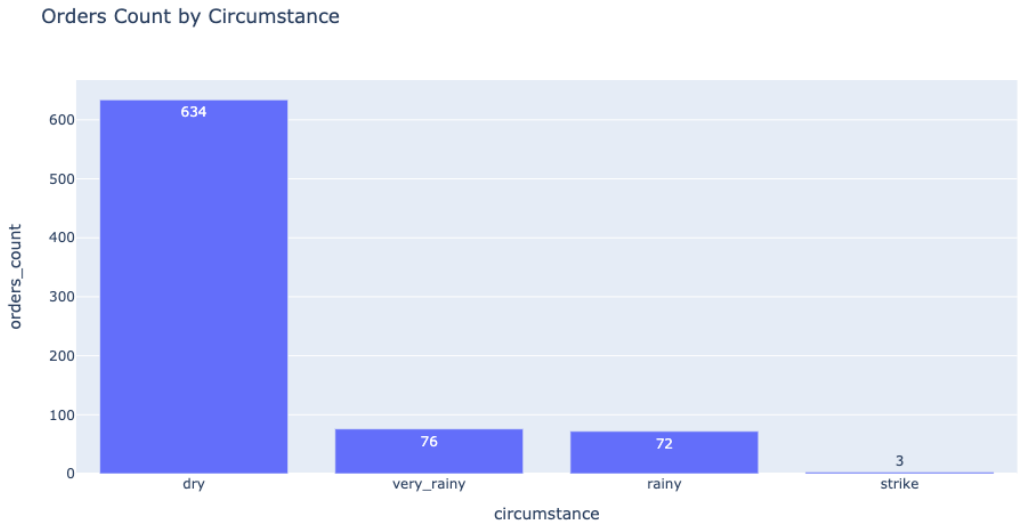
Indeed, 'drivers_available' is a number fixed by the company using the information they already have on orders, thus, nothing can be learned from it that is not already known.

```
Pearsons correlation between orders and drivers_available: 0.728
```

1.2.3 Distribution of the circumstance variable (Bar Chart)

Below is a bar chart that gives us the distribution of the categorical 'circumstance'

[6]:



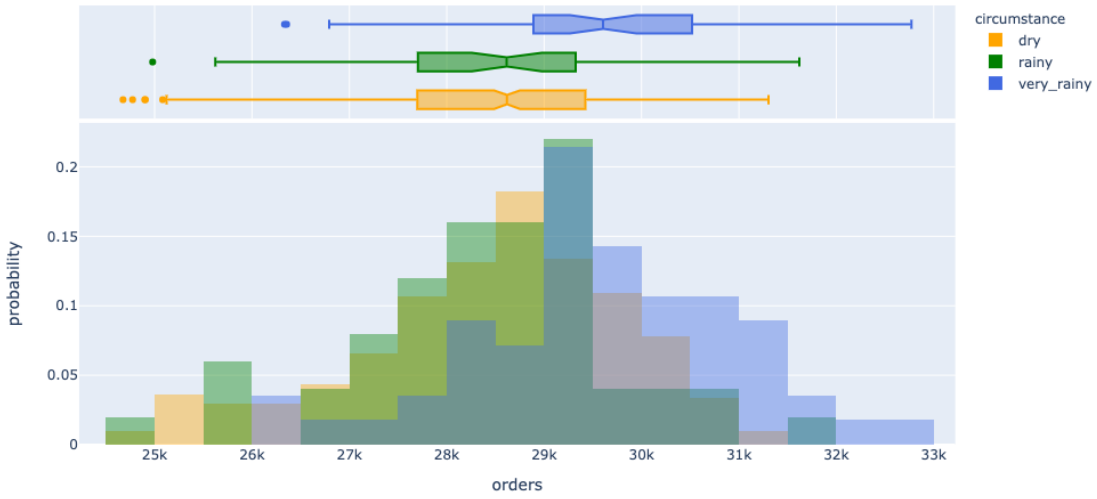
1.3 Key Trends

In order to analyze the trends in this dataset we will only look at it from the first day 2012-01-02 to 2013-06-20. The time series is more or less seasonal before that date. At that point the number of order starts skyrocketing.

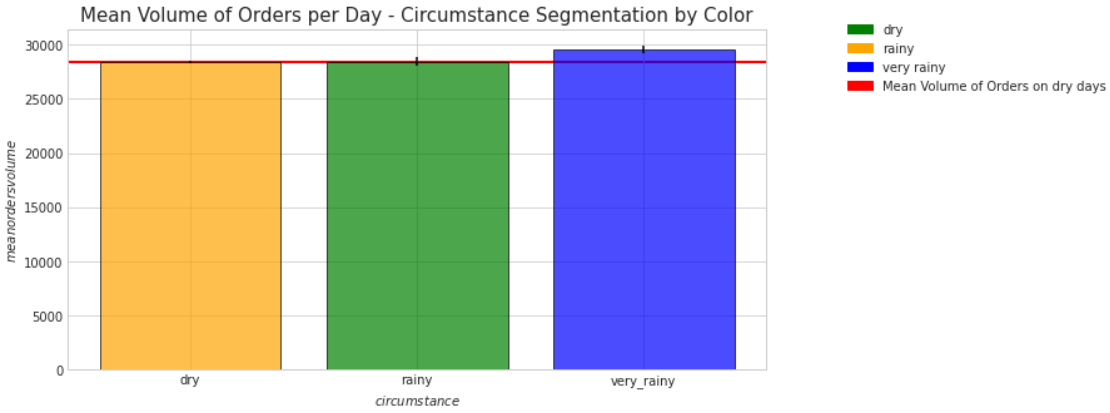
If we consider the whole time series we risk introducing bias to the the analysis. For example, if the year following 2013-06-20 is particularly dry, the data will tell us that dry days induce more orders. However, correlation does not mean causation and that spike in sales is probably completely unrelated to the overall circumstances of that year

1.3.1 Orders Distribution across Circumstance segments

[8]:



As we can see dry and rainy days are more or less consistent in terms of orders distribution. Very rainy days clearly stand out, both in the box plot and the histogram. This confirms what we saw in the first graph, the spikes do translate into an overall higher probability of having more orders on very rainy days.

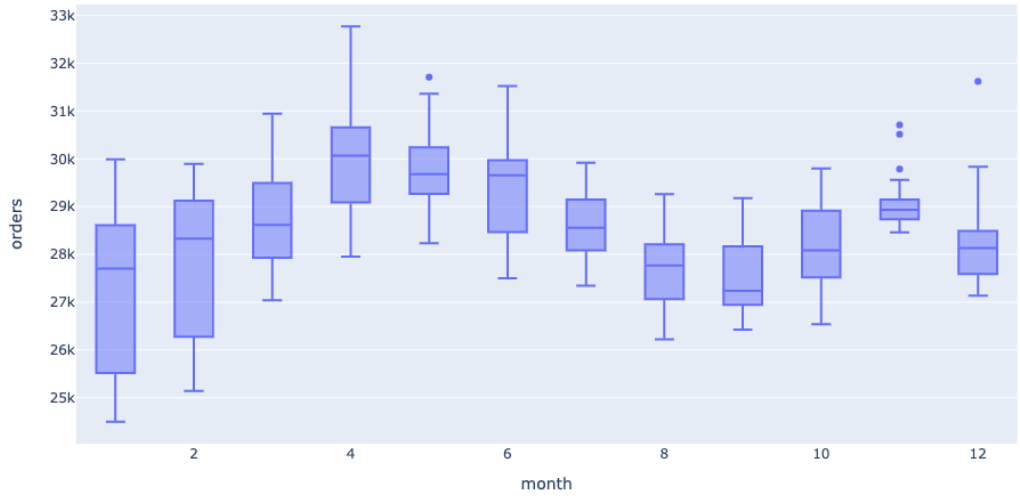


What appears to be black dots on top of those bars are actually 95% confidence intervals. Thus, the confidence interval of the mean volume of orders on very rainy days being well above the "dry days" threshold, we can safely say that the variations we see on very rainy days are note random but the fruit of an actual pattern.

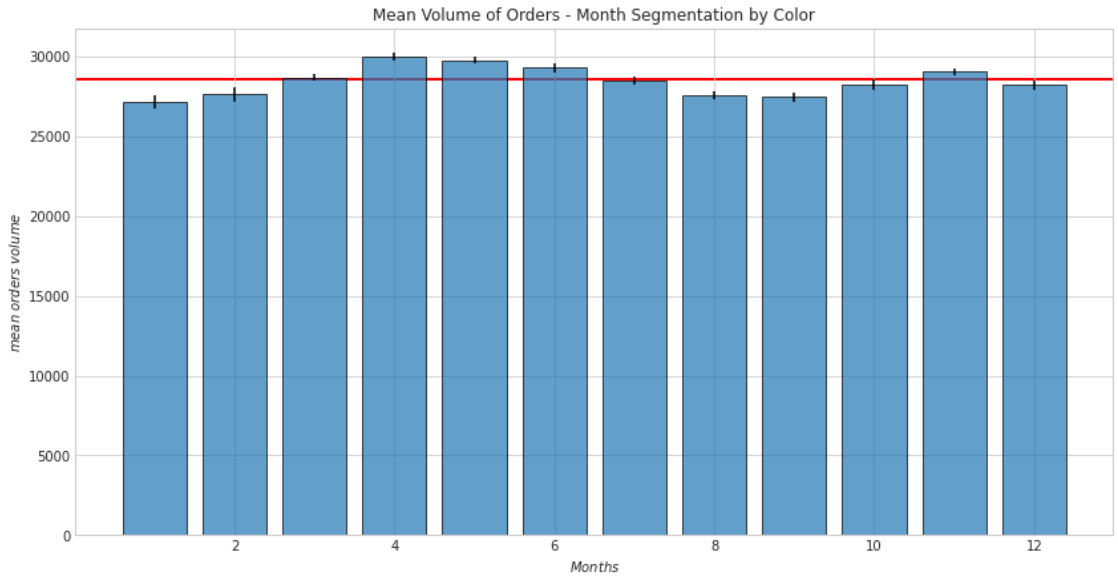
1.3.2 Looking for seasonality in the number of orders

Monthly seasonality The box plots below show a clear variation between the months.

[10]:



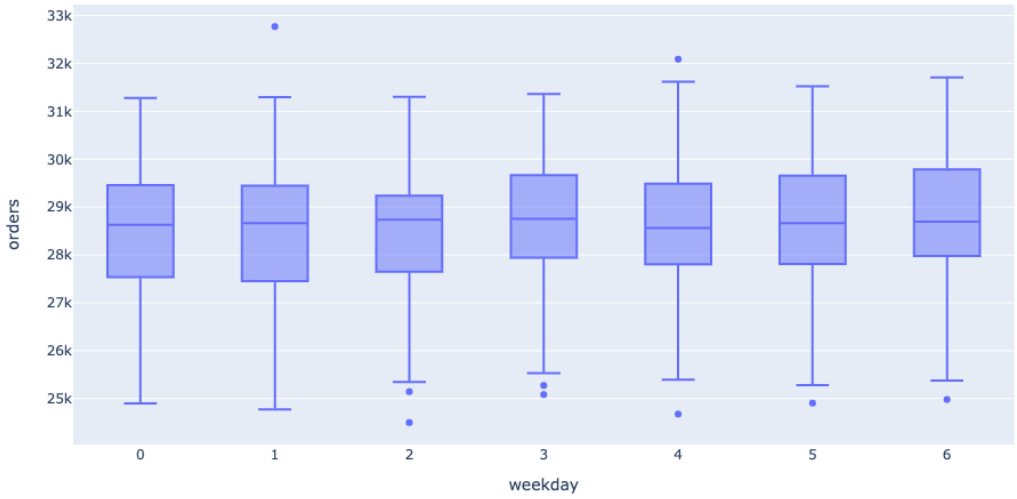
Let us verify that these variations are not random by looking at the mean monthly orders and their respective confidence interval



The red line is the global mean number of orders across the studied section of the data. When the confidence interval of a given month is below or above that line, we can say that at least 95% of the time, the mean orders volume for that month will still be on that side of the line. Thus guaranteeing that these variations are almost certainly not random but systematic.

Weekly seasonality The box plots below show no variation between weekdays.

[12]:



Weekdays seem to be presenting no valuable information to determine the order volume

1.4 Modeling

In this section we will see two main different types of modeling. Each type is better adapted to predict orders for a different purpose. Which model is eventually retained depends on the needs of the clients/stakeholders

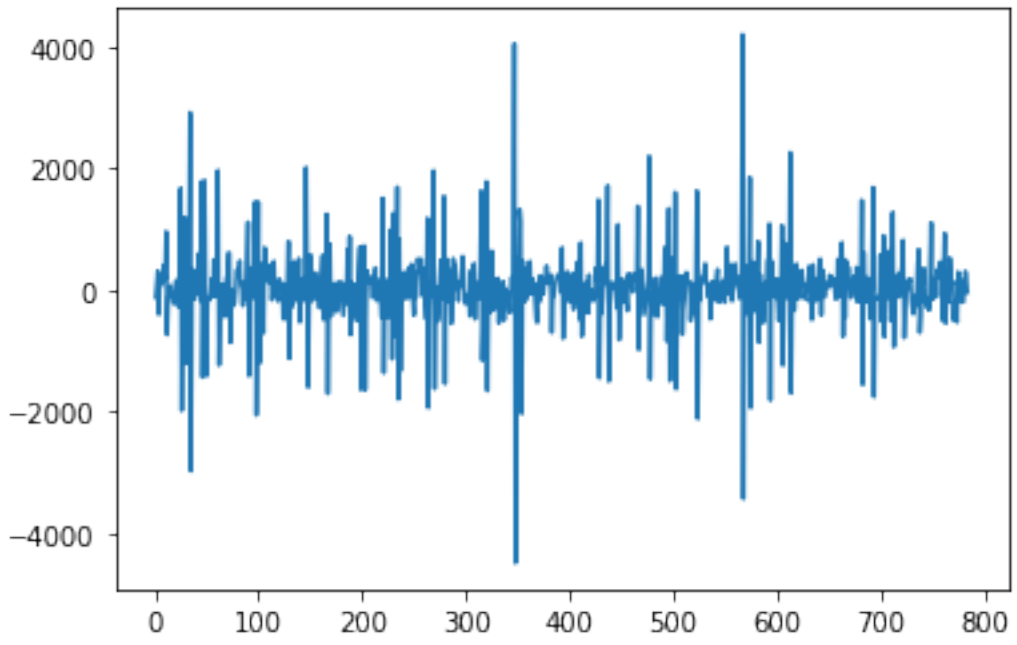
1.4.1 Time Series Modeling

ARIMA Here, a model is trained on the first chronological segment of the data. It then makes one prediction at a time and uses that prediction to make the following one. One of the most famous, and reliable model of this type is ARIMA (Autoregressive integrated moving average).

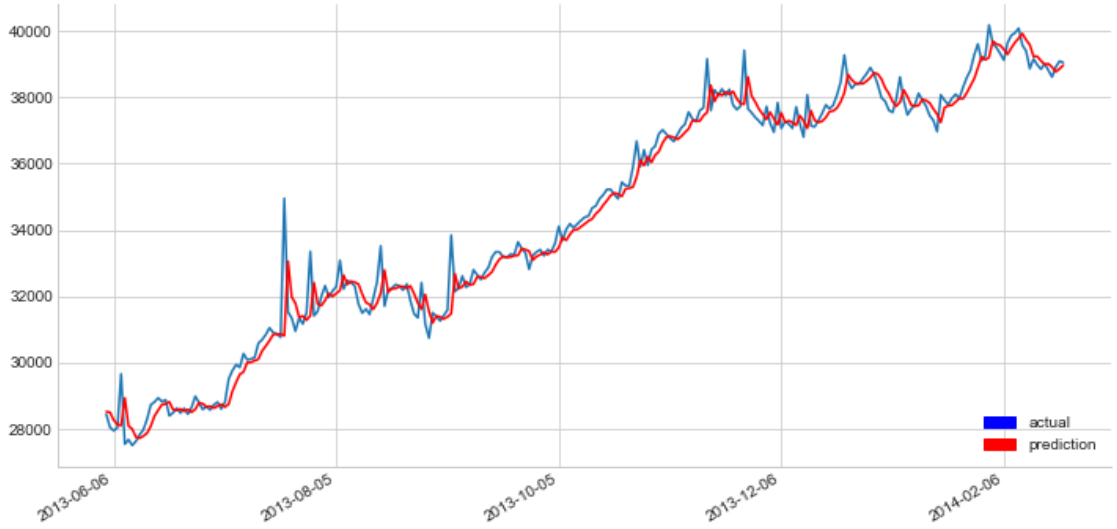
ARIMA uses previous values of the time series as well as its moving averages to make prediction. When the time series is not seasonal it needs to be differentiated for ARIMA to work well.

As we can see below, one degree of differentiation is enough to make our series seasonal.

[13]:
```
[<matplotlib.lines.Line2D at 0x1695b94c0>]
```




```
Test RMSE: 542.984
```



Given that ARIMA was trained on the first segment of the data before the upward trend in orders even began it is safe to say that it performs quite well.

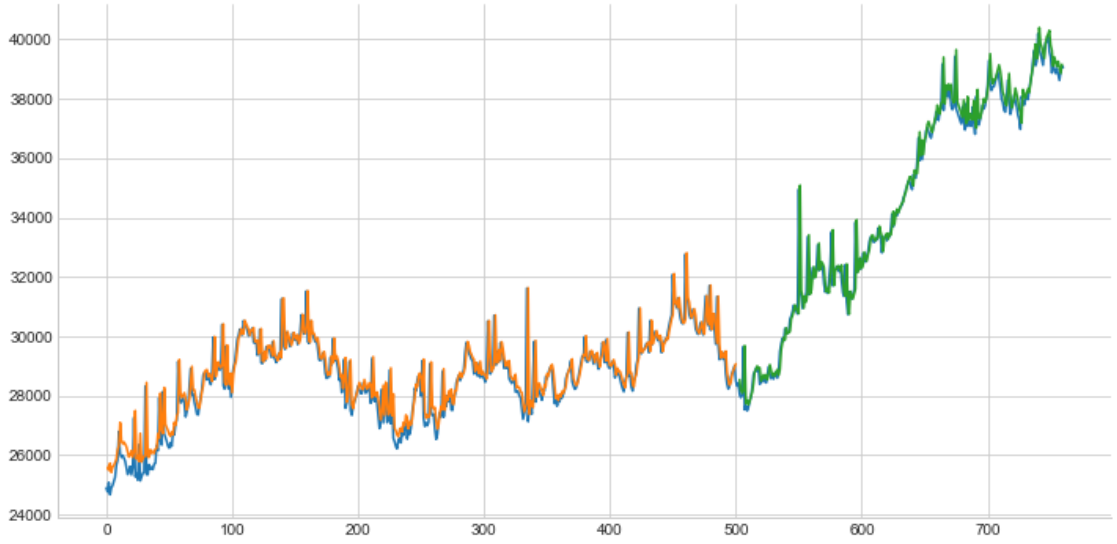
The parameters chosen were ARIMA(1,1,2) meaning that the time series was differentiated once and that we ask the model to only look two steps behind to determine the following one.

In this case ARIMA looks at the previous raw value and its difference with the moving average

between the two previous ones.

LSTM Neural Network Since deep learning started becoming mainstream in all areas of modeling, LSTMs which are good at predicting sequences (music and speech generation) are now being used to predict time series.

```
Epoch 1/10
500/500 [==============================] - 4s 2ms/step - loss: 0.0239
Epoch 2/10
500/500 [==============================] - 1s 3ms/step - loss: 0.0048
Epoch 3/10
500/500 [==============================] - 2s 3ms/step - loss: 0.0025A: 0s -
Epoch 4/10
500/500 [==============================] - 1s 3ms/step - loss: 0.0018
Epoch 5/10
500/500 [==============================] - 1s 3ms/step - loss: 0.0021
Epoch 6/10
500/500 [==============================] - 2s 3ms/step - loss: 0.0022
Epoch 7/10
500/500 [==============================] - 1s 3ms/step - loss: 0.0017
Epoch 8/10
500/500 [==============================] - 2s 3ms/step - loss: 0.0025
Epoch 9/10
500/500 [==============================] - 1s 2ms/step - loss: 0.0019
Epoch 10/10
500/500 [==============================] - 1s 2ms/step - loss: 0.0018
Train Score: 681.88 RMSE
Test Score: 631.37 RMSE
```



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 1, 10)             480
_________________________________________________________________
lstm_1 (LSTM)                (None, 10)                840
_________________________________________________________________
dense (Dense)                (None, 1)                 11
=================================================================
Total params: 1,331
Trainable params: 1,331
Non-trainable params: 0
_________________________________________________________________
```

This two-layer LSTM network is also good at predicting orders with an RMSE only slightly higher than ARIMA (632.03 vs 542.984). This being said, these models usually need greater volumes of data. In fact, when ARIMA optimizes for 3 parameters, the above LSTM optimizes for 1331 parameters. It is then understandable that a simpler model such as ARIMA has an edge in a data scarce situtation like this one. In the event we get more data, the LSTM might end up outperforming ARIMA

1.4.2 Machine Learning Regression

Time series models, by definition only use previous values of orders to make predictions and offer a good benchmark to try and apply machine learning with synthetic features built after having performed analytics in the first part of this exercise

Additional regressors

[17]:

	date	orders	circumstance	drivers_available	month	weekday	\
3	2012-01-05	25083.0	0	4691.0	1	3	
4	2012-01-06	24676.0	0	5308.0	1	4	
5	2012-01-07	24905.0	0	5905.0	1	5	
6	2012-01-08	24980.0	0	4512.0	1	6	
7	2012-01-09	25124.0	0	4186.0	1	0	

	orders_lag_1	orders_lag_2	month_cat
3	24774.0	24899.0	-1
4	25083.0	24774.0	-1
5	24676.0	25083.0	-1
6	24905.0	24676.0	-1
7	24980.0	24905.0	-1

```
This is a modified version of the initial dataframe.
For the regression we keep the following variables:
    - orders: float (target)
```

```
- circumstance: binary (1 when very rainy 0 otherwise)
- drivers_available: float
- month_cat (three categories:
        * 0 when the global mean is within the confidence interval of that months'
            mean orders
        * 1 when it is above
        * -1 when it is below)
- order_lag_i: the number of orders i days before that day
```

Below is the performance of the random forest when the data is shuffled. The synthetic features do help it greatly, and our hypotheses are confirmed. It is also the best performing model so far (albeit on a different test set)

```
Test RMSE with synthetic feature: 530.774
Test RMSE without synthetic feature (only order lags and drivers available):
635.936
```

When the data is not shuffled the Random Forest performs extremely poorly

```
Test RMSE with synthetic feature: 4950.237
```

1.5 Conclusion

The previous analysis and modeling shows us that very rainy days and the given month are quite predictive for orders.

The model to be used depends highly on the problem we want to solve. If the goal is to predict future orders then, traditional time series model that only look at previous values are the best choice given that the Random Forest Regressor performed very poorly on unshuffled data.

However if the goal is to impute missing values (i.e make an estimate of the number of orders on a day for which we lost/don't have data) the Random Forest is the way to go.

2 Part 2

2.1 Introduction

[21]:

	deviceId	devicetype	user_id	\
0	dev-a6ec70f1	NaN	id-52760576	
1	dev-42671881	Sony Ericsson ST15i (ST15i_1250-2612)	id-22891520	
2	dev-a6ec70f1	NaN	id-14748672	
3	dev-52480977	samsung GT-N7100 (t03gxx)	id-36466688	
4	dev-a6ec70f1	NaN	id-18005504	

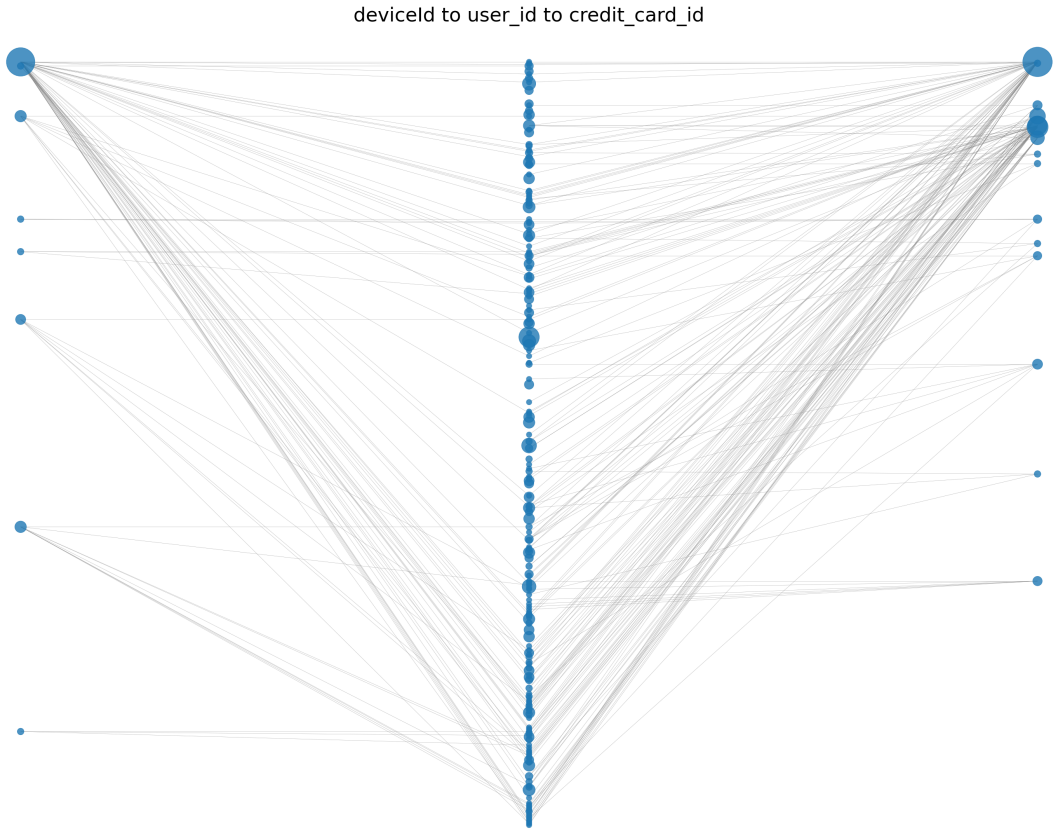
	credit_card_id	domain
0	card-f62c9e0f51	ddk.com
1	card-ab9b2255e1	yahoo.com
2	card-f62c9e0f51	dkdl.com

```
3  card-ab9b2255e1  yahoo.com
4  card-f62c9e0f51  djdjs.com
```

In studying this dataset we will focus on the relationships between deviceId, user_id and credit_card_id.

2.1.1 First Tripartite graph

[22]: `<networkx.classes.graph.Graph at 0x1695ff940>`



The three columns above represent deviceId, user_id and credit_card_id in that order.

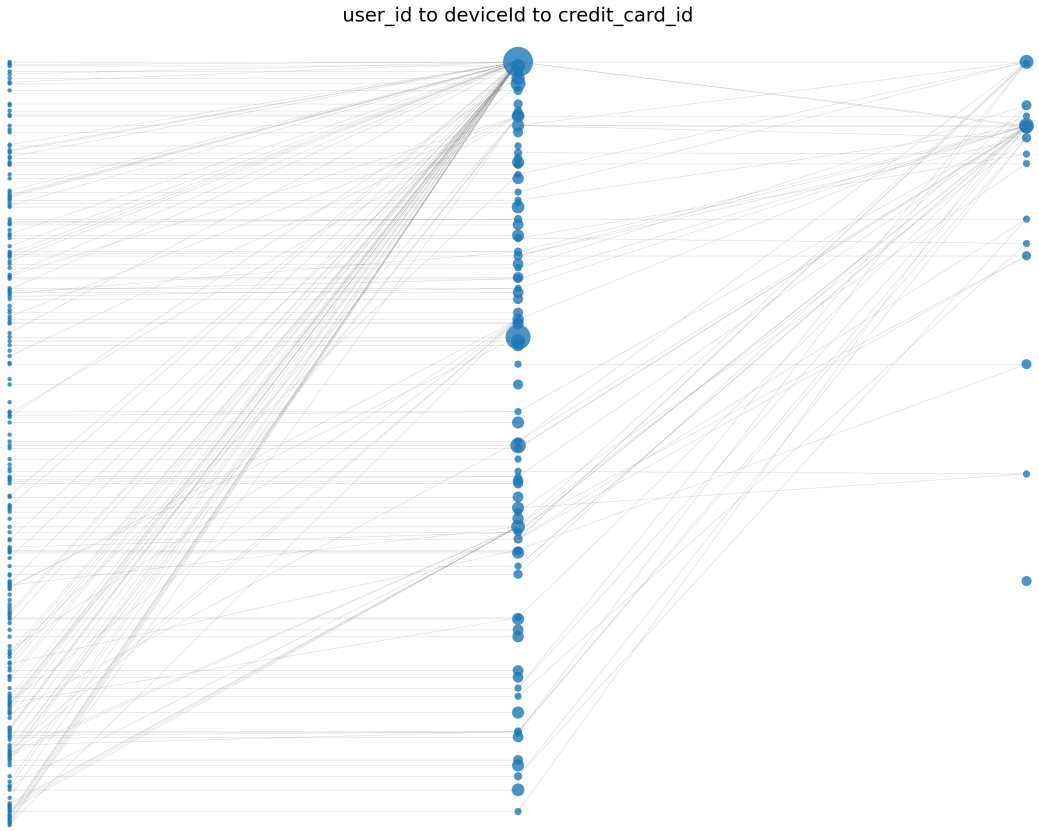
For better visibility, we remove all devices that are used by 2 or less users and all credit cards that are used by 2 or less users as that scenario is not as suspicious.

We can clearly see that a few major devices are used by dozens of users who then go on to use a few major credit cards, which brings us to our first hypothesis.

Hypothesis 1 Fraudsters create dozens of different accounts on their devices to use stolen credit cards, probably in order to mislead anybody who would try to track them by looking at the data

2.1.2 Second Tripartite graph

[23]: <networkx.classes.graph.Graph at 0x1695c22e0>



This helps us see even more clearly the pattern behind hypothesis 1. The fact that the device with the most users linked to it only uses 2 credit cards further consolidates the theory that only one person hides behind all these users. The link device to credit card shows that credit cards are often used on many different devices which leads us to Hypothesis 2.

Hypothesis 2 Fraudsters or networks of fraudsters tend to use many different devices for the same reason why they create many accounts. Either they share the stolen credit cards' data between them to maximize purchases before it gets blocked. Or, if they are working alone, they use multiple devices on which they create multiple accounts to create as much confusion as they possibly can while using a stolen card.

3 Part 3

3.1 Introduction

[25]:

	order_id	order_amount	delivery_latitude	delivery_longitude	order_date
0	ord-93386	27.51	51.798	-0.816	17/11/2016
1	ord-79675	29.56	51.537	-0.366	09/11/2016
2	ord-73206	33.05	52.484	-1.944	27/11/2016
3	ord-75858	27.51	52.556	-2.106	16/11/2016
4	ord-61645	29.76	51.385	-0.109	29/11/2016

```
This data set has 5 features:
    - order_id : string
    - order_amount: float
    - longitude: float
    - latitude: float
    - order_date: datetime
```

We can use latitude and longitude with external shp files to assign Regions and LSOAs to every row. Lower Layer Super Output Areas (LSOA) are a geographic hierarchy designed to improve the reporting of small area statistics in England and Wales. It is one of the most granular space subdivision in the country, which is the equivalent of a street bloc.

[26]:

	order_id	order_amount	order_date	rgn17nm	\
0	ord-93386	27.51	17/11/2016	South East	
1	ord-79675	29.56	09/11/2016	London	
2	ord-73206	33.05	27/11/2016	West Midlands	
3	ord-75858	27.51	16/11/2016	West Midlands	
4	ord-61645	29.76	29/11/2016	London	

	geometry	LSOA11NM
0	POINT (481745.376 211701.874)	Aylesbury Vale 020A
1	POINT (513425.065 183275.887)	Ealing 010C
2	POINT (403899.969 287343.600)	Birmingham 053B
3	POINT (392910.385 295356.435)	Wolverhampton 032B
4	POINT (531684.755 166802.497)	Croydon 015C

Now that we have regions and LSOA names we can segment our analysis by these categorical variables. Note that instead of longitude and latitude, we have a geometrical point typical of geopandas.DataFrame

3.2 Fraud by Region

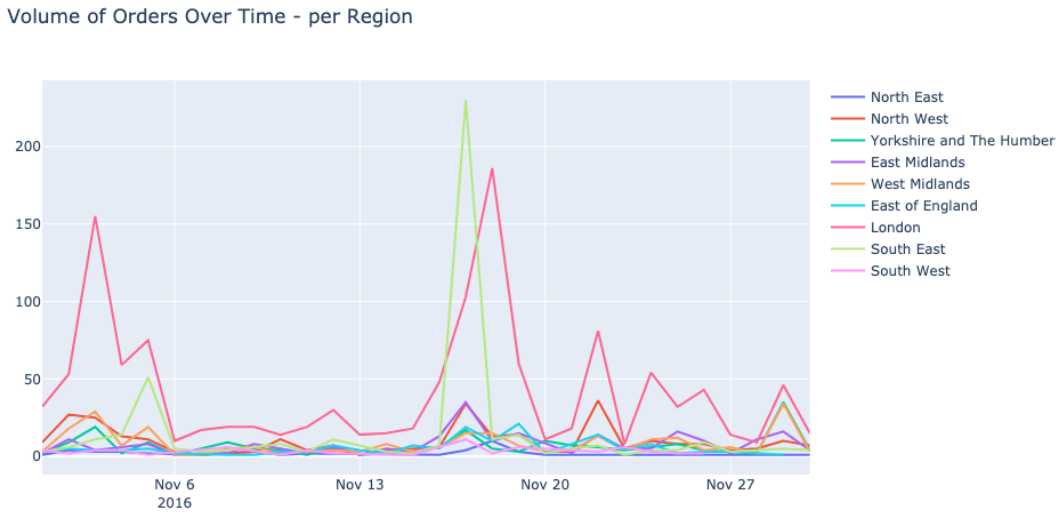
We start by the largest subdivision which is the region. Let us look at fraudulent orders over time across the 9 regions present in the dataset

3.2.1 Over Time

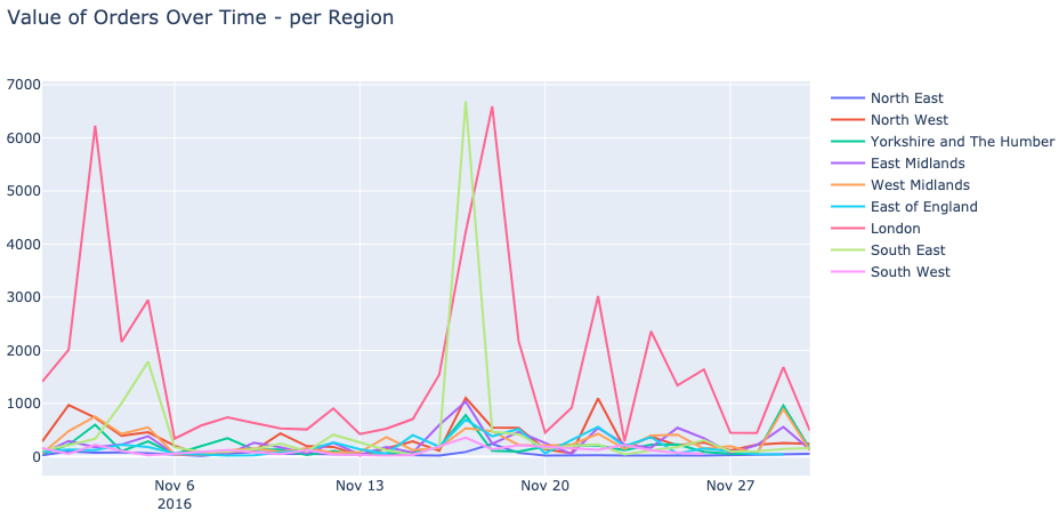
[28]:

		order_amount	
		sum	count
rgn17nm	order_date		
East Midlands	2016-11-01	56.37	2
	2016-11-02	294.31	11
	2016-11-03	173.88	4
	2016-11-04	226.34	6
	2016-11-05	384.62	8

[29]:



[30]:

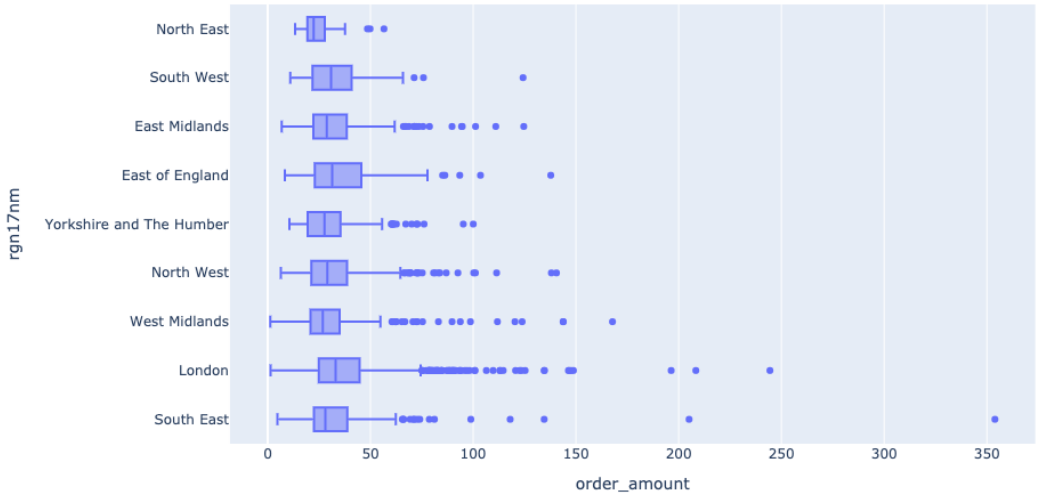


These two line plots show that fraudulent orders do not happen uniformly over time. In fact, fraud is punctual. We can see that the lines shift from baseline when fraud happens, because it happens all at the same time (i.e in the South East). This being said, London distinguishes itself from other regions in the frequency at which fraud happens. When most regions are predominantly flat with some dents here and there, the London line looks more like shark teeth.

3.2.2 Distribution of orders value

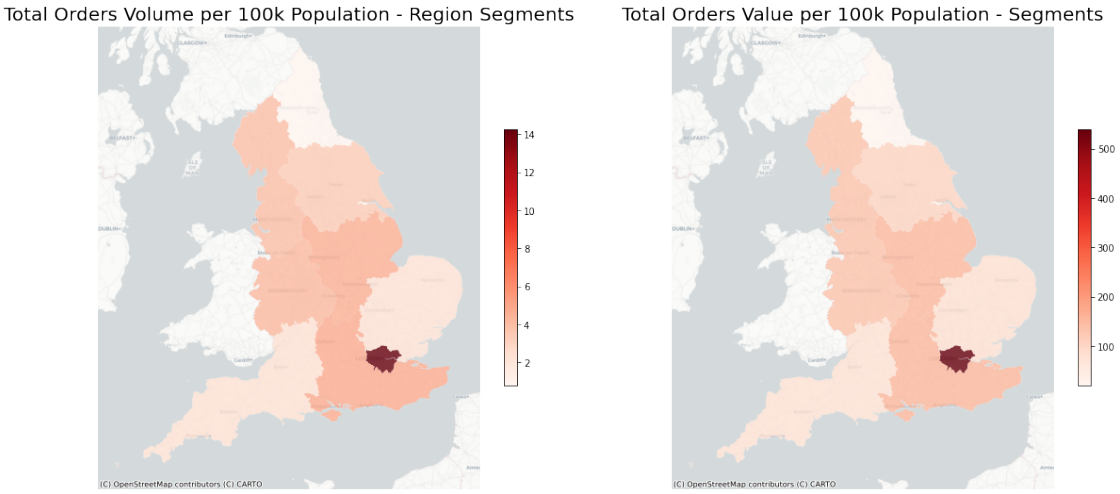
Let us look at the distribution of fraudulent orders value across regions. Are there regions where the value of orders is generally higher than other ?

[31]:



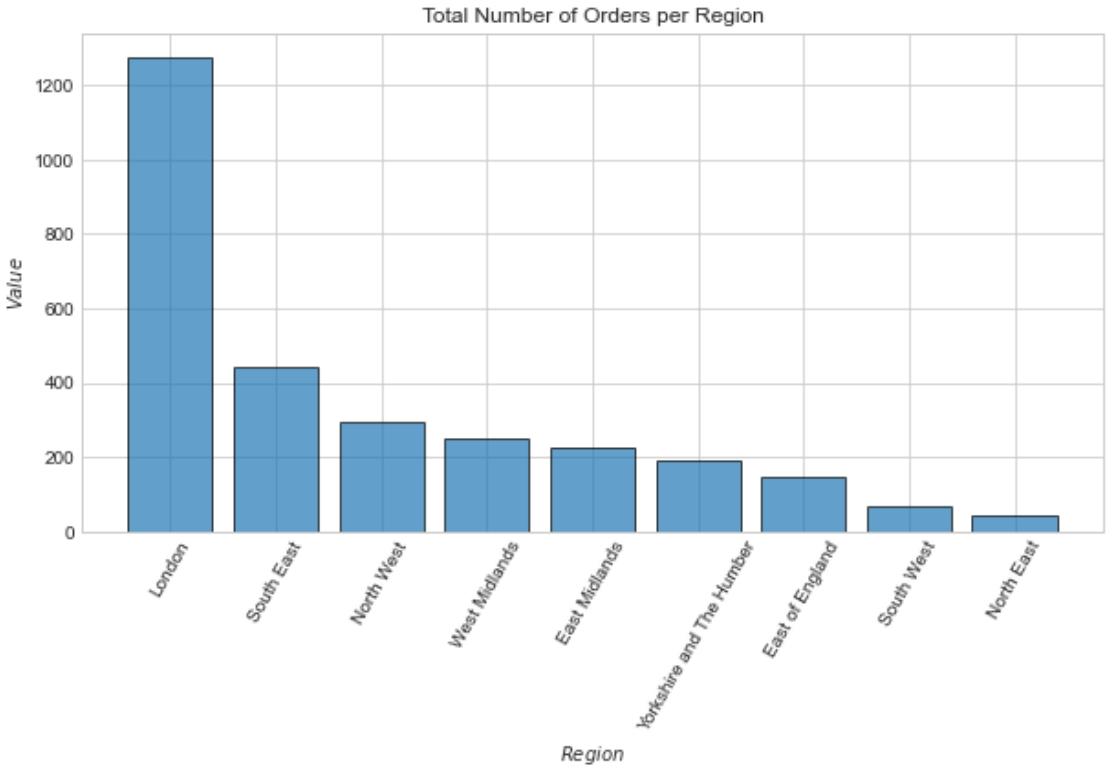
This tells us that most fraudulent orders are in the same range of values between 1 and 75 pounds, with slightly more outliers in London, the South East and other regions

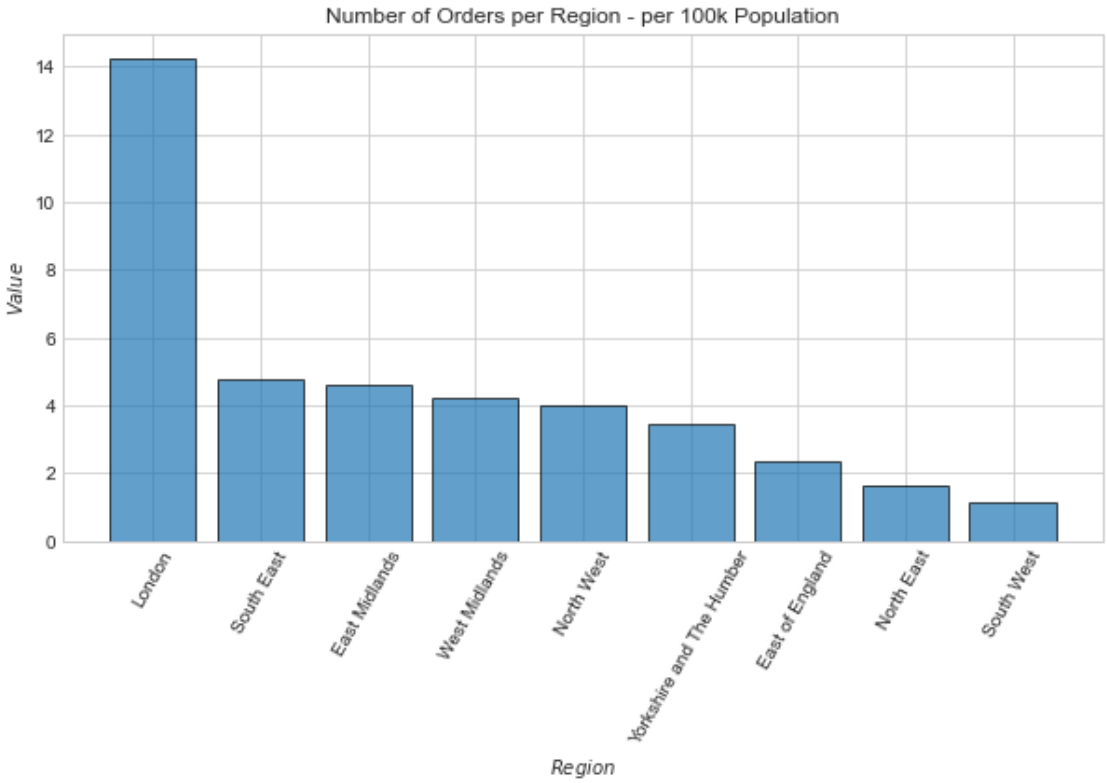
3.2.3 HeatMaps of England

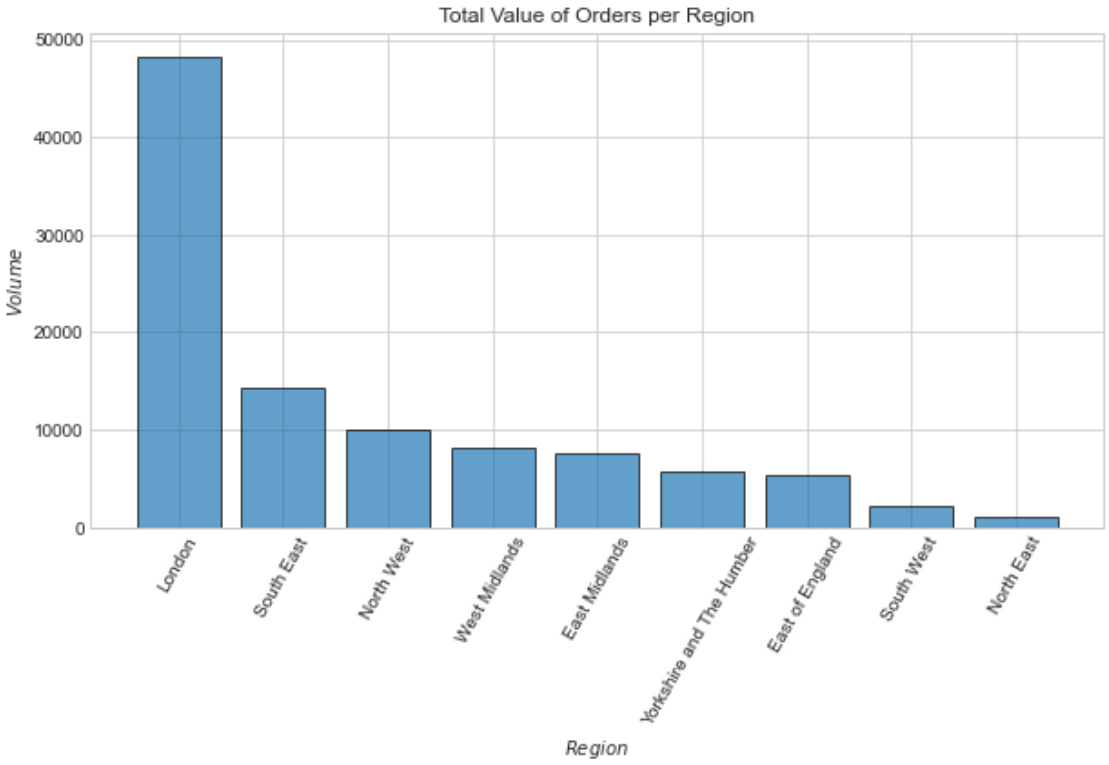


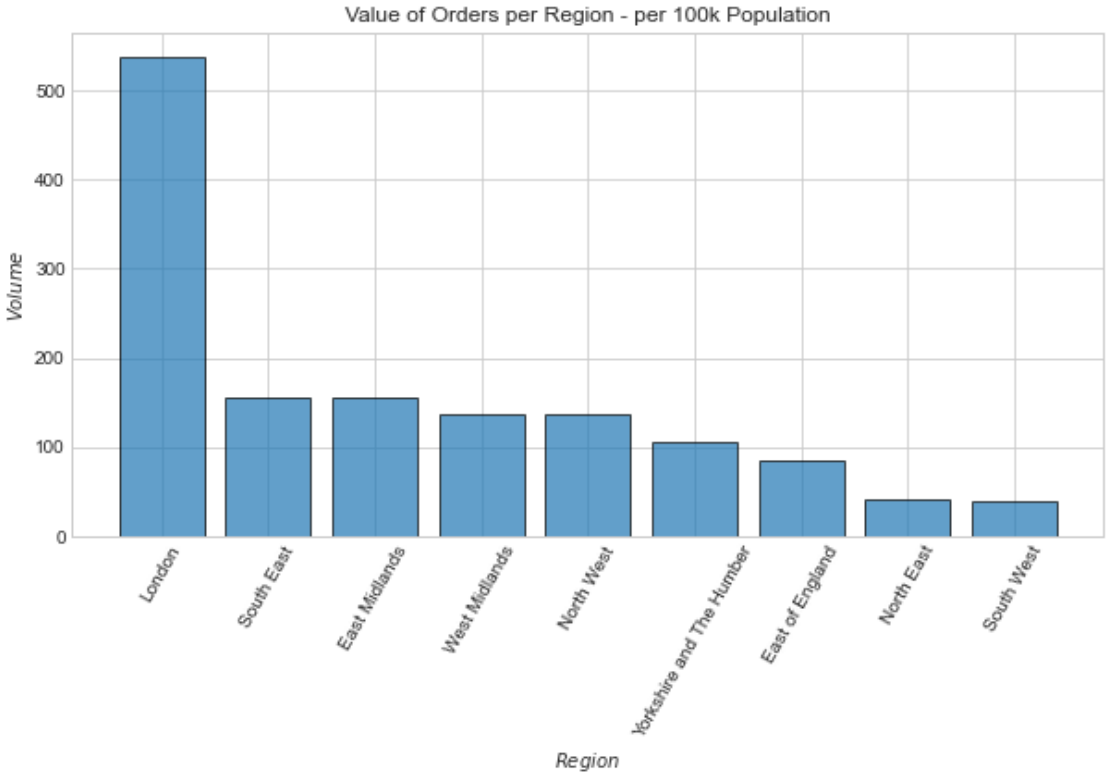
Even when the data is scaled according to the regions' populations London always clearly stands out. Looking at the bar charts below will tell us exactly by how much more, London experiences fraudulent orders.

3.2.4 Bar Charts for Volume and Value of Orders Segmented by Region




Number of Orders per Region - per 100k Population
Value
0
2
4
6
8
10
12
14
London
South East
East Midlands
West Midlands
North West
Yorkshire and The Humber
East of England
North East
South West
Region


Total Value of Orders per Region
50000
40000
30000
20000
10000
0
Volume
London
South East
North West
West Midlands
East Midlands
Yorkshire and The Humber
East of England
South West
North East
Region


Value of Orders per Region - per 100k Population
500
400
300
200
100
0
Volume
London
South East
East Midlands
West Midlands
North West
Yorkshire and The Humber
East of England
North East
South West
Region

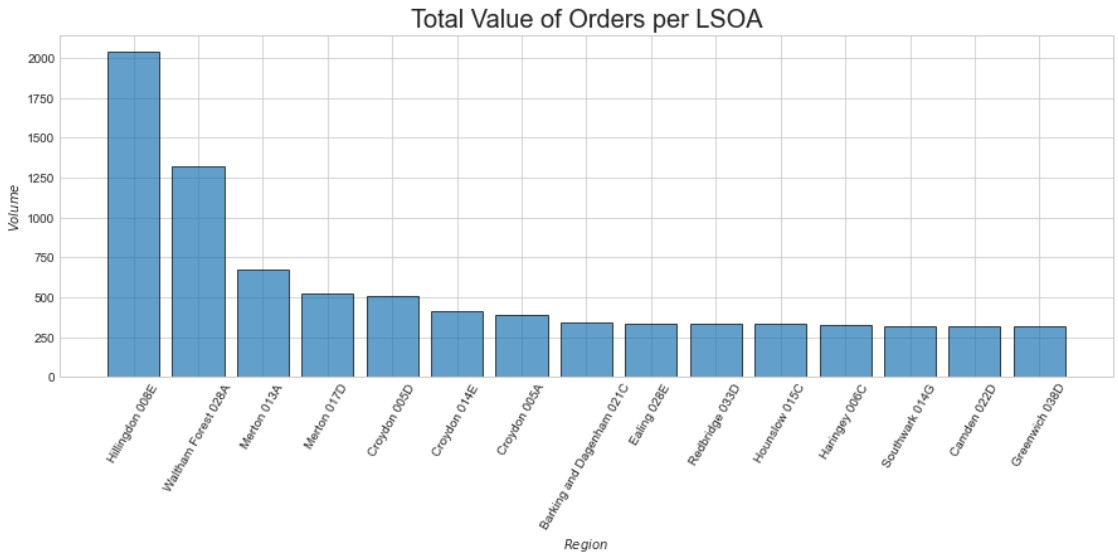
London dominates in fraud on all metrics. We assume that the 80/20 principle applies here (The Pareto principle states that for many outcomes roughly 80% of consequences come from 20% of the causes) and we recommend to concentrate all efforts of analysis and monitoring to the Greater London Region

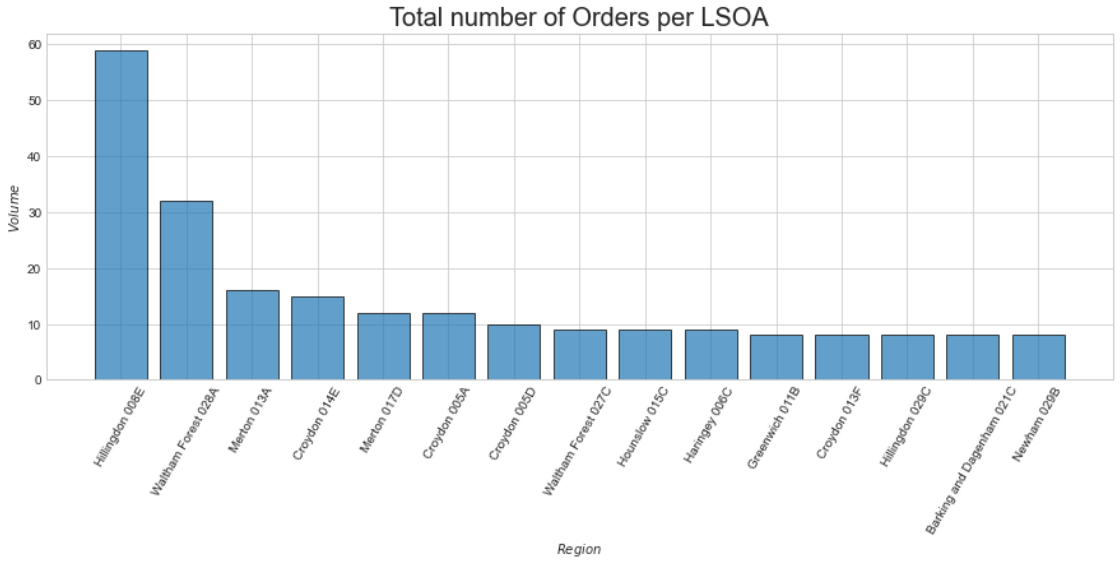
3.3 London Deep Dive

Even when the numbers are scaled by its large population, London remains the most important fraud hub in the country. Let us use the smallest subdivision we have at our disposal (LSOA) to determine from where exactly these fraudulent transactions are taking place.

3.3.1 Order Values and Volumes by LSOA - Bar Charts

Below are the top 15 LSOAs by total number of orders and their respective values. The values decrease quickly and start stagnating. This demonstrates that some LSOAs have hosted major one-time fraud schemes, probably like the one that happened in London on November 18th or in the South East on November 17th (refer to the first two graphs of part 3)





[36]:
```
order_date
18/11/2016    59
dtype: int64
```

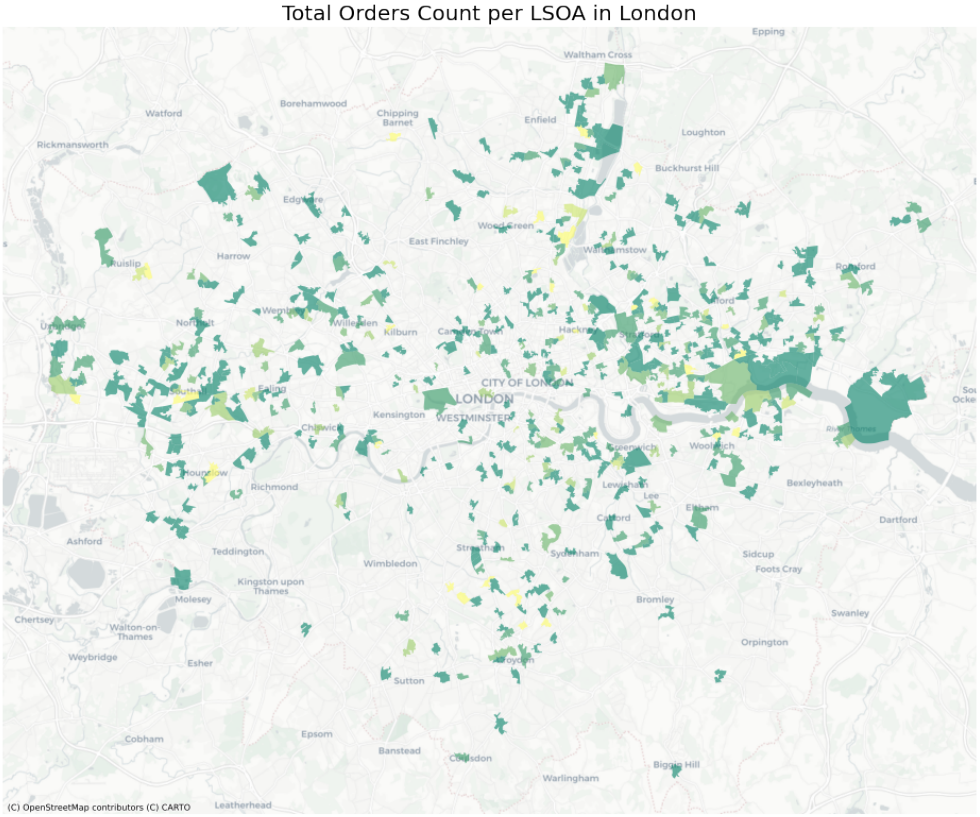
This confirms that the 59 fraudulent orders that happened in Hillingdon 008E took place on 18/11/2016 which confirms this section's hypothesis. Not only that, but these orders all had the exact same coordinates. This dataset can help locate the exact place where major frauds took place.

3.3.2 How can this help us detect fraud ?

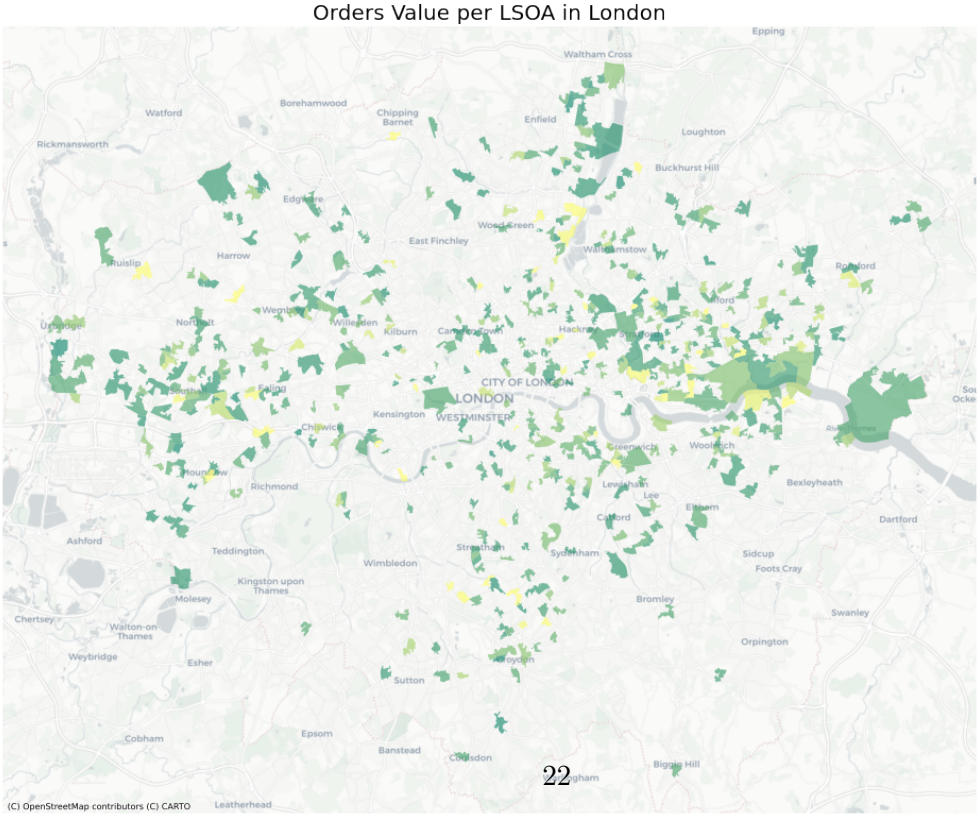
Being able to locate fraud, especially outliers like the Hillington incident, is a good thing. It can help the authorities catch major fraudsters. However, Ravelin is in the business of fraud detection. The interest is not to find fraudsters after the fact, but to detect and block them in real time. Now that we know that most fraudulent orders happen in London, how can we use this to build predictive features for a model.

3.3.3 HeatMaps of London

Total Orders Count per LSOA in London



Orders Value per LSOA in London



These maps show that the fraudulent orders spatial distribution in London is far from random. Some areas could be considered fraud hubs with a large concentration of neighboring LSOAs in which fraud occurs. Finding a good way of binning these LSOAs into categories of areas in London could result in the creation of a predictive feature to train a fraud detection algorithm

3.3.4 Limitations of the dataset

This dataset only gives the date, the place and the "value" of the crime. Other information that could be useful:

```
1. Time:
Knowing the exact time of an order would help us establish patterns
as to how frequently fraudsters during a given day use a stolen
credit card to make orders.

2. Type of purchase:
Knowing what kind of products/service fraudsters usually go for would
allow us to bin them according to their risk factor. Thus creating
a predictive categorical variable

3. credit_card_id, user_id, device_id:
Having the information in the dataset of Part 2 would enable us to
visualize these frauds in their wider network. Furthermore, we would
prove the hypothesis of the multiple devices and users for a single
individual as they would all point to the same location.
```