Checkpoint 4 - Machine Learning

Team Name: The Wild Blazers

Team Members: Anish Bhardwaj

Model Goals

1. Determine an officer's potential for risk spread across different bands (upto 5 bands) based on the information in data_officer combined with district information for each district. The potential for such a system is to recommend officers for sensitivity training based on their and their district's information in case they seem to exhibit high risk.

2. Run K Means Clustering on officer data to see what factors seem to be important and influence officers being grouped together. This will help see if there are some standout characteristics that seem to divide officers in different groups.

SQL and Data

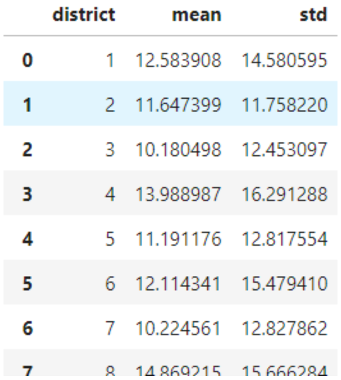
I used the data from the previous Checkpoint (d10re_1.csv and d5.csv in CSVs and SQL) and the data queried from the SQL command in the CSVs and SQL folder (data_officer.sql) to generate a CSV called officers. Officers contains the following columns (district, gender, race, major_award_count, allegation_count, sustained_count, unsustained_count, birth_year) about each officer.

Data Cleaning

To Clean this data and to make sure that I had all the bands setup for the classification process, I used the Data_Cleaning.ipynb jupyter notebook. In this notebook which is linked here, I first imported all three of the CSVs mentioned above. D10re_1 has a lot of data that includes officer demographics (gender and race), population demographics (race), allegations per capita, allegations per police officer, etc. all segmented by districts. D5.csv has data that is essentially the one_hot_encoded data of the police and population demographics in terms of which is the majority for that district. Asian and Native american columns are not present in this as they are not the majority for either police race or population race across any district in Chicago. The data from the sql command is loaded to officers.csv.

The first thing I did was join the demographic one_hot_encoded data with the officers data on the district column. I then extracted just the allegations_per_officer for each district from d10re_1.csv and joined this with the new modified officers data to create a dataframe called officersnew2. I made sure to drop any rows with null values present (only 2 rows were there). I then took only officers who were born after 1955 to make sure the data was actually relevant to the process.

I then calculated the mean and standard deviation of allegations per district so as to get the metrics with which I could create the bands.



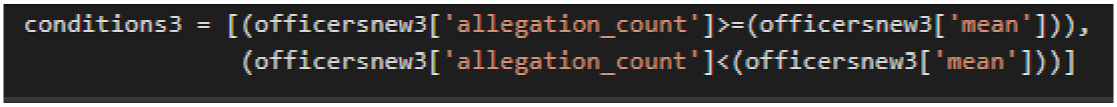
	district	mean	std
0	1	12.583908	14.580595
1	2	11.647399	11.758220
2	3	10.180498	12.453097
3	4	13.988987	16.291288
4	5	11.191176	12.817554
5	6	12.114341	15.479410
6	7	10.224561	12.827862
7	8	14.869215	15.666284

After calculating all of these, I joined this with the dataframe with all the officer information. I then got to work creating the different bands in which the officers would be divided. My original plan was to simply move the exact number away for each standard deviation but the distributions were skewed (skewed left) in such a way that this was not possible. As a result, I worked around this by playing around with different splits such that I could divide the officers by risk in a manner that made sense.

The conditions I used for the different band splits and the Graphs of the distributions can be seen below.

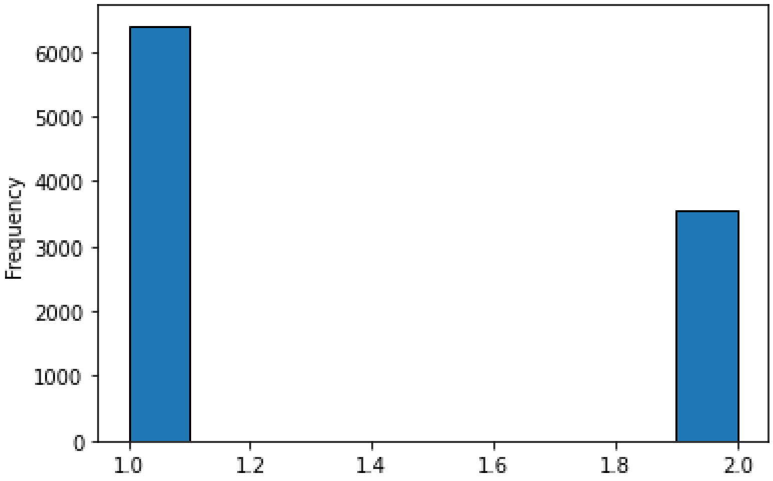
2-Class Bands (1 - Low Risk, 2 - High Risk)

Conditions



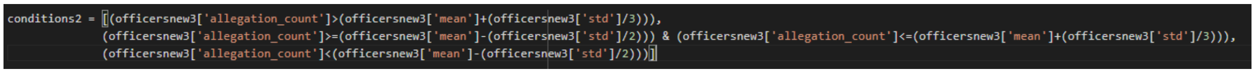
```
conditions3 = [(officersnew3['allegation_count']>=(officersnew3['mean'])),
               (officersnew3['allegation_count']<(officersnew3['mean']))]
```

Distribution



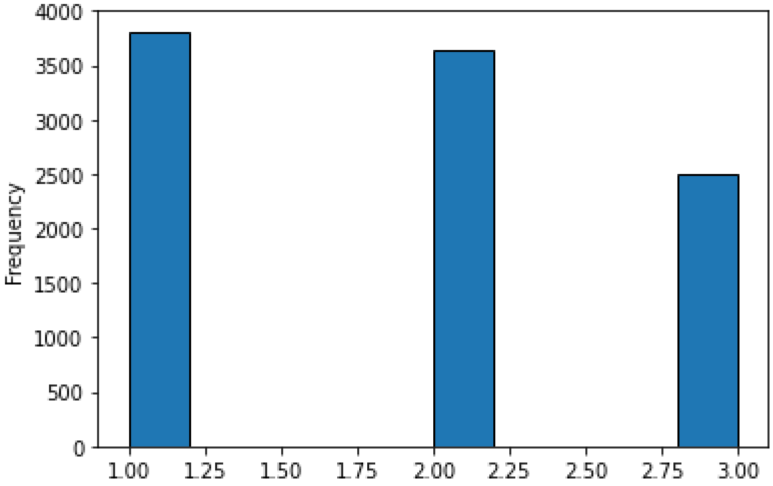
3-Class Bands (1 - Low Risk, 2 - Medium Risk, 3 - High Risk)

Conditions



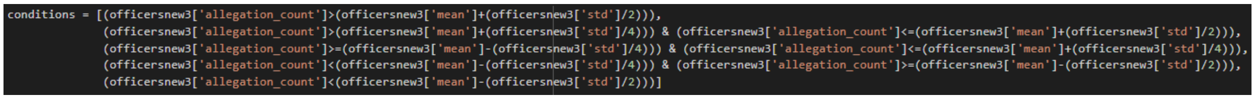
```
conditions2 = [(officersnew3['allegation_count']>(officersnew3['mean']+(officersnew3['std']/3))),
               (officersnew3['allegation_count']>=(officersnew3['mean']-(officersnew3['std']/2))) & (officersnew3['allegation_count']<=(officersnew3['mean']+(officersnew3['std']/3))),
               (officersnew3['allegation_count']<(officersnew3['mean']-(officersnew3['std']/2)))]
```

Distribution



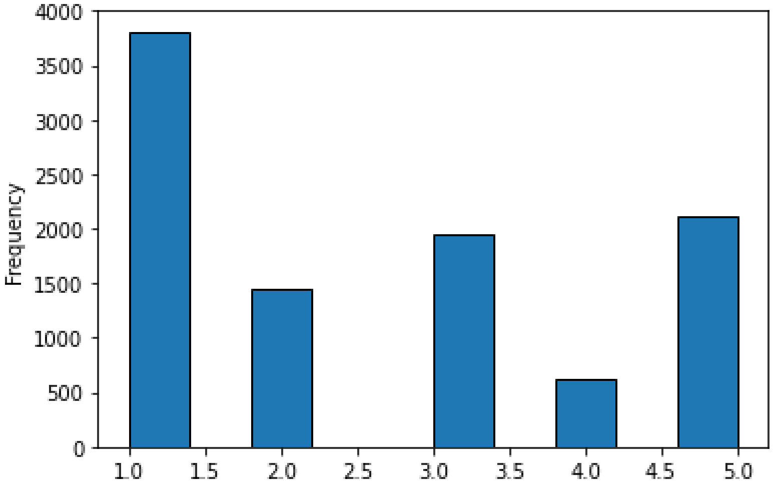
5-Class Bands (1 - Very Low Risk, 2 - Low Risk, 3 - Medium Risk, 4 - High Risk, 5 - Very High Risk)

Conditions



```
conditions = [(officersnew3['allegation_count']>(officersnew3['mean']+(officersnew3['std']/2))),
              (officersnew3['allegation_count']>(officersnew3['mean']+(officersnew3['std']/4))) & (officersnew3['allegation_count']<=(officersnew3['mean']+(officersnew3['std']/2))),
              (officersnew3['allegation_count']>=(officersnew3['mean']-(officersnew3['std']/4))) & (officersnew3['allegation_count']<=(officersnew3['mean']+(officersnew3['std']/4))),
              (officersnew3['allegation_count']<(officersnew3['mean']-(officersnew3['std']/4))) & (officersnew3['allegation_count']>=(officersnew3['mean']-(officersnew3['std']/2))),
              (officersnew3['allegation_count']<(officersnew3['mean']-(officersnew3['std']/2)))]
```

Distribution



Following each band creation, I created columns that corresponded to these bands. For 5 bands, the column name was y, for 3 bands, y2, and for 2 bands, y3

I then downloaded the final CSV that I made for the dataframe. It is called cleaneddata_2.csv.

Supervised Machine Learning

Goal: Determine an officer's potential for risk spread across different bands (upto 5 bands) based on the information in data_officer combined with district information for each district. The potential for such a system is to recommend officers for sensitivity training based on their and their district's information in case they seem to exhibit high risk.

Now that I had cleaned the data and created these different bands, to accomplish this goal, I would have to train a few models with the data (district, gender, race, allegation_count, sustained_count, unsustained_count, birth_year, black, white, hispanic, blackpop, whitepop, hispanicpop) corresponding to the officers with the band values as the target labels. The code for this can be seen by opening Models_for_classification.ipynb or by clicking here.

I start in this notebook by first uploading the cleaneddata_2.csv file as a dataframe. I then dropped the unnecessary columns from the cleaned data (std, mean, etc.) leaving only the columns mentioned above. I then iterated through the different target columns (y, y2, and y3) for models on different bands. Considering this was a classification task, I chose the DecisionTree, KNeighbours, and Logistic Regression Classifiers for training the models.

Decision Trees are a good choice for classification tasks because they handle both categorical and continuous data. They are also quite easy to interpret, and are capable of capturing complex decision boundaries. K-Nearest Neighbors is another algorithm that can be used for classification. It works well with noisy data and can also capture complex decision boundaries. Finally, Logistic Regression is another great algorithm that does well with simple classification problems but doesn't do well with large numbers of classes. It can also handle both categorical and continuous data easily. Each of these classifiers has different strengths and weaknesses, which makes them suitable for testing whether a model can be fit over our data which is why I chose these to determine the ability to classify officers in different risk bands.

To test each of these models on the different datasets, I also used a library called cross-validate. This library allowed me to perform K-fold validation which essentially splits the data into K parts (in this case 5) and iterates over taking 4 parts for training and 1 part for testing with each part getting a chance to be a validation part. This is a great way to test a models potential as it ensures that the model isn't just performing well for a specific set of data.

The results of training these models across the different number of bands can be seen in the table below. The best performing model has its accuracy in bold for each set of bands.

	Decision Tree Classifier	K Neighbours Classifier	Logistic Regression Classifier
2 Bands	94.93%	94.62%	**95.44%**
3 Bands	89.60%	88.50%	**90.50%**
5 Bands	**81.42%**	79.22%	77.08%

As can be seen in the table above, we can get quite good accuracy across the board even with 5 bands. The best performing models for both 2 and 3 bands was the logistic regression classifier. This was expected because Logistic Regression performs very well for a small number of classes. When the number of bands was increased to 5, the Logistic Regression Classifier did not perform as well but the Decision tree classifier performed very well.

Final Thoughts with Goal 1

This was a very interesting task and showed that given officer data, we can create bands that determine, to a reasonably high degree, even with 5 bands the risk level of an officer. The higher the risk of an officer is the greater the chance is for the officer to have allegations against them and thus officers with a high band can be recommended sensitivity training that would help them become better officers with a less chance of being biased.

Unsupervised Machine Learning

Goal: Run K Means Clustering on officer data to see what factors seem to be important and influence officers being grouped together. This will help see if there are some standout characteristics that seem to divide officers in different groups.

For this Goal, I wanted to take the same officer data I already had and then, using different factors (different data columns) across 2 clustering scenarios, see what features stood out in terms of importance for them. I changed the dataframe from the first to the second scenario based on the reasoning given below.

Clustering 1: Linked Here

Labels: district, gender, race, major_award_count, allegation_count, sustained_count, unsustained_count, birth_year, black, white, hispanic, blackpop, whitepop, hispanicpop

Reasoning: Simply using all the data that wasn't mean/stdev or class data for Clustering

Clustering 2: Linked Here

Labels: district, gender, race, major_award_count, allegation_count, sustained_count, black, white, hispanic, blackpop, whitepop, hispanicpop

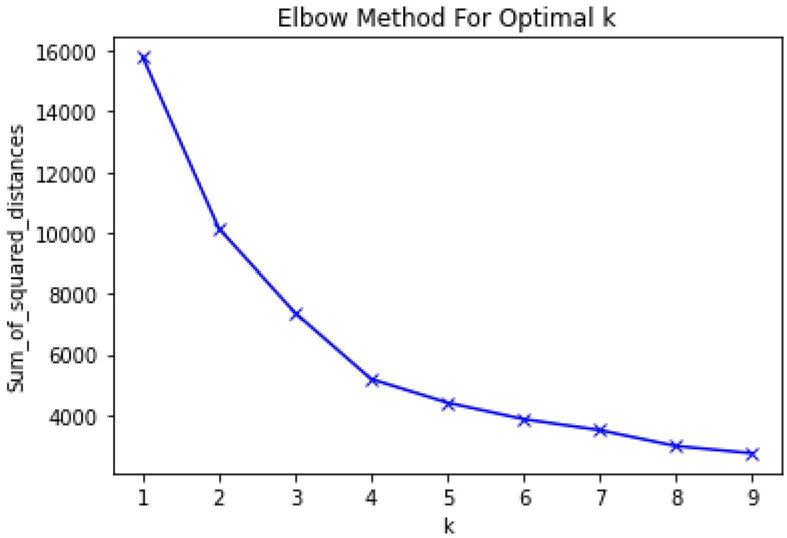
Reasoning: Removing Birth_year and unsustained_count as they skewed the results in a manner that felt like there were no clear insights

For Each of these, I first imported the dataset that we had created when cleaning the data. I then dropped all the columns except those specified here. Then, I followed this by performing one round of clustering. I went from 1 to 10 clusters each time following which I graphed the sum of squared distances. I did this to use the elbow method through which I was able to find the optimal number of clusters each time. I then used the Kmeans feature importance library (Linked Here) which is a wrapper class for the sklearn KMeans system to determine the most important features for each clustering set with the optimal number of clusters.

Within this I used the WCSS method which determines which feature had the maximum impact on the Within-Cluster Sum of Squares which is what KMeans attempts to minimize. I then graphed the 3 most important features across each cluster that had been created and drew inferences from that. Here are the results

Clustering method 1

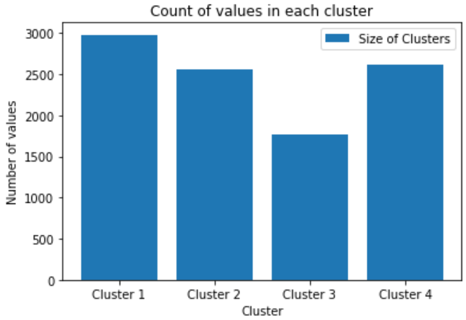
For this clustering set, the elbow mechanism can be seen below.



Based on this, the optimal number of clusters was 4. After running the feature importance wrapper, the following 3 columns were determined to be the most important features for determining the clusters (From highest to lowest impact).

1. Birth Year
2. Allegation_count
3. Unsustained_count

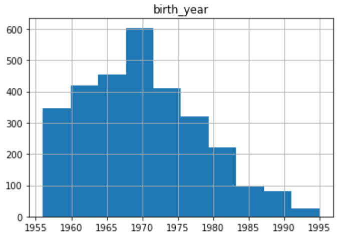
The distribution of clusters can be seen in the graph below



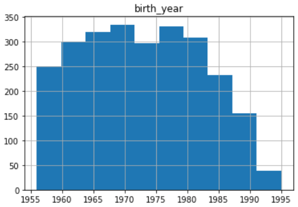
We can see that there is a significant amount of officers in each cluster. I then graphed all the different columns but performed interpretations on just the 3 most important features

Feature 1: Birth Year

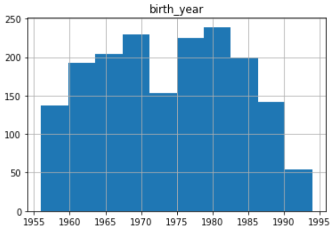
Cluster 1



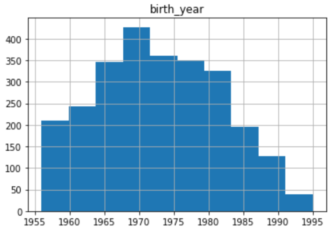
Cluster 2



Cluster 3



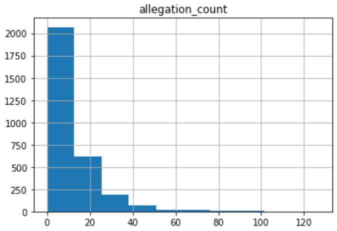
Cluster 4



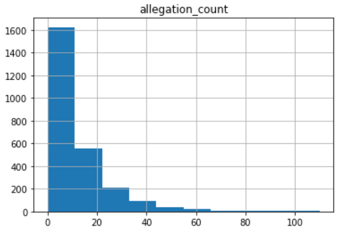
As can be seen by the graphs here, the distribution of birth_year is similar for cluster 2 and cluster 3 and similar for cluster 1 and cluster 4. 1 and 4 are skewed to older individuals and 2 and 3 are skewed towards younger officers. 1 represents the oldest individuals followed in order of oldest to youngest distributions by 4, 2, and 3.

Feature 2: Allegation_count

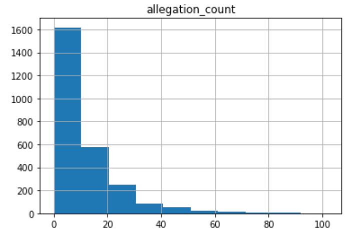
Cluster 1



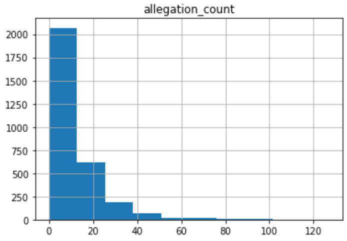
Cluster 2



Cluster 3



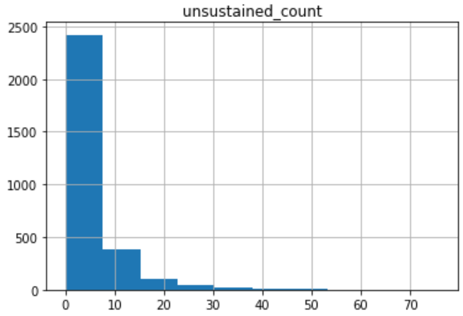
Cluster 4



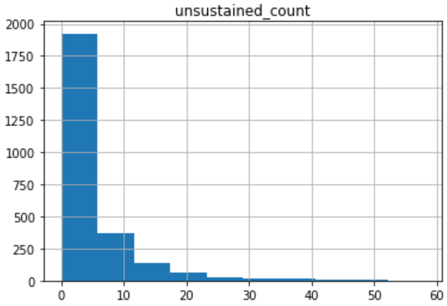
We can see from these plots that the distribution of allegations is very similar across the board showing how heavy of an impact the birthyear had with it nearly controlling the entire distribution. This is why I performed multiple different clustering scenarios to see how and if I could infer information from the clusters.

Feature 3: Unsustained Count

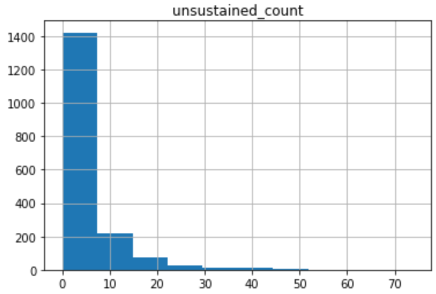
Cluster 1



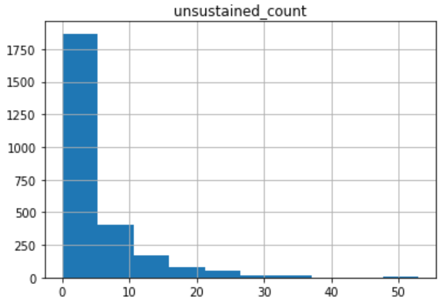
Cluster 2



Cluster 3



Cluster 4

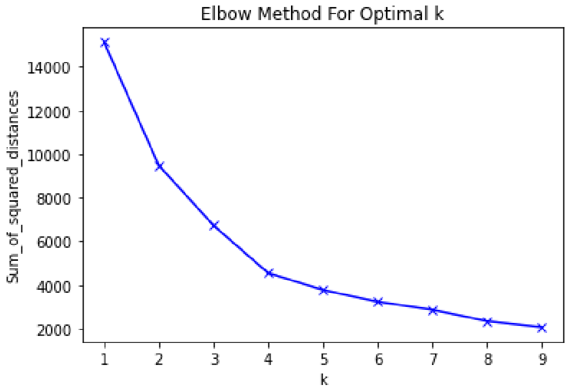


The final feature deemed most important was unsustained count. As can be seen by the graphs here, there is not much inference to be made as the distribution is spread and paks in the 1 to 10 range. However, if we look closer, we see that Cluster 2 and 4 have fewer unsustained counts after 5 counts and cluster 1 and 3 have fewer (the next bar) after 7.5

Based on this clustering mechanism, I was unable to truly get much information. As a result, I decided to perform clustering again but this time omit the birth_year and the unsustained count class.

Clustering 2 (Removed Birth_year and unsustained_count)

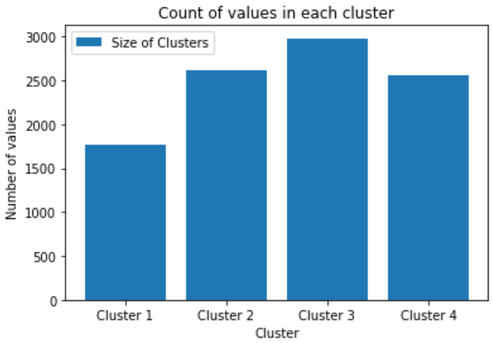
Elbow Method



Number of optimal Clusters: 4

Most important Features: Allegation_count, District, Race

Clustering Distribution:

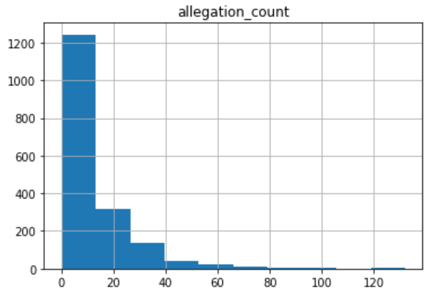


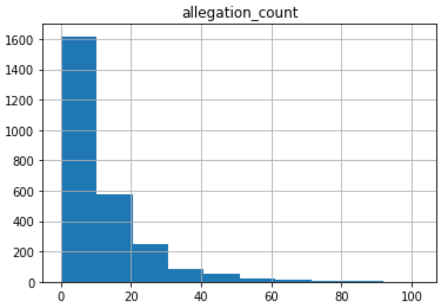
<u>*Important Features:*</u>

Allegation_count:

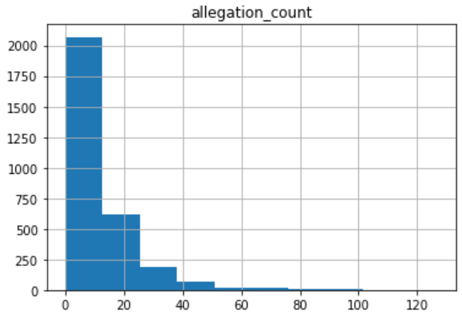
Cluster 1

Cluster 2

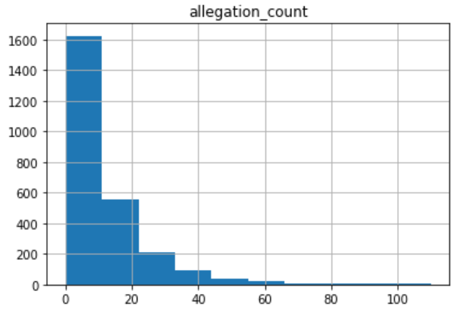




Cluster 3

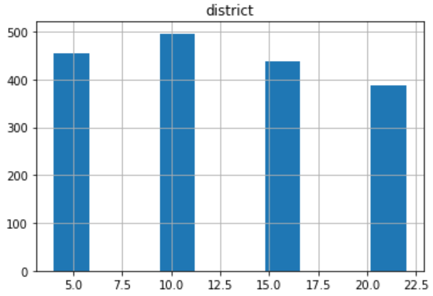


Cluster 4

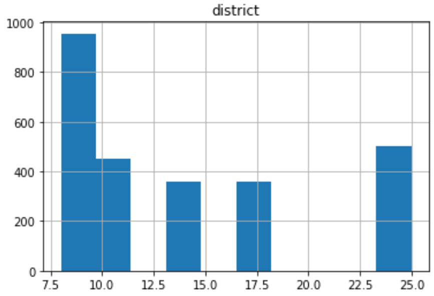


District:

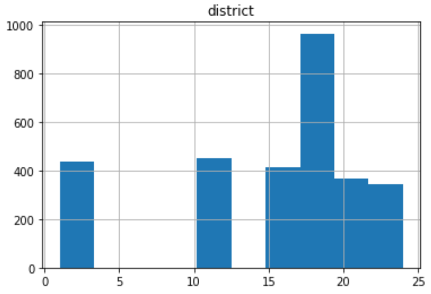
Cluster 1



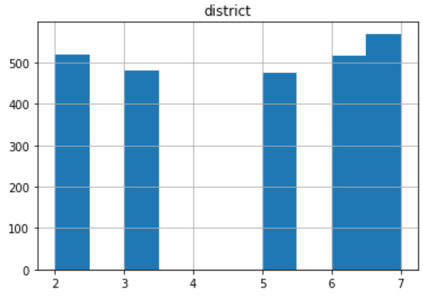
Cluster 2



Cluster 3

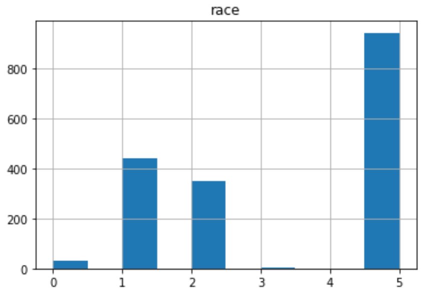


Cluster 4

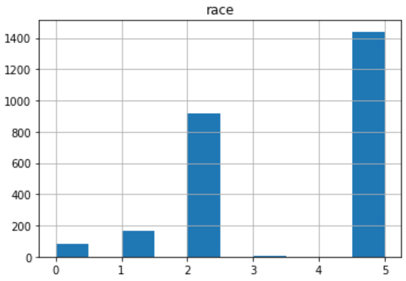


Race:

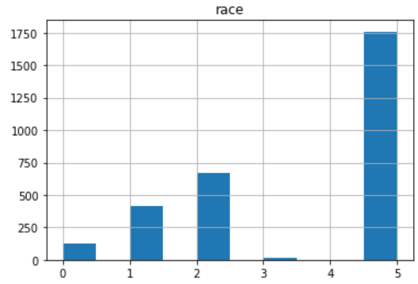
Cluster 1



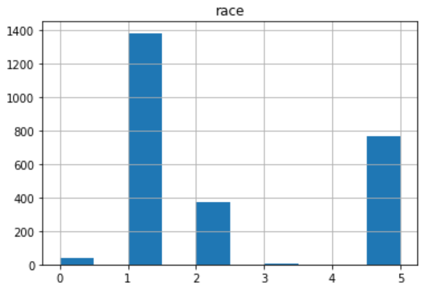
Cluster 2



Cluster 3



Cluster 4

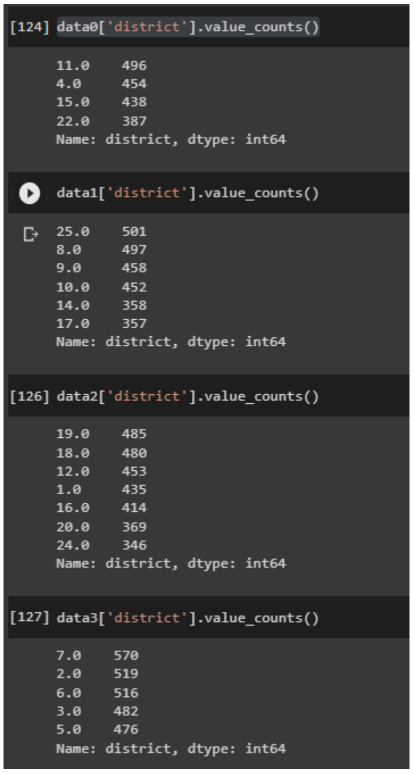


Now we have some interesting insights coming in. When we remove Birth_year and Unsustained_count and look at the 3 most important features, we have some interesting stuff. The allegation counts across clusters seems to be relatively similar. In terms of race, here is how the different values map to different races

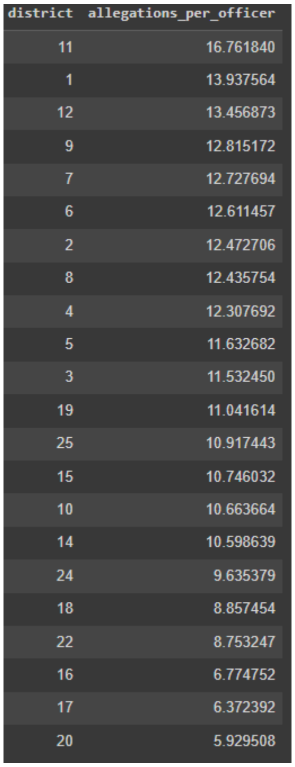
0 - Asian/Pacific, 1 - Black, 2 - Hispanic, 3 - Alaskan Native, 4 - Unknown, 5 - White

This was the final feature deemed most important. As can be seen by the graphs here, there is only one class that really differs entirely from other clusters and that is cluster 4. In cluster 4, there is a larger number of Black officers than any other officers. In the other clusters, we see that cluster 1 has a majority of White officers followed by Black officers, Cluster 2 has a majority of White officers followed by Hispanic officers, and cluster 3 also has a majority of white officers followed by Hispanic officers.

The interesting insights though seem to be in the District information. There is a clear set of districts attributed to each cluster



```
[124] data0['district'].value_counts()

      11.0    496
      4.0     454
      15.0    438
      22.0    387
      Name: district, dtype: int64

      data1['district'].value_counts()

      25.0    501
      8.0     497
      9.0     458
      10.0    452
      14.0    358
      17.0    357
      Name: district, dtype: int64

[126] data2['district'].value_counts()

      19.0    485
      18.0    480
      12.0    453
      1.0     435
      16.0    414
      20.0    369
      24.0    346
      Name: district, dtype: int64

[127] data3['district'].value_counts()

      7.0    570
      2.0    519
      6.0    516
      3.0    482
      5.0    476
      Name: district, dtype: int64
```



district	allegations_per_officer
11	16.761840
1	13.937564
12	13.456873
9	12.815172
7	12.727694
6	12.611457
2	12.472706
8	12.435754
4	12.307692
5	11.632682
3	11.532450
19	11.041614
25	10.917443
15	10.746032
10	10.663664
14	10.598639
24	9.635379
18	8.857454
22	8.753247
16	6.774752
17	6.372392
20	5.929508

We can see from this that there definitely is a clear set of districts attributed to each cluster. Let's look at our data once again to see whether there is some clear reason for this distinction…perhaps based on the attributes of allegations across the districts in each of these?

Well on looking at this further, unfortunately, while it did bin multiple specific districts together and had no overlap, there seems to be no relation to the allegation count. However, all hope is not lost when this is seen, it's just an indication that we still may not have explored the perfect level of labels and clusters to find this out. I still believe that there is scope for showcasing something with this type of clustering as can be seen with the differences across the race based clusters in the second clustering experiment. It does seem to showcase that there is some indication of a reasoning to which allegations, race, and districts are important to clustering. The findings just seem to be inconclusive for me.

Final Thoughts with Machine Learning

- We created some pretty nifty models that can classify officers based on risk perception quite well across even 5 bands of risk. The models showcase potential for creating a Machine Learning based risk perception system which can flag individuals for sensitivity training to have officers that are less prone to allegations against them and thus resulting in officers who better serve our communities
- We tried to perform clustering on two sample datasets modified against the data_officer combined with district data CSV. The findings were slightly inconclusive but do seem to show that allegation rate, district, race of the officer, birth year, and unsustained count may have potential in showcasing some connection across officers

Future Work

- Playing even further with the ML models based on risk and incorporating data that work in specific with violent interactions to better sensitize such officers
- Continuing to work on clustering to see whether some connection can be made across the aforementioned promising data variables
- Working with more variables and hyperparameters across the current created models mentioned here to improve them even further