911 Calls Capstone Project

Данные для этого проекта можно скачать из [Kaggle] (https://www.kaggle.com/mchirico/montcoalert (https://www.kaggle.com/mchirico/montcoalert)) или из dropbox. Описание полей:

- lat : String variable, Latitude
- lng: String variable, Longitude
- desc: String variable, Description of the Emergency Call
- zip: String variable, Zipcode (индекс)
- title: String variable, Title
- timeStamp: String variable, YYYY-MM-DD HH:MM:SS (дата и время)
- twp: String variable, Township (поселение)
- addr: String variable, Address
- e: String variable, Dummy variable (always 1)

Import numpy and pandas

In [129]:

Import matplotlib, seaborn

In [130]:

Загрузите csv файл

In [131]:

head()

In [155]:

Out[155]:

	lat	lng	desc	zip	title	timeStamp	twp	addr	e	Reason	Hour	Month	Day of Week
0	40.297876	-75.581294	REINDEER CT & DEAD END; NEW HANOVER; Station ...	19525.0	EMS: BACK PAINS/INJURY	2015-12-10 17:40:00	NEW HANOVER	REINDEER CT & DEAD END	1	EMS	17	12	Thu
1	40.258061	-75.264680	BRIAR PATH & WHITEMARSH LN; HATFIELD TOWNSHIP...	19446.0	EMS: DIABETIC EMERGENCY	2015-12-10 17:40:00	HATFIELD TOWNSHIP	BRIAR PATH & WHITEMARSH LN	1	EMS	17	12	Thu
2	40.121182	-75.351975	HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-St...	19401.0	Fire: GAS-ODOR/LEAK	2015-12-10 17:40:00	NORRISTOWN	HAWS AVE	1	Fire	17	12	Thu

Простые вопросы

5 самых частых индексов (zipcodes)?

In [134]:

Out[134]:
```
19401.0    6979
19464.0    6643
19403.0    4854
19446.0    4748
19406.0    3174
Name: zip, dtype: int64
```

5 самых частных поселений (twp)?

In [135]:

Out[135]:
```
LOWER MERION    8443
ABINGTON        5977
NORRISTOWN      5890
UPPER MERION    5227
CHELTENHAM      4575
Name: twp, dtype: int64
```

Посмотрите на столбец 'title', сколько разных кодов в нём существует?

In [136]:

Out[136]:
```
110
```

Создание полей

В поле title перед кодом указана причина звонка - EMS, Fire, Traffic. Используйте .apply() с lambda-выражением для создания отдельного поля Reason, в котором будет указана причина.

Например, если в title указано EMS: BACK PAINS/INJURY , Reason будет EMS.

In [137]:

Какое самое рспространённое значение в поле Reason?

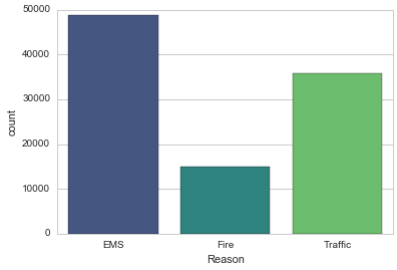
In [138]:

Out[138]:
```
EMS        48877
Traffic    35695
Fire       14920
Name: Reason, dtype: int64
```

Используйте seaborn для создания countplot по полю Reason.

In [139]:

Out[139]: <matplotlib.axes._subplots.AxesSubplot at 0x12d3830b8>



Какой тип данных у поля timeStamp?

In [140]:

Out[140]: str

Используйте [pd.to_datetime] (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.to_datetime.html (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.to_datetime.html)) для конвертирования этого поля в DateTime.

In [184]:

** Теперь вы можете получать отдельные свойства этого поля, например hour:**

```
time = df['timeStamp'].iloc[0]
time.hour
```

** Используйте .apply() для создания 3 новых полей: Hour, Month, и Day of Week (день недели). **

In [142]:

Обратите внимание на то, что Day of Week - это integer 0-6. Используйте .map() для того, чтобы нормально назвать дни недели:

```
dmap = {0:'Mon',1:'Tue',2:'Wed',3:'Thu',4:'Fri',5:'Sat',6:'Sun'}
```

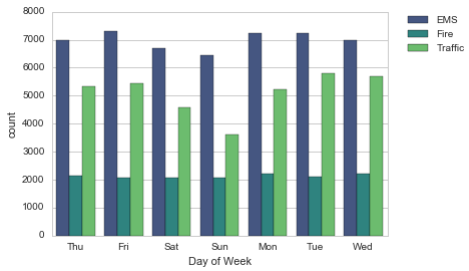
In [143]:

In [144]:

Используйте seaborn для создания countplot по полю Day of Week в разрезе (hue) поля Reason.

In [168]:

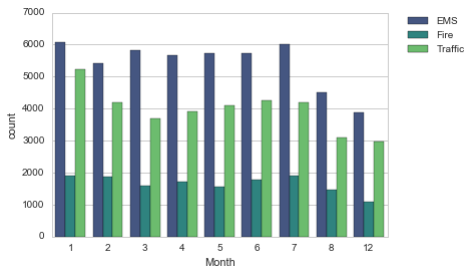
Out[168]: <matplotlib.legend.Legend at 0x12f614048>



То же самое для Month:

In [3]:

Out[3]: <matplotlib.legend.Legend at 0x10330ada0>



Вы должны были заметить, что там отстутствуют некоторые месяцы. Давайте попробуем заполнить эту информацию с помощью Pandas

Сгруппируйте данные по месяцам, и используйте count() для подсчёта. Вызовите head() для этого DataFrame.

In [169]:

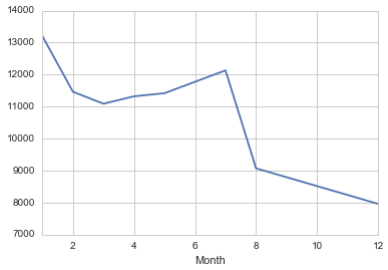
Out[169]:

	lat	lng	desc	zip	title	timeStamp	twp	addr	e	Reason	Hour	Day of Week
Month												
1	13205	13205	13205	11527	13205	13205	13203	13096	13205	13205	13205	13205
2	11467	11467	11467	9930	11467	11467	11465	11396	11467	11467	11467	11467
3	11101	11101	11101	9755	11101	11101	11092	11059	11101	11101	11101	11101
4	11326	11326	11326	9895	11326	11326	11323	11283	11326	11326	11326	11326
5	11423	11423	11423	9946	11423	11423	11420	11378	11423	11423	11423	11423

Постройте обычный plot, чтобы посмотреть звонки по месяцам.

In [175]:

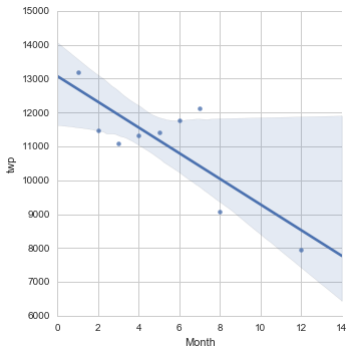
Out[175]: <matplotlib.axes._subplots.AxesSubplot at 0x133a3c080>



Попробуйте использовать seaborn lmplot() для того, чтобы посмотреть на регрессию. Возможно, для этого вам придётся сбросить индекс (reset index).

In [187]:

Out[187]: <seaborn.axisgrid.FacetGrid at 0x1342acd30>

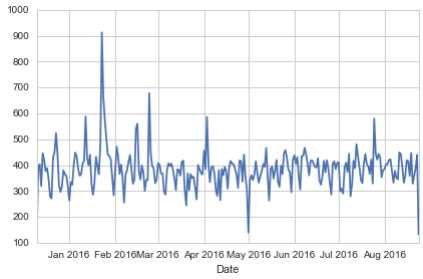


Создайте новое поле 'Date' из timeStamp. Для этого используйте apply и .date()

In [193]:

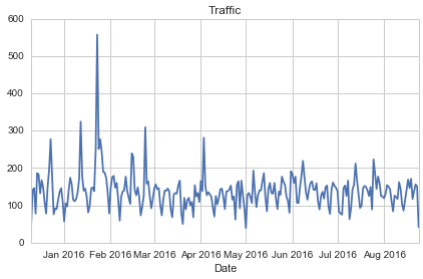
Теперь снова сгруппируйте по Date, используйте count() для подсчета, и постройте plot.

In [197]:

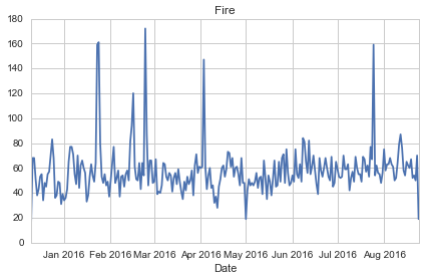


Постройте такой же график, но разбитый на 3 разных графика с разными причинами вызовов (Reason)

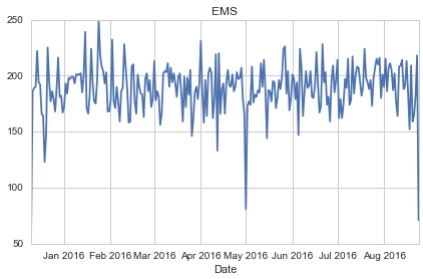
In [199]:



In [201]:



In [202]:



Попробуем построить тепловые карты, используя seaborn. Для начала необходимо реструктурировать dataframe так, чтобы в столбцах у нас были часы, а индексами дни недели. Лучше всего это сделать с помощью groupby, а затем [unstack] (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.unstack.html (http://pandas.pydata.org/pandas-docs/stable/generated/pandas.DataFrame.unstack.html)).

In [203]:

Out[203]:

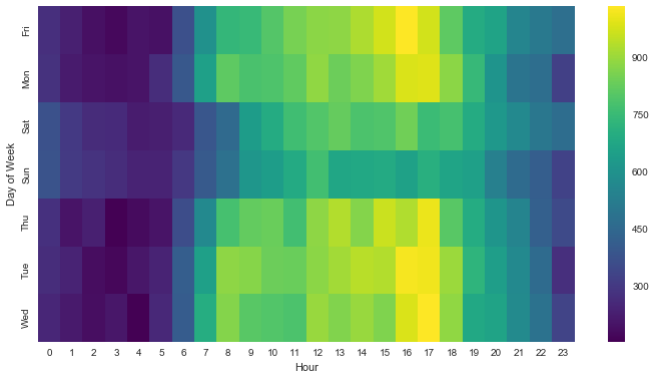
Hour	0	1	2	3	4	5	6	7	8	9	...	14	15	16	17	18	19	20	21	22	23
Day of Week																					
Fri	275	235	191	175	201	194	372	598	742	752	...	932	980	1039	980	820	696	667	559	514	474
Mon	282	221	201	194	204	267	397	653	819	786	...	869	913	989	997	885	746	613	497	472	325
Sat	375	301	263	260	224	231	257	391	459	640	...	789	796	848	757	778	696	628	572	506	467
Sun	383	306	286	268	242	240	300	402	483	620	...	684	691	663	714	670	655	537	461	415	330
Thu	278	202	233	159	182	203	362	570	777	828	...	876	969	935	1013	810	698	617	553	424	354

5 rows × 24 columns

Теперь постройте HeatMap из этого DataFrame.

In [204]:

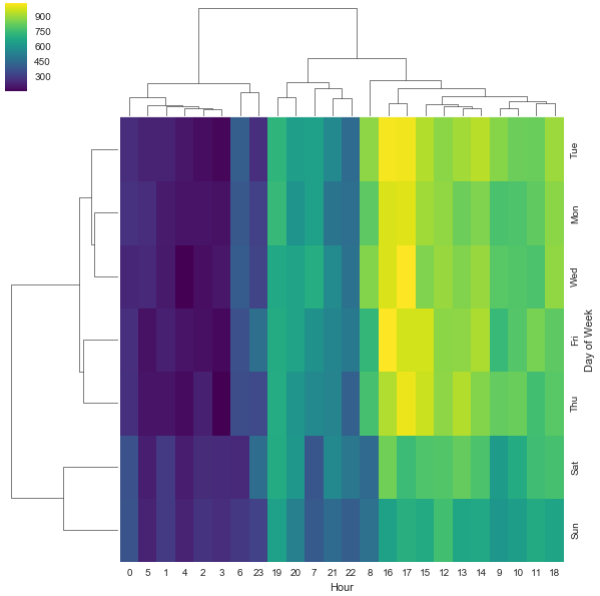
Out[204]: `<matplotlib.axes._subplots.AxesSubplot at 0x1253fa198>`



Теперь постройте clustermap из того же DataFrame.

In [205]:

Out[205]: <seaborn.matrix.ClusterGrid at 0x1304fb668>



Теперь повторите то же самое для DataFrame, с месяцами в столбцах.

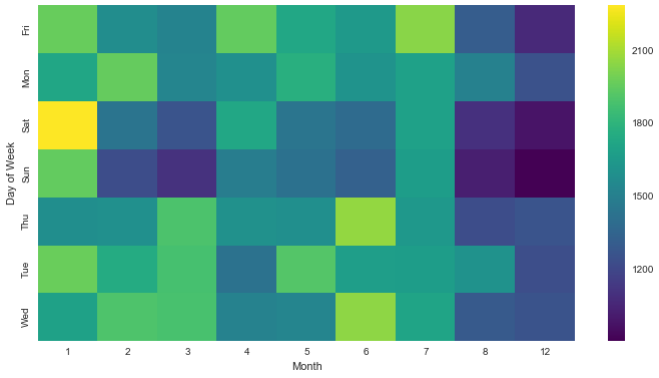
In [207]:

Out[207]:

Month	1	2	3	4	5	6	7	8	12
Day of Week									
Fri	1970	1581	1525	1958	1730	1649	2045	1310	1065
Mon	1727	1964	1535	1598	1779	1617	1692	1511	1257
Sat	2291	1441	1266	1734	1444	1388	1695	1099	978
Sun	1960	1229	1102	1488	1424	1333	1672	1021	907
Thu	1584	1596	1900	1601	1590	2065	1646	1230	1266

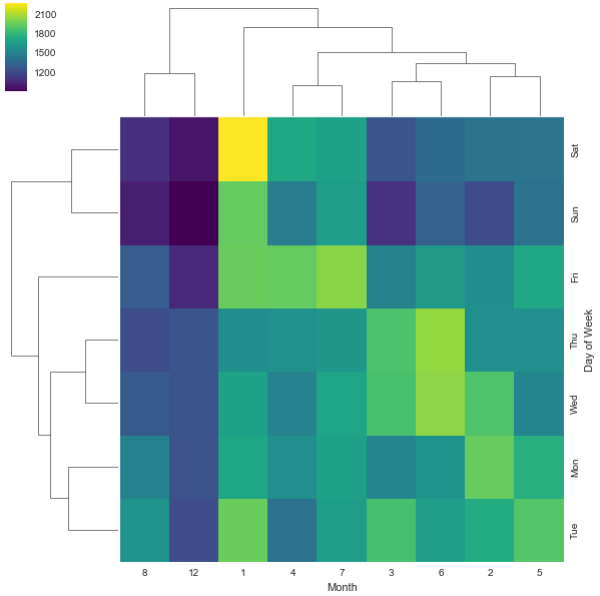
In [208]:

Out[208]: <matplotlib.axes._subplots.AxesSubplot at 0x1304fbd30>



In [209]:

Out[209]: <seaborn.matrix.ClusterGrid at 0x12a1a61d0>



Поздравляем! Проект успешно завершён! :)