Analysis of Vacant Buildings

December 19, 2018

Team: Bhanu Phanindra, Phani Teja Kesha

1 Title: Deserted neighborhood: Reason for crime?

Overview:We are working on vacant buildings data set. We planning to derive relation if possible with factors like crime, service requests and tax affecting vacancy of buildings in area. This dataset contains building addresses, police district and NoticeDate of buildings that are vacant. It does not contain any info regarding non-vacant buildings.

Datasets used are vacant builings, Real_PropertyTaxes, 311_Customer_Service_Requests and BPD_Part_1_Victim_Based_Crime_Data.

The following are the technical details given in openbaltimore about service requests dataset. Number of rows and columns are 3.25M and 22 respectively where each row is a service request. Despite being huge dataset it has large number of downloads. A table preview containging all columns is given in the same page which gives info about what we can expect from this dataset even before we download. Other than these technical details about dataset nothing else is mentioned about services or what this dataset is actually.

Dataset looks almost clean with some missing values in few columns. Looks like particular standard is followed for data entry. So there is not much ambiguity in notation. Although format of StatusDate, DueDate and CloseDate is same, CreatedDate is slightly different. I also spotted one duplicate row in 50 rows, which is mentioned as duplicate in service request itself. I think it only possible to get duplicate entries only when service request method is API. This can be investigated further when processing the data. Also may be not all duplicate requests are identified, there is possibility that different service request methods have been tried for exact same service.

One of the redundent information is GeoLocation, since Latitude and Longitude have their own colums. GeoLocation column can be dropped with out any worries. Before looking at output of tail(50) I assumed SRRecordID to be integer. But it looks like some random text is present at end of digits. Not sure whether it is intended or faulty recordID. Since it is just an Id it does not matter much.

Some technical details about vacant builings dataset given in open baltimore are Number of rows and columns are 16.7k and 9 respectively where each row is contains information about vacant building. It has 5.4K downloads and updated twice a month. There are not many null values in the dataset only one row in which there is not building address, so lets go ahead and delete the row which has a null value. To make the dataset clean.

Some technical details about property tax dataset given in open baltimore are Number of rows and columns are 239k and 16 respectively where each row is contains tax information for each builing. - There are some NULL values in the column AmountDue. - Null values in columns

CityTax and StateTax but the NULL values are not seen as frequently as seen in the AmountDue. - There are only two values in the column ResCode that is either it is a princial residence or not a principal residence. - The CouncilDistrict column looks like a code for the area but it is given in float value even though all values seems to be integer - There are some null values in columns Neighbourhood, PoliceDistrict, CouncilDistrict, Location. There is a row in which all these are null, i suspect all these to be null if any one of the value is null.

Column names with null values count

```
In [5]: df_vacant.isnull().sum(axis=0)

Out[5]: ReferenceID        0
        Block              0
        Lot                0
        BuildingAddress    1
        NoticeDate         0
        Neighborhood       0
        PoliceDistrict     0
        CouncilDistrict    0
        Location           0
        dtype: int64
```

Now let's clean the dataset such that it has no null values and no outliers in the given dataset. To achieve this we can follow some preprocessing techniques like filling with a global value, predicting the value etc.

```
In [9]: df_vacant.isnull().sum(axis=0)

Out[9]: ReferenceID        0
        Block              0
        Lot                0
        BuildingAddress    0
        NoticeDate         0
        Neighborhood       0
        PoliceDistrict     0
        CouncilDistrict    0
        Location           0
        dtype: int64
```

Now lets check for unique values in each column so that we can find outliers if there are any in dataset.

```
In [10]: df_vacant['CouncilDistrict'].unique()

Out[10]: array([ 7,  9, 11, 10, 12, 13,  8,  6,  5,  4, 14,  2,  1,  3],
                dtype=int64)
```

Its considering the coloumn as type sensitive,convert into a single case for no confusion and to get clearer results

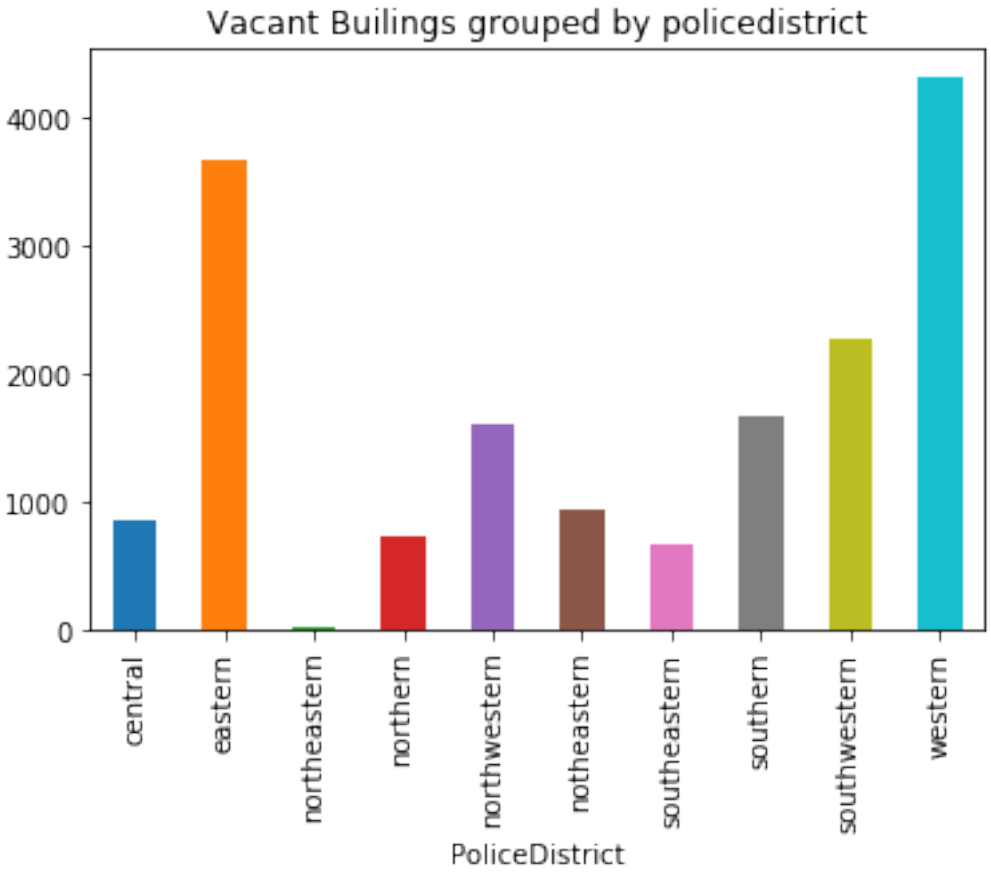
```
In [11]: df_vacant['PoliceDistrict'].unique()

Out[11]: array(['WESTERN', 'Western', 'SOUTHERN', 'Southwestern', 'CENTRAL',
                'Southern', 'EASTERN', 'SOUTHEASTERN', 'SOUTHWESTERN', 'Eastern',
                'Southeastern', 'Northwestern', 'NORTHWESTERN', 'NORTHERN',
                'Notheastern', 'Central', 'Northern', 'NORTHEASTERN'], dtype=object)
```

Here we see typo, which resulted in duplicate values. This is handled further down in notebook.

```
In [12]: df_vacant['PoliceDistrict']=df_vacant['PoliceDistrict'].str.lower()

In [14]: df_vacant['PoliceDistrict'].unique()

Out[14]: array(['western', 'southern', 'southwestern', 'central', 'eastern',
                'southeastern', 'northwestern', 'northern', 'notheastern',
                'northeastern'], dtype=object)
```

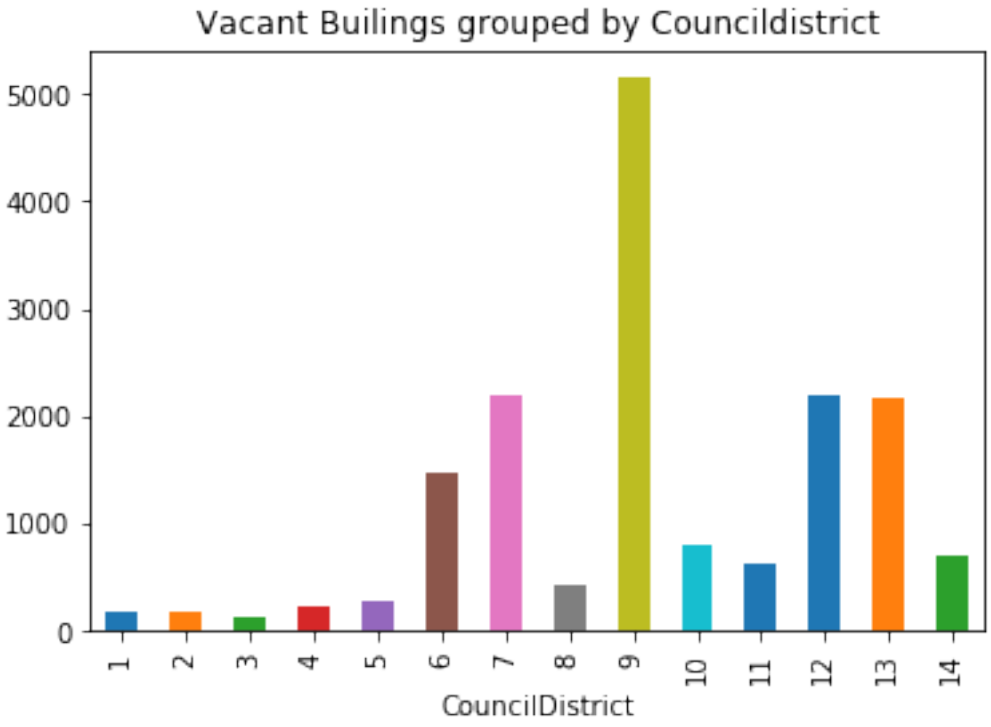
The vacant buildings in each police district is counted and plot for police district vs its count is created. From the below plot we can see following insights. 1. SouthEastern and Northern districts haveleast number of vacant buildings. 2. Western has highest number of vacant buildings.

```
In [18]: df_vacant.groupby(['PoliceDistrict']).count()['Lot'].plot(kind='bar', title='Vacant Bu

Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x17fd5bf0fd0>
```



Plot when groupby is used on PoliceDistrict.

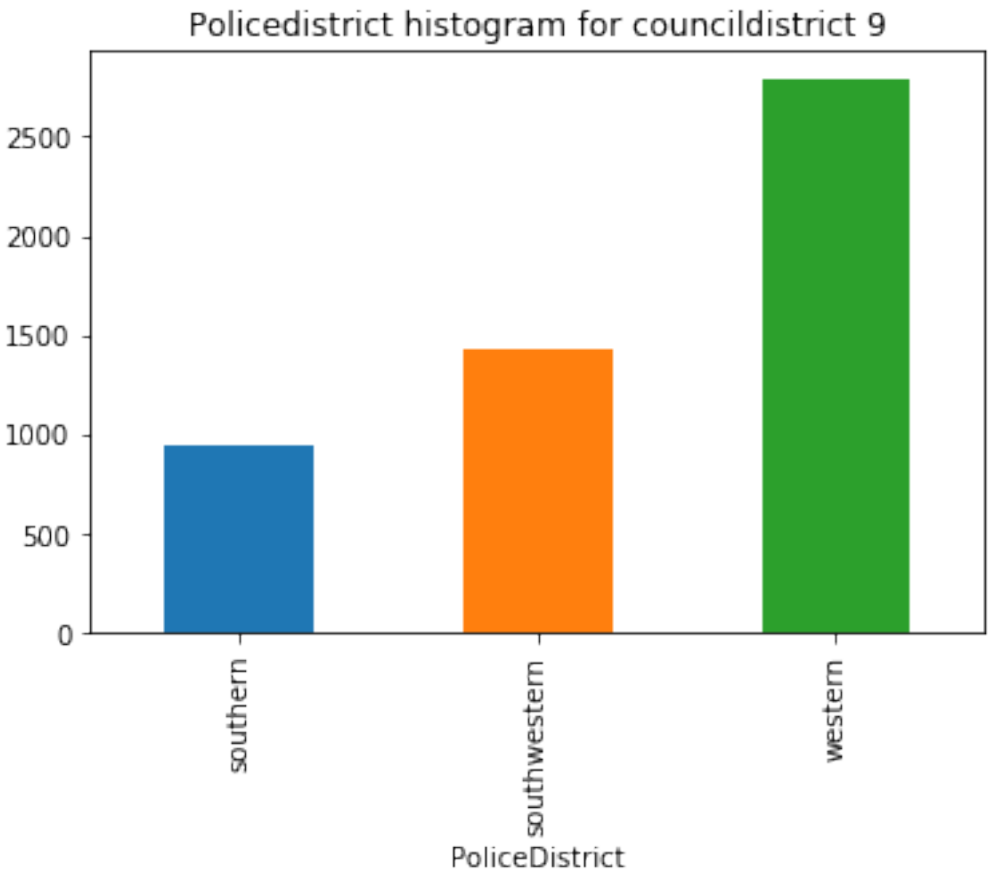
```
In [19]: df_vacant.groupby(['CouncilDistrict']).count()['Lot'].plot(kind='bar',title='Vacant Bu

Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x17fd5d54e48>
```



The vacant buildings in each council district is counted and plot for council district vs its count is created. From the below plot we can see following insights. 1. Third council have least number of vacant buildings. 2. Nineth council has highest number of vacant buildings. 3. Maximum is leading other districts by large amount

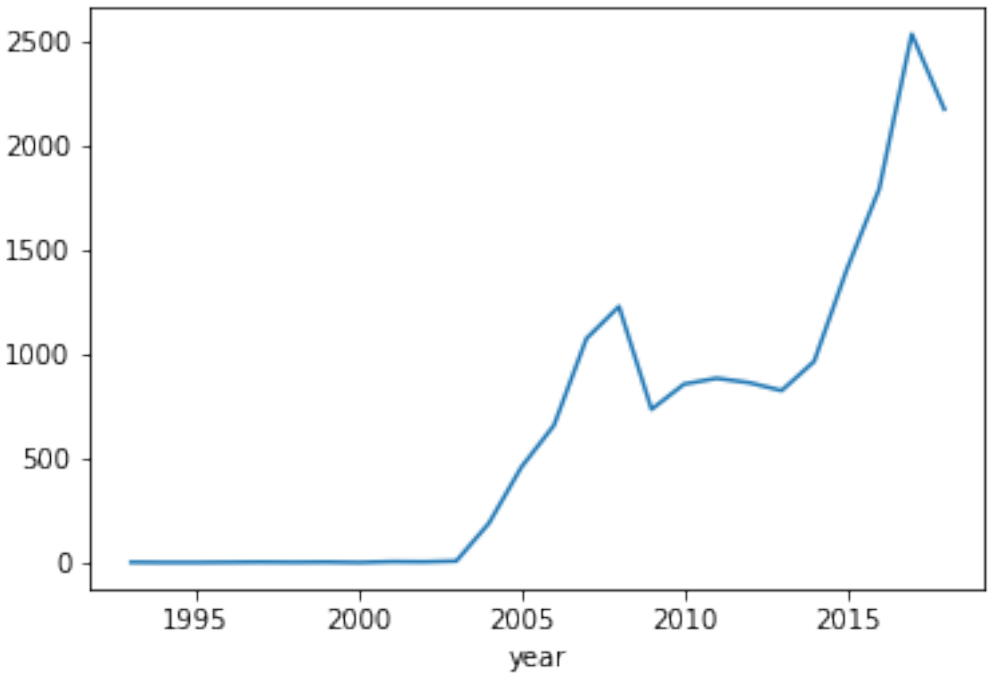
Checking if most of the Western district vacant buildings fall under council District 9

```
In [20]: df_vacant[df_vacant['CouncilDistrict'] == 9].groupby('PoliceDistrict').count()['Lot']

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x17fd5ffe438>
```



Not only Most of the western district falls under district 9, it has southern and southwestern incuded in it.

```
In [23]: %matplotlib inline
         df_date.groupby(['year']).count()['Day'].plot(kind='line')

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x17fd5be4f98>
```



Above plot is vacant buildings vs year of the notice date. The above plot shows us the number of vacant buildings with respect to the year in notice period. There are some homes which are vacant from 2004. That means there are many buildings which are left vacant from a decade. This may show us that if a builing is vacant from many years probably its going to stay that way for long time.

Now lets divide the time frame in to two buckets of equal span and see how many vacant buildings fall under each bucket.

```
In [24]: count = 0
         for i in range(0,len(df_date)):
             if df_date.iloc[i]['year']<=2011:
                 #print(df_date.iloc[i])
                 count+=1
         print(count)
```

```
6121
```

Out of the total 16.6K vacant buildings, almost 40% (6121) are being vacant from 2011.

1.0.1 Just like Newtons laws of motion if building is vacant it tends to be in vacant state!

```
In [25]: from collections import Counter
         listDist = []
         count = 0
```

```
        for i in range(0,len(df_date)):
            if df_date.iloc[i]['year']<=2004:
                count+=1
                listDist.append(df_vacant.iloc[i]['PoliceDistrict'])
        print(Counter(listDist))
```

```
Counter({'western': 87, 'central': 43, 'southwestern': 34, 'eastern': 24, 'northwestern': 9, '
```

We have previosuly seen that western district has the most number of vacant buildings. Looking at the vacant buildings which are vacant from the year 2004. There are 87 buildings which are vacant in western district, which raises our suspicion. Now lets check other datasets to check if there is any correlation with vacant buildings.

Lets dive into Crime dataset

```
In [29]: df_crime.isnull().sum(axis=0)
```

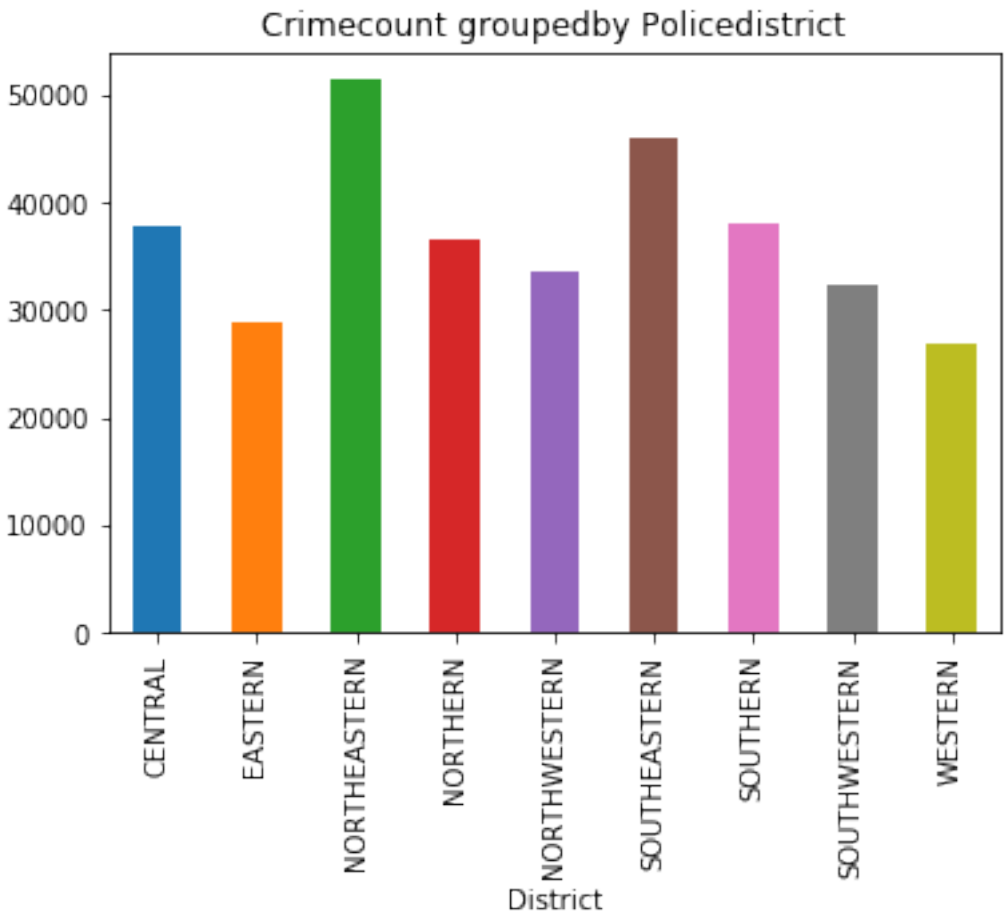
```
Out[29]: CrimeDate              0
         CrimeTime              0
         CrimeCode              0
         Location            3971
         Description          248
         Inside/Outside     25566
         Weapon            261423
         Post                 295
         District             112
         Neighborhood        4213
         Longitude           3969
         Latitude            3969
         Location 1          4755
         Premise            21012
         crimeCaseNumber   335571
         Total Incidents      847
         dtype: int64
```

There are many null values in the dataset. Restricting the cleaning part to required columns only.

Even though Western district has highest number of vacant buildings, it has least crime rate. This goes completely opposite to our intuition. Reason for this has something to do be western district has more number of buildings automatically leading to more number of vacant buildings. To find the truth we must find percentage of vacant buildings in District rather than number of vacant buildings. Since vacant buildings dataset does not contain any information about non vacant buildings, we will use propery tax dataset to these details which contains buildings information. We consider these buildings as non-vacant and calculate percentage of vacant buildings.

```
In [35]: df_crime[df_crime['District'] != 0].groupby(['District']).count()['Location'].plot(ki
```

```
Out[35]: <matplotlib.axes._subplots.AxesSubplot at 0x17fd56b6978>
```



The Crime incidents in each police district is counted and plot for police district vs its count is created. From the above plot we can see following insights. 1. Western and Easter districts havel east number of crime incidents. 2. Northeastern has highest number of crime incidents.

Now let's check an Property taxes dataset The reason why we wanted to use property dataset for the following 1) we can find non vacant buildings 2) we can find number of buildings in each police district 3) Can a build a model which will learn if a building is vacant or not This is done by considering the buildings which are present in property tax dataset and not presnt in vacant building dataset as Non vacant buildings.

```
In [38]: df_property.isnull().sum(axis=0)

Out[38]: PropertyID          0
         Block               0
         Lot                 0
         Ward                0
         Sect                0
         PropertyAddress     5
         LotSize             0
```

```
CityTax            18976
StateTax           18778
ResCode                0
AmountDue          98686
AsOfDate               0
Neighborhood       16122
PoliceDistrict     16122
CouncilDistrict    16094
Location           16082
dtype: int64
```

Since only thing that is important for us is PropertyAddress, check for null values in that column and try to join with address in vacant buildings dataset.

```
In [49]: df_property.groupby(['PoliceDistrict']).count()['Ward']
```

```
Out[49]: PoliceDistrict
         0               16121
         Central          8938
         Eastern         19539
         Northern        30503
         Northwestern    23621
         Notheastern     42261
         Southeastern    27163
         Southern        25251
         Southwestern    25706
         Western         19401
         Name: Ward, dtype: int64
```

Again results are surprising. Number of buildings in Western district is low side(least but one). This means not only there few buildings in western district of baltimore, but most of them are vacant. Previously, we assumed that western district has many buildings is a wrong assumption. So that means there is a less crime rate in vacant neighboorhood.

If buidlings are vacant - No crimes! Voila

```
In [54]: relative_vacancy = {}
         for key in policeDistrict_property:
             if key != 0:
                 relative_vacancy[key] = (policeDistrict_vacant[key.lower()] / policeDistrict_
```

```
In [55]: print(relative_vacancy)
```

```
{'Central': 9.487581114343254, 'Eastern': 18.72664926557142, 'Northern': 2.353866832770547, 'N
```

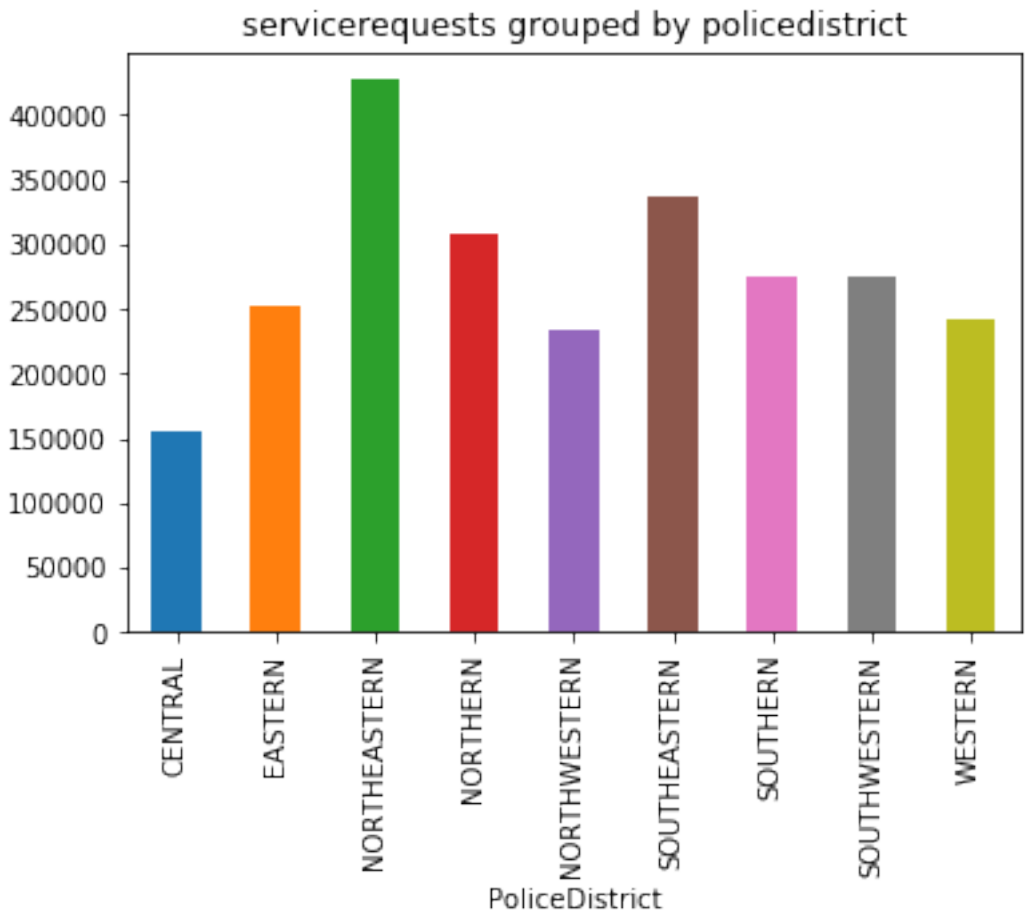
Western leads by large margin followed by Eastern just like in first bar graph. At this point big picture is starting to get clear. Western district has least number of crimes because there are not many non vacant buildings and more vacant buildings which means less population and less crimes.

Let's try to find if there is any relation between 311 requests and vacant buildings.

Customer Service Requests

```
In [65]: df_request.groupby(['PoliceDistrict']).count()['SRType'].plot(kind='bar', title = 'se
```

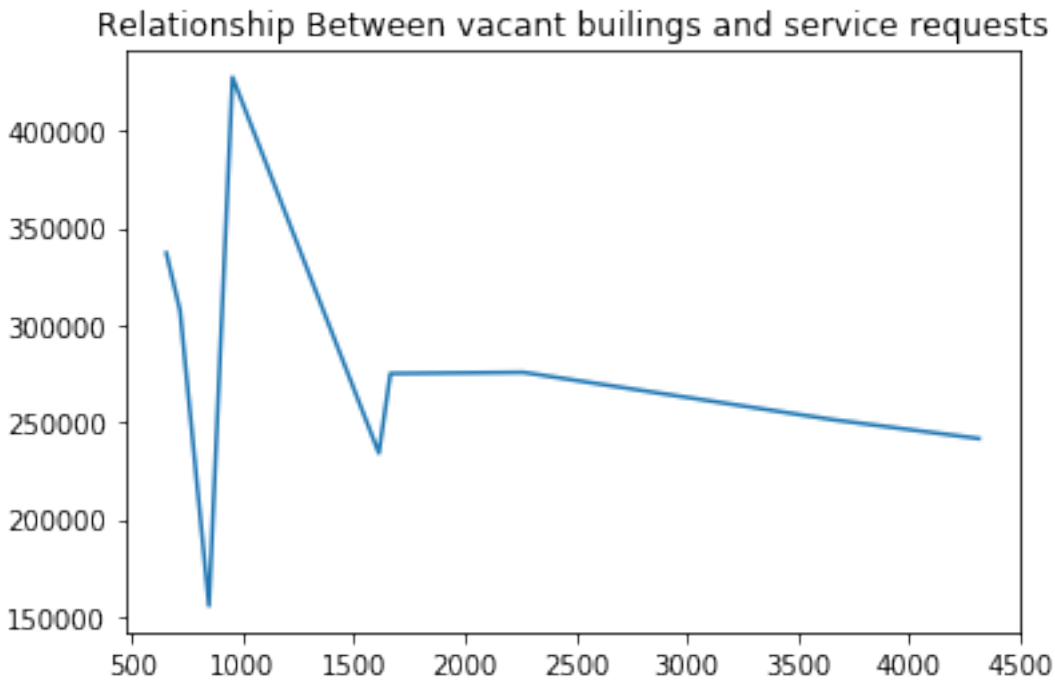
```
Out[65]: <matplotlib.axes._subplots.AxesSubplot at 0x17ff23f5be0>
```



The service requests in each police district is counted and plot for police district vs its count is created. From the above plot we can see following insights. 1. Central and Western districts havel least number of service requests. 2. Northeastern has highest number of service requests.

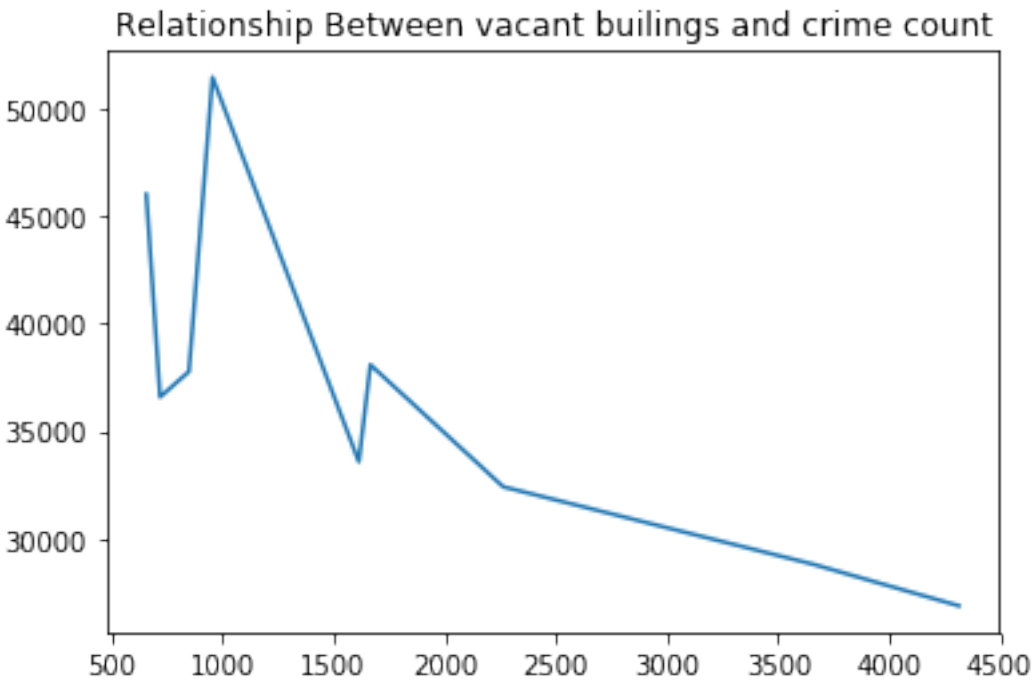
The previously seen crime incidents also follow the same pattern.

```
In [77]: #Now plotting to see if there is any relationship between vacancy and service request

         plot_list = sortMethod(vacant_list, service_request_list)
         plt.plot(plot_list[0], plot_list[1])
         # plt.scatter(plot_list[0], plot_list[1])
         plt.title('Relationship Between vacant builings and service requests')

         plt.show()
```



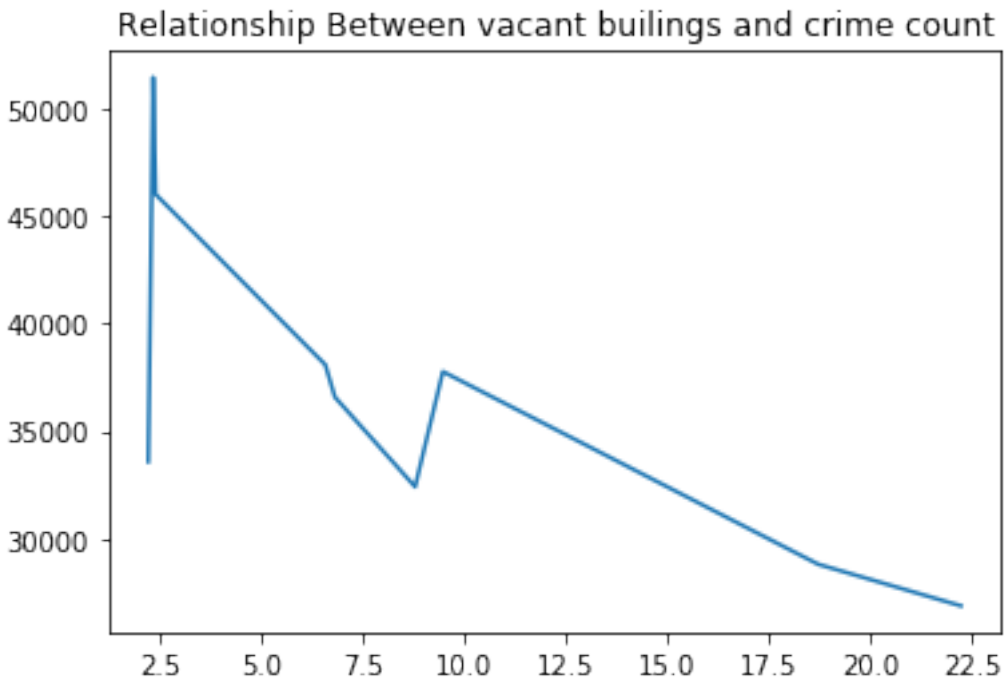
Even though there are few outliers for lower values of vacant builings, the graph in overall is decreasing. This means that with increase in vacant buildings there are less service requests.

```
In [111]: #Plot to see relation between vacancy and crime count
          plot_list = sortMethod(vacant_list, crime_list)
          plt.plot(plot_list[0], plot_list[1])
          # plt.scatter(plot_list[0], plot_list[1])

          plt.title('Relationship Between vacant builings and crime count')

          plt.show()
```

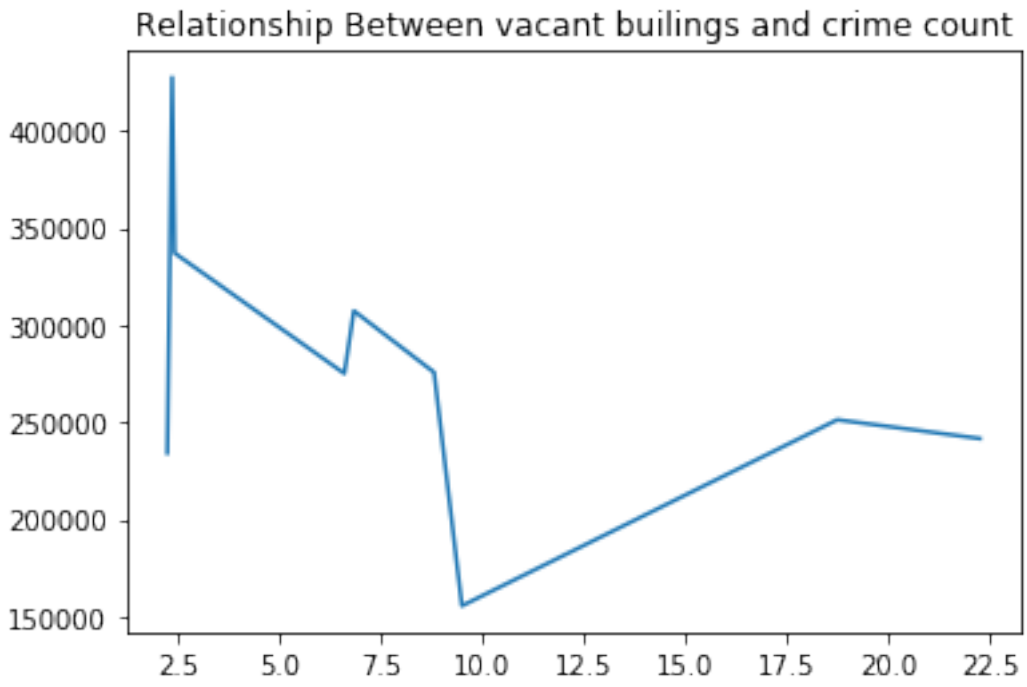


From the above plot we can infer these things 1) Increase in vacancy buildings cause less crime rate in that neighborhood

```
In [79]: #Plot to check if relation holds for relative vacancy and crime count
         plot_list = sortMethod(relative_vacancy_list, crime_list)
         plt.plot(plot_list[0], plot_list[1])
         # plt.scatter(plot_list[0], plot_list[1])

         plt.title('Relationship Between relative vacant builings and crime count')
         plt.show()
```



In [80]:
```
#Plot to check if relation holds for relative vacancy and service requests
plot_list = sortMethod(relative_vacancy_list, service_request_list)
plt.plot(plot_list[0], plot_list[1])
# plt.scatter(plot_list[0], plot_list[1])

plt.title('Relationship Between realative vacant builings and service requests')
plt.show()
```

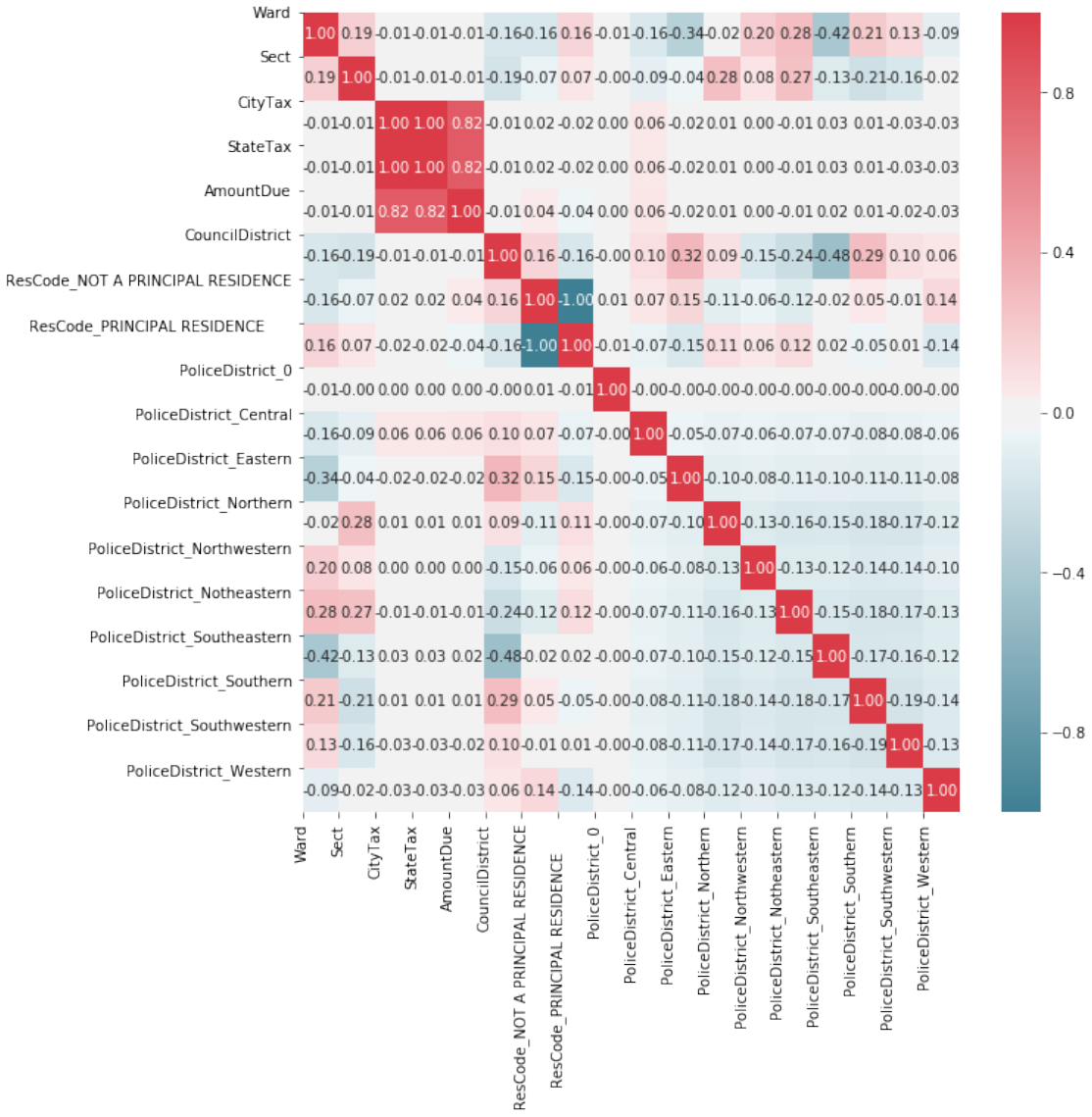


The above plots will strengthen our assumptions because the relative buildings list also follows the same pattern as the vacant buildings list.

1.1 Building Knn and RandomForestClassifier Models

The vacant buildings are marked with a status of 0 and non vacant buildings are marked with a status of 1 which are used as labels for the model.

```
In [104]: corr = df_res.corr()
          fig, ax = plt.subplots(figsize=(10, 10))
          colormap = sns.diverging_palette(220, 10, as_cmap=True)
          sns.heatmap(corr, cmap=colormap, annot=True, fmt=".2f")
          plt.xticks(range(len(corr.columns)), corr.columns);
          plt.yticks(range(len(corr.columns)), corr.columns)
          plt.show()
```



The above heat map shows corelation between two columns. The only attributes with good corelation are - City tax - State tax - Amount Due

```
In [94]: from  sklearn.ensemble import RandomForestClassifier
         RFC = RandomForestClassifier()
         RFmodel = RFC.fit(TrainData,TrainLabel)
         AccuracyTrain=RFmodel.score(TrainData,TrainLabel)
         AccuracyTest=RFmodel.score(TestData,TestLabel)
         print("Accuracy for Vacant train Data :",AccuracyTrain*100)
         print("Accuracy for vacant test data :",AccuracyTest*100)
         f = list(RFmodel.feature_importances_)
```

```
Accuracy for Vacant train Data : 97.72538351091967
Accuracy for vacant test data : 95.28487229862476
```

```
In [ ]: from sklearn.neighbors import KNeighborsClassifier
        KnnModel = KNeighborsClassifier()
        KnnModel.fit(TrainData, TrainLabel)
        KnnTrainAccuracy = KnnModel.score(TrainData, TrainLabel)
        print("Accuracy of HR Training Data for KNN Algorithm is: ",KnnTrainAccuracy*100)
        KnnTestAccuracy = KnnModel.score(TestData, TestLabel)
        print("Accuracy of HR Testing Data for KNN Algorithm is: ",KnnTestAccuracy*100)
```

- The dataset when trained with Random forest classifer has a accuracy of 97% on the validation data and 95% on the test data
- When trained with Knn the acccuracies are about 95% for both the cases

```
In [98]: for i in range(len(columnsList)):
             print(columnsList[i] + " ---> " + str(importanceList[i]))
```

```
Ward ---> 4.322874588473496
Sect ---> 7.7262465263923525
CityTax ---> 24.33784019062489
StateTax ---> 30.611299282635635
AmountDue ---> 20.412379331688733
CouncilDistrict ---> 3.3071283552871025
ResCode_NOT A PRINCIPAL RESIDENCE ---> 2.830741742828997
ResCode_PRINCIPAL RESIDENCE       ---> 2.5885429379510567
PoliceDistrict_0 ---> 2.0928479339436864e-06
PoliceDistrict_Central ---> 0.21666048925451747
PoliceDistrict_Eastern ---> 0.19729841819597554
PoliceDistrict_Northern ---> 0.3682500025443614
PoliceDistrict_Northwestern ---> 0.23161108176382147
PoliceDistrict_Notheastern ---> 0.19345876641263712
PoliceDistrict_Southeastern ---> 0.22161454101800837
PoliceDistrict_Southern ---> 0.44611462284803327
PoliceDistrict_Southwestern ---> 0.3504654958147314
PoliceDistrict_Western ---> 1.6374715334177157
```

Here we have checked the feature importance for each column in the dataset and the inferred these 1) City tax, state tax, amount due has a major impact in classifying the data 2) Policedistrict columns is not so useful in classifying 3) Principal residence did not have much impact on classifying the data

2 Conlcusion and Future Work

Thus we reasoned why buildings are vacant and saw how each factor is effecting building vacancy. The following are the relations that we saw that are effecting vacancy. - With increase in vacant buildings, we see few service requests in that neighbourhood - With increase in vacant buildings, we see less crime rates in that neighbourhood

We can also try to find relation interms of income earned neighbourhood wise, if we find dataset that matches with baltimore addresses. We also made attemps to find wheather crime took place in the vacant builings or not by using Haversine formula. But we came to know that calculating distances for entire dataset is intractable and computationally very expensive.