Socioeconomic Data Analysis - Albany, NY

Anonymous

December 8, 2021

Socioeconomic Data Analysis for Albany, NY

Data set is a compilation of census and SE features, including the location of Free Food Fridges, socioeconomic features of each neighborhood such as median salaries and education levels, demographic features such as racial and generational subpopulations, and as well as WalkScore data on ease of public services in Albany, NY.

1. KNN
2. KMeans
3. PCA
4. Regression

Data Set Geography and Classification Labels

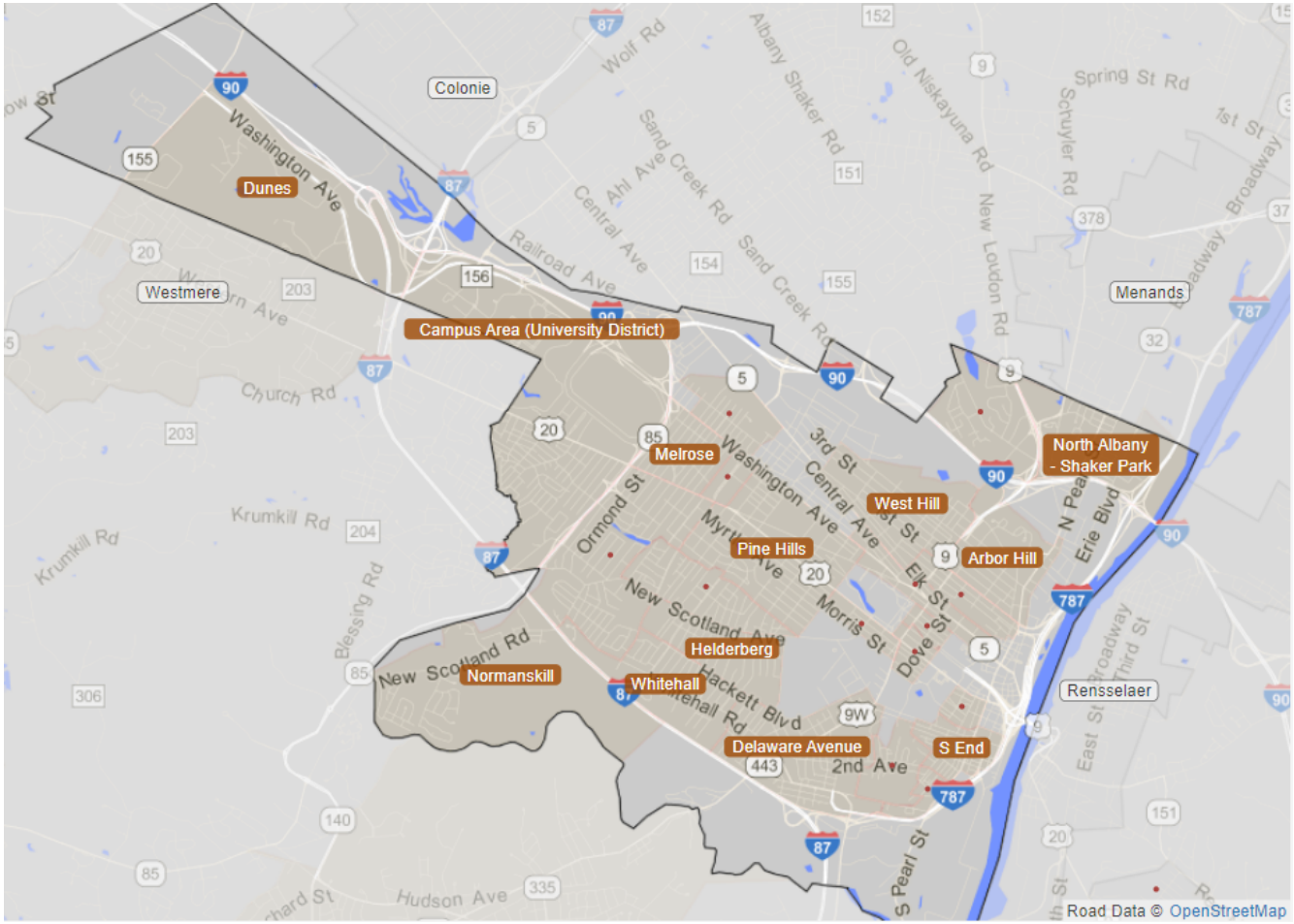


Figure: A sample of neighborhood locations in Albany, NY. In all, 26 had full-feature data (Pastures and West End were missing many data).

Classification Labels - Free Food Fridges in Albany

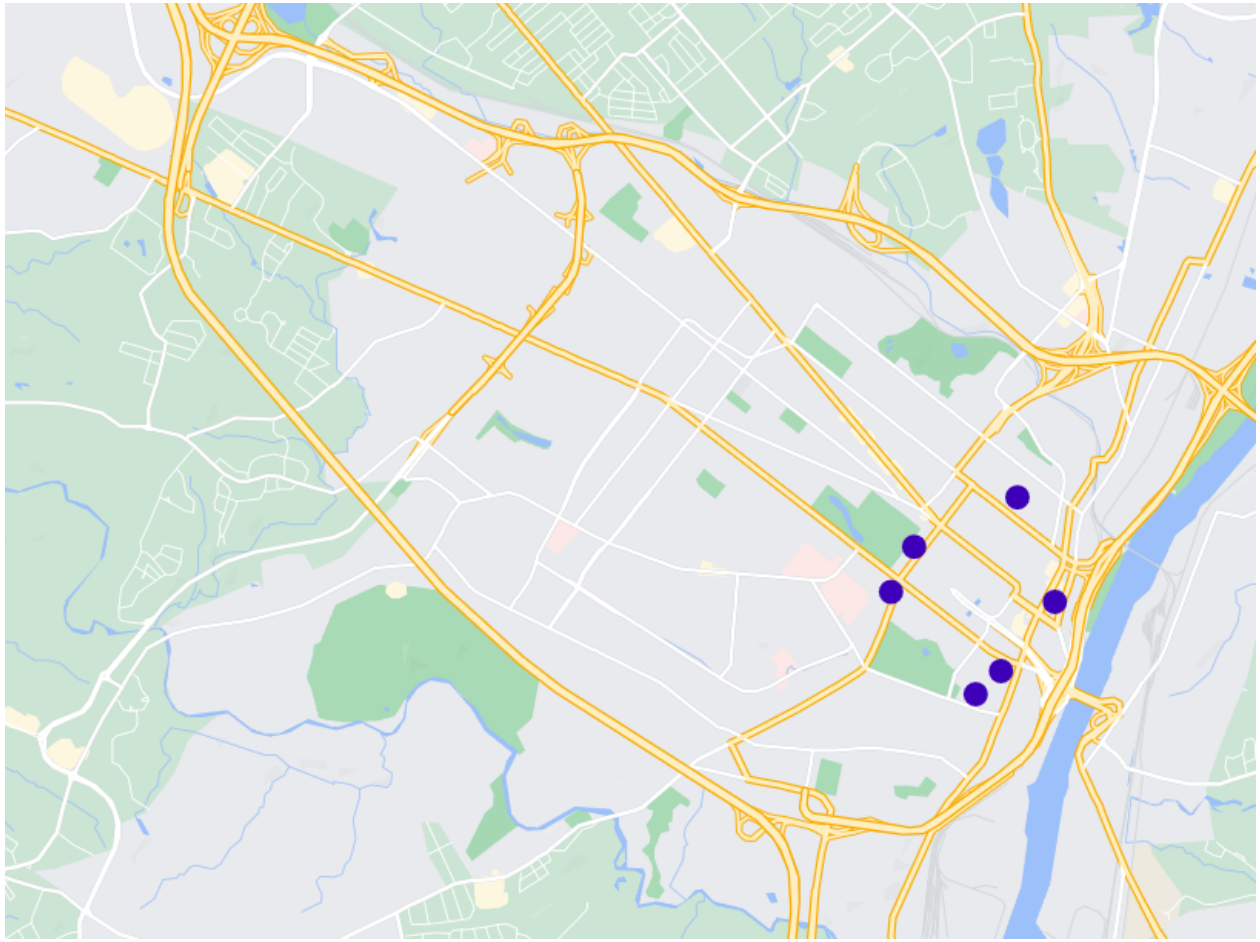


Figure: Given SE and census data, can we recover those features used in determining if a neighborhood is 'impoverished'? (img: `maps.google.com`)

Benchmark and CLT

Over a large number of independent trials n, a classifier which always chooses 'no fridge' will obtain an average accuracy of $p = 21/26 \approx 0.808$. Moreover, for a given significance level α, we obtain a $(1 - \alpha)$-confidence interval

$$p \pm z_\alpha \sqrt{\frac{p(1-p)}{n}}$$

(This guides the following assessment of classifier significance.)

True Classification Accuracy

Problem

Small sample sizes (26 neighborhoods/data points) and imbalanced data (5 fridges) create large variances in classification accuracies.

Example

```
getAvgAccuracyNTrials(['marital_status_widowed'], dfScaled,trials = 1,k_max = 8)
```

```
array([0.5   , 0.333, 0.167, 0.833, 0.833, 0.833, 0.833])
```

```
getAvgAccuracyNTrials(['marital_status_widowed'], dfScaled,trials = 1,k_max = 8)
```

```
array([0.833, 1.   , 0.833, 0.833, 0.5  , 1.   , 0.833])
```

Resampling Process

Solution

Repeat the random train-test split and classification process a large numbers of times (100,000 times, $t \approx 7min$ per k value with one feature).

Example

```
getAvgAccuracyNTrials(['marital_status_widowed'], dfScaled,trials = 100000,k_max = 2)
```

```
array([0.81913358])
```

(all the following accuracies are averages over 10,000 trials, $t \approx 30sec$ per k value with one feature)

Accuracies of Single-Feature Classifiers

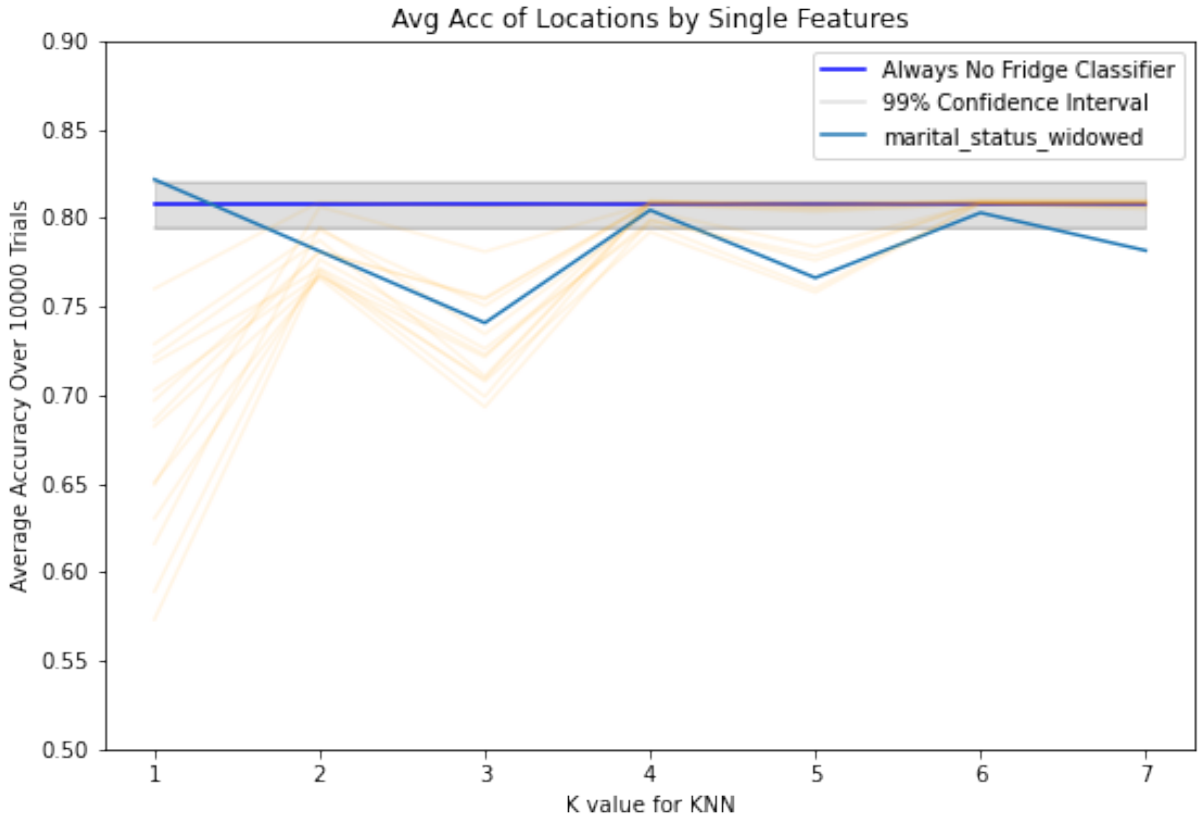
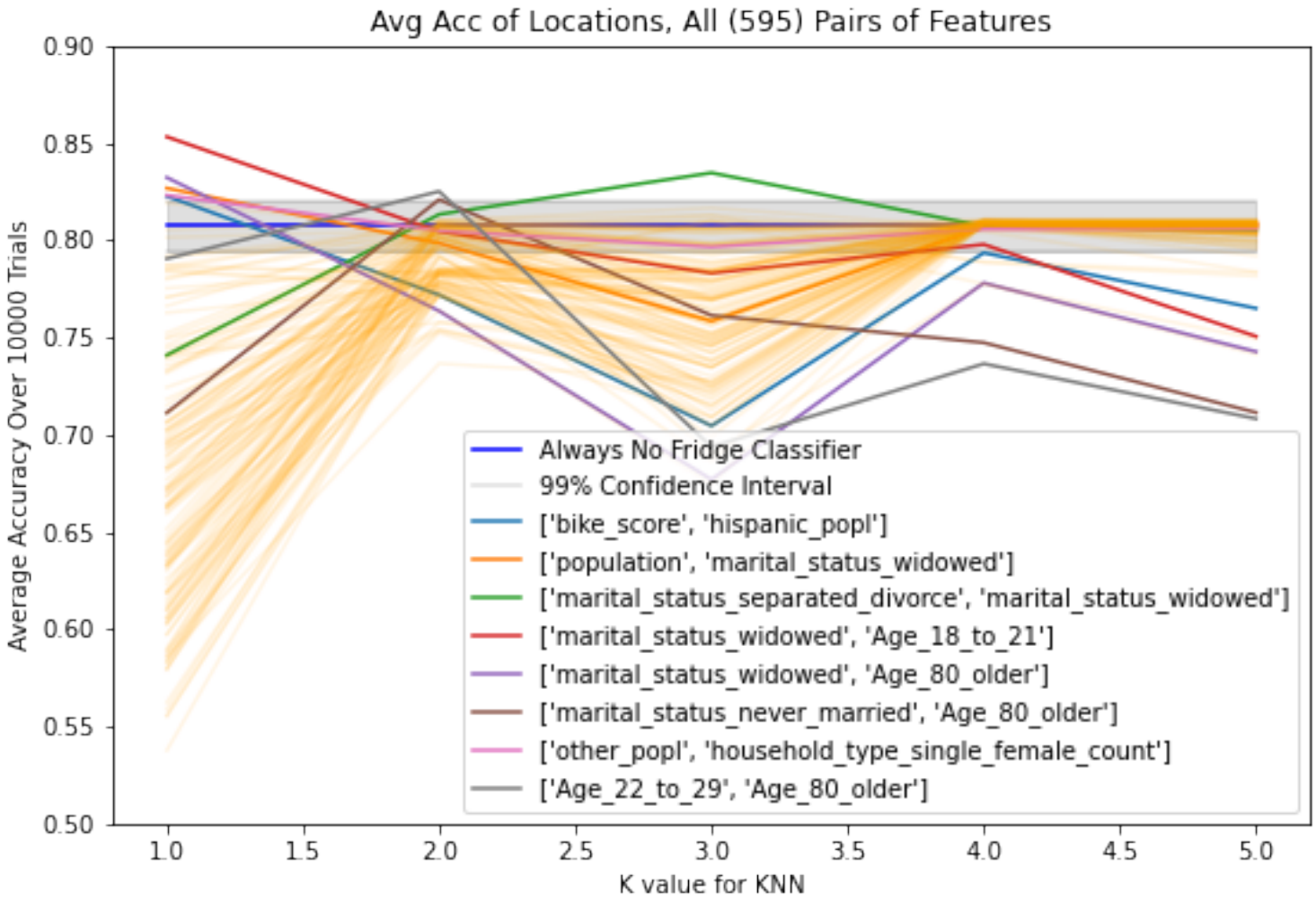
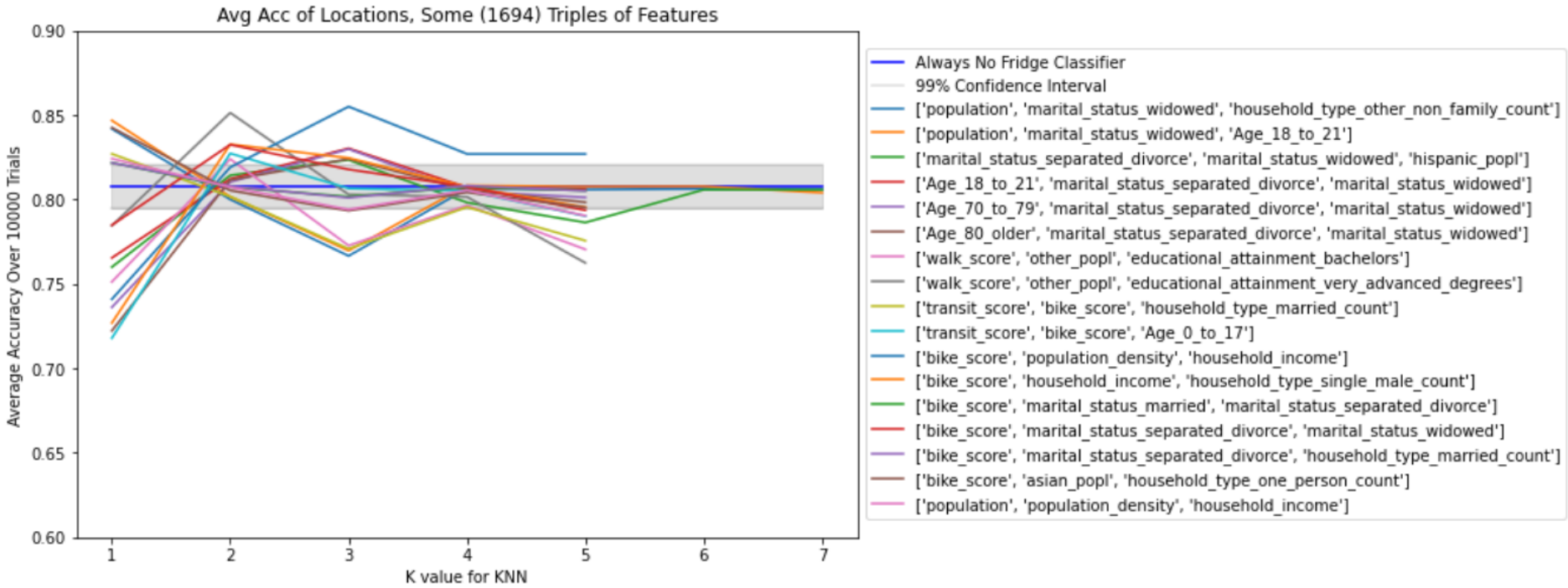


Figure: Solid lines are plots for features which have an accuracy exceeding the 99% CI. Feint yellow lines, a feature with an accuracy within the CI. Many others do not enter or exceed the CI (not shown).

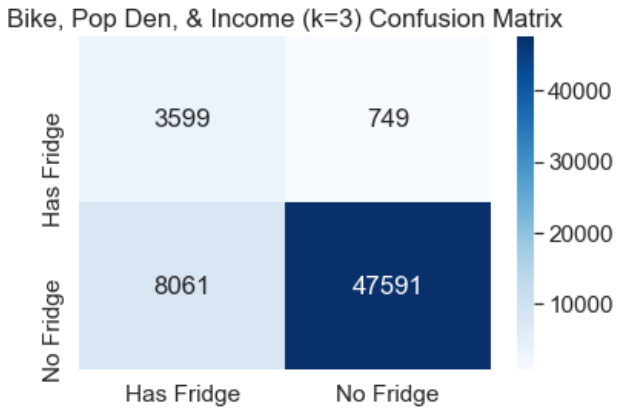
Accuracies of Two-Feature Classifiers

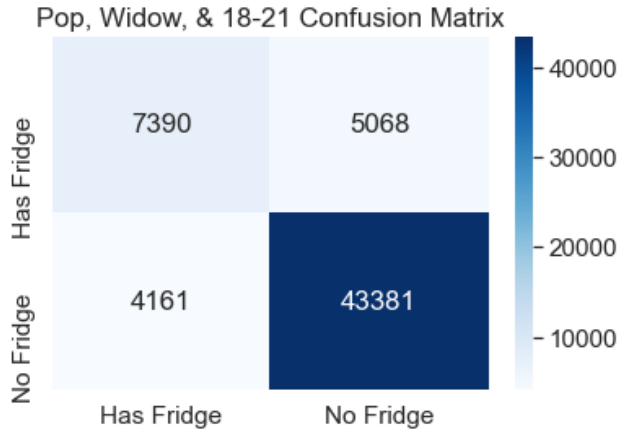


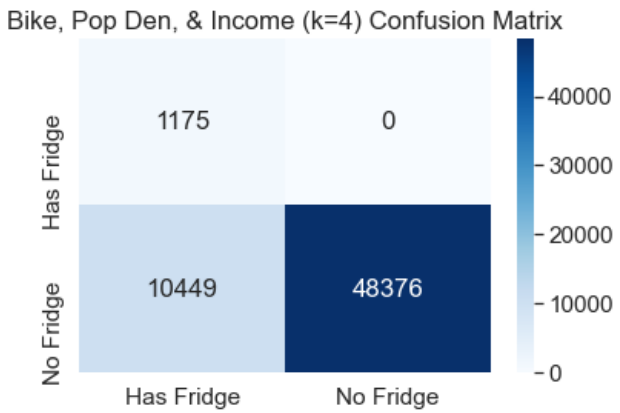
Accuracies of Three-Feature Classifiers



Confusion Matrices







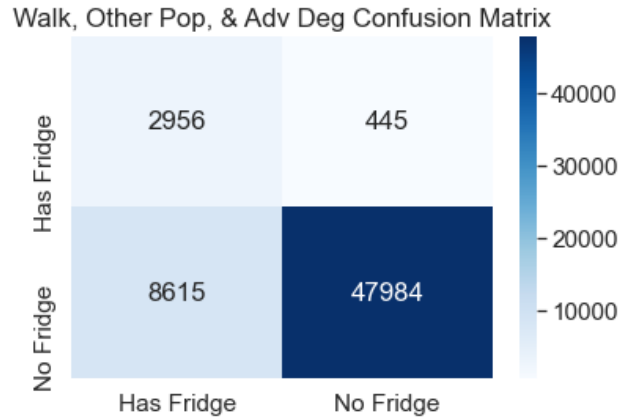


Figure: Confusion matrices for top three triple-feature classifiers.

Thank You!

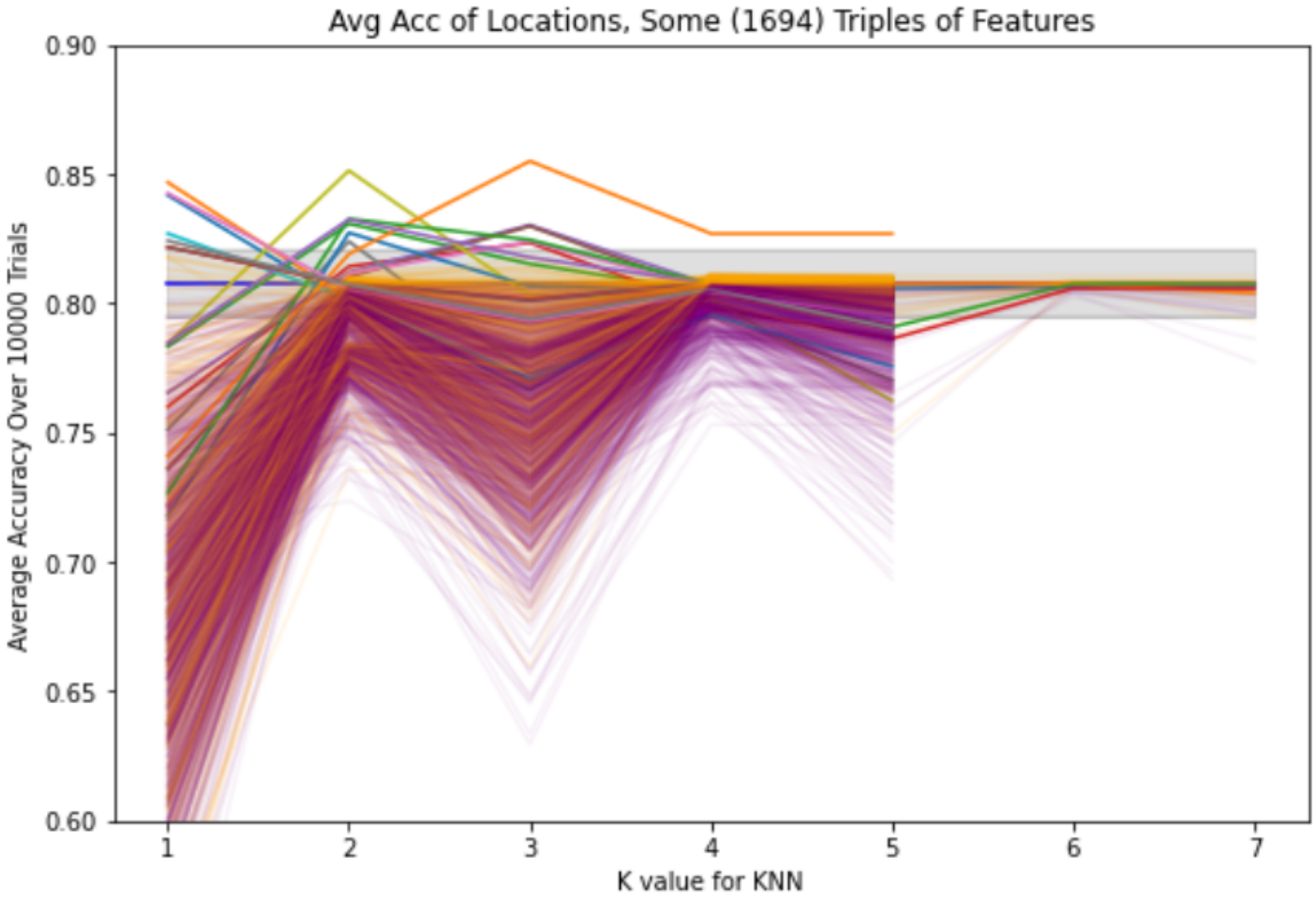


Figure: Future example for Dr. Curry's list of terrible plots. 1694 line plots in one figure :D

K-Means Clustering

Problem

Find desirable neighborhoods in Albany for moving to.

Solution

Cluster neighborhoods with K-Means and select a desirable set.

Data

	nei_final_simple	household_income	transit_score	food_stamps_total	population_density
0	Arbor Hill	28097.25	62.0	1160.0	10976.51
1	Beverwyck	31694.00	62.0	607.0	17353.66
2	Bishops Gate	49526.58	34.0	156.0	2940.25
3	Buckingham Lake/Crestwood	76063.14	44.0	144.0	4639.05
4	Center Square	52087.17	68.0	401.0	27820.03
5	Central Ave	40074.41	66.0	380.0	12472.39

Figure: Selected features from data set for clustering neighborhoods.

Feature Explanation

After inspecting the data set and running initial clusterings, the following features were found to contribute in some way to the desirability of a neighborhood.

1. Household Income
2. Transit Scores
3. Food Stamps
4. Population Density

K-Means Clustering (k = 5)

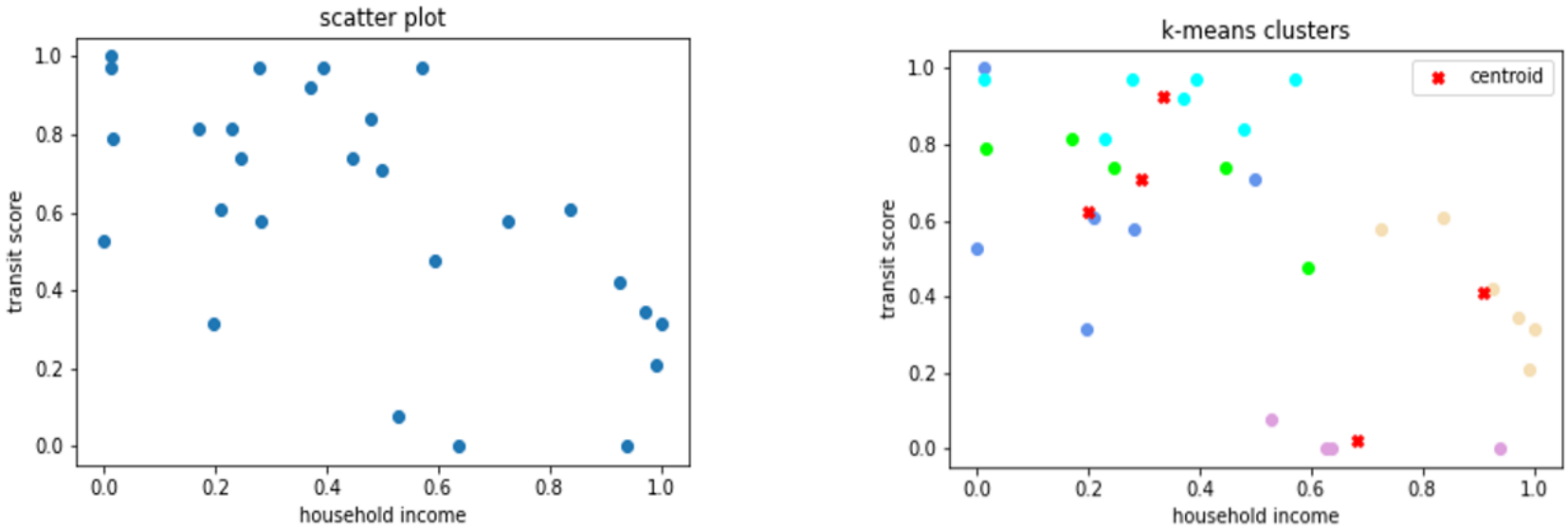


Figure: (Left) Plot of four features projected onto the features Transit Score and Household Income. (Right) The same plot, but with neighborhood data colored by cluster.

Optimal K-Value

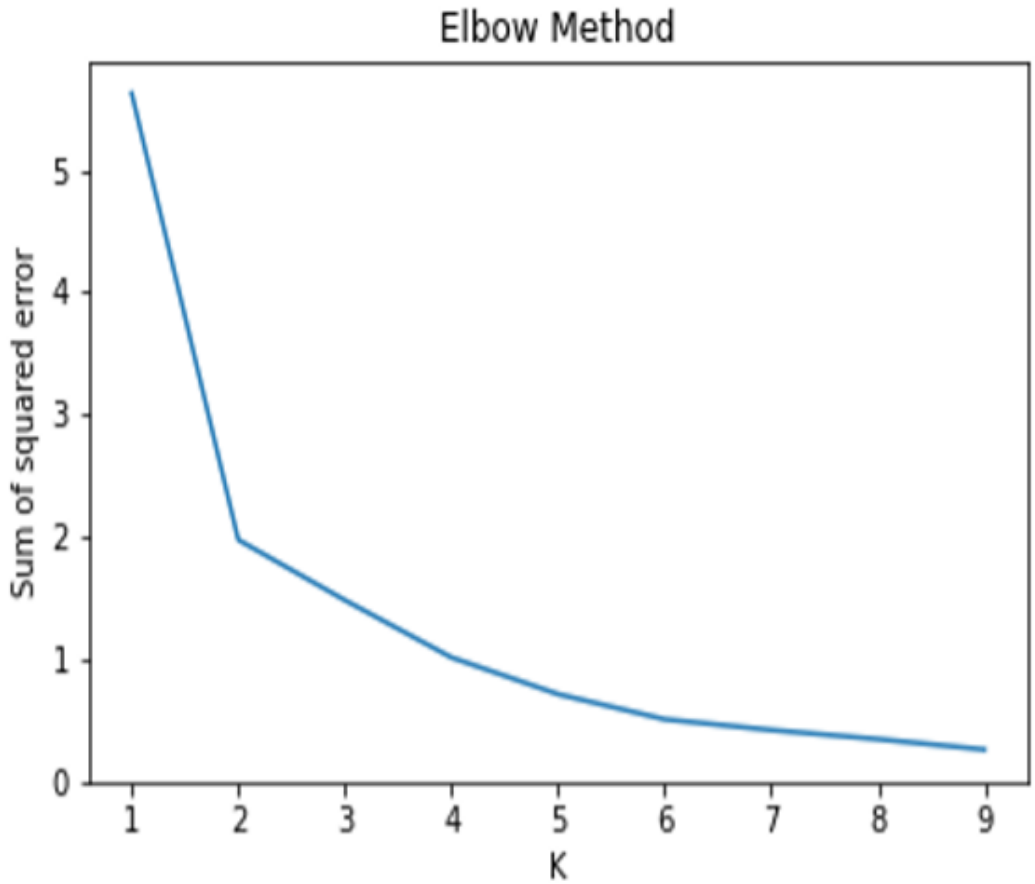


Figure: Plot of SSE as values of k vary. The values beyond the elbow at $k = 2$ provide small improvements.

K-Means Cluster (k = 2)

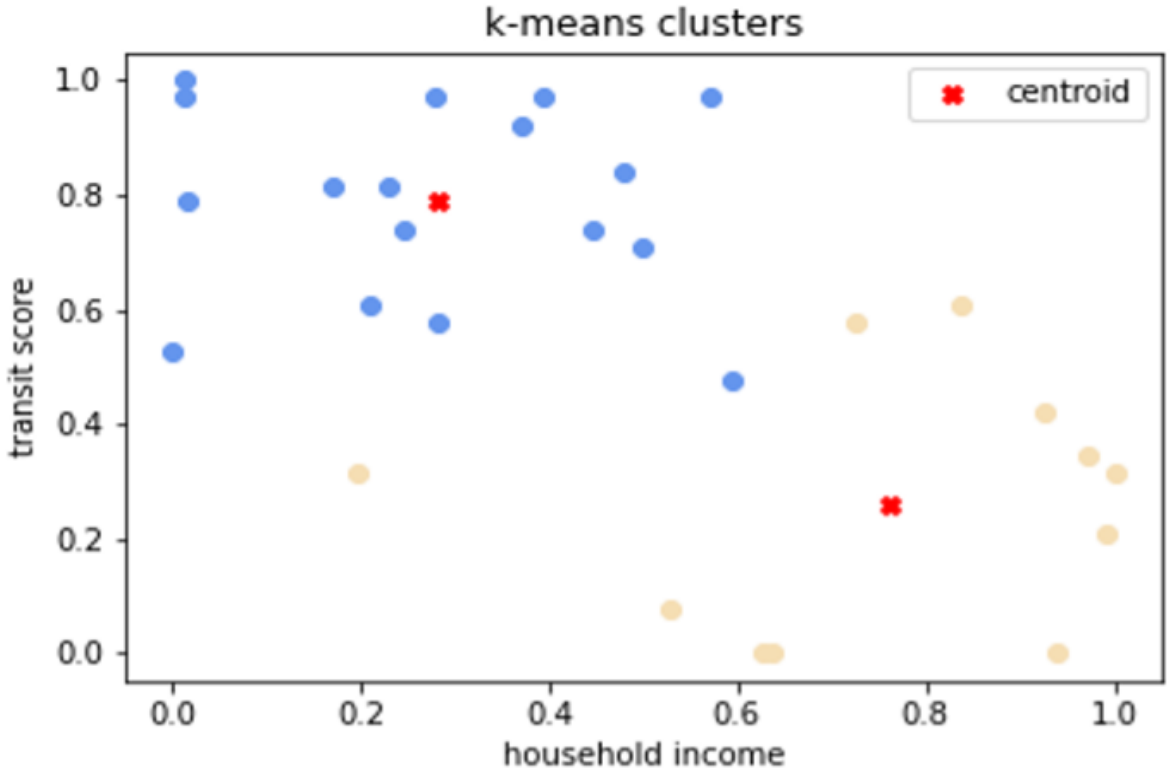


Figure: Plot of four features projected onto transit score and household income. Points are colored by cluster.

Selected Neighborhood

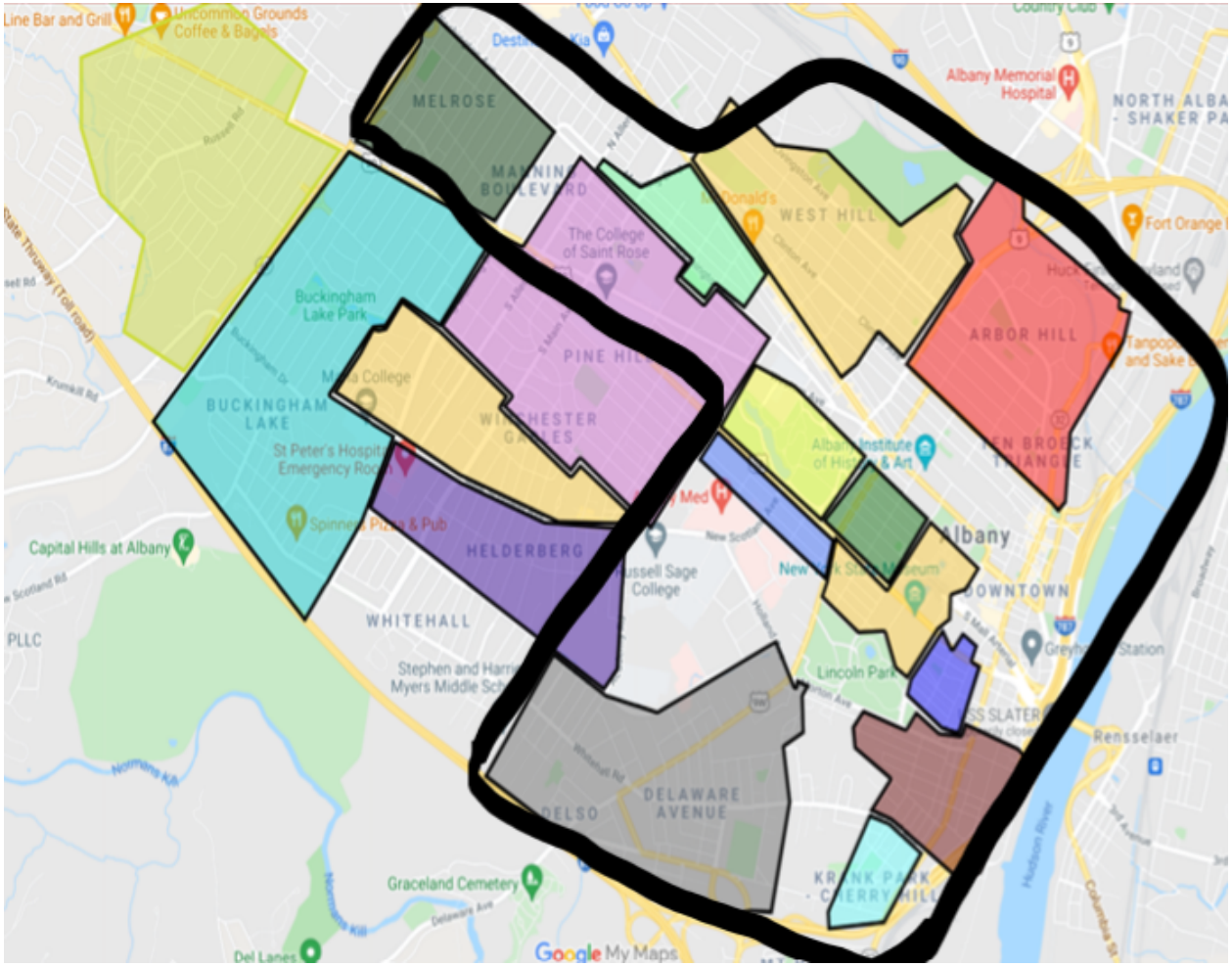


Figure: The selected cluster includes neighborhoods such as Arbor Hill, West Hill, Melrose, and others (circled on map).

Thank You!

Dimensionality Reduction

Clean and Normalize Data

PCA is sensitive to scaling. The following scaling methods were used and compared in my study;

1. maximum absolute scaling, and
2. min-max scaling.

PCA with MaxAbs Scaling

With this scaling three components were found to capture 80% of the variance and information of the data.

```
array([0.52770388, 0.18953096, 0.1050589 , 0.04406292, 0.03069867,
       0.02201854, 0.01668323, 0.01423243])
```

Figure: Explained variance ratio computed in PCA.

Features in Components

The first principal component has the corresponding features;

1. married,
2. very advanced degrees, and
3. one person household.

Visualization of Data Under MaxAbs Scaling

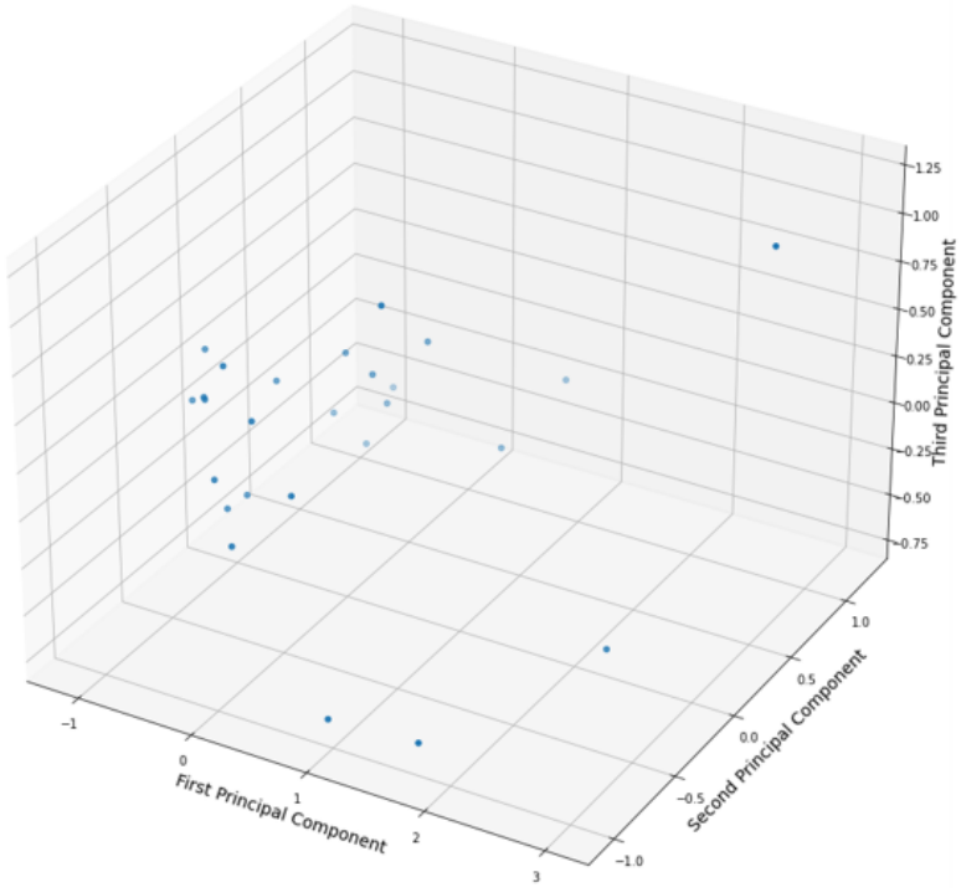


Figure: Graph of data after being projected onto the first three principal components.

PCA Under Min-Max Scaling

```
array([0.49948707, 0.21246639, 0.10871998, 0.04505744, 0.02913967,
       0.02343402, 0.01580902, 0.01560273])
```

Figure: Explained Variance Ratio of PCA with Min-Max Scaling.

As you can see, the first three principal components still capture 80% of the data's variance and information.
The corresponding features are the same except the last one - Age 22 to 29.

Feature Clustering (based on MinMax scaler)

First Group:

- Married Status.
- Advanced degree.
- One person household.

Second Group:

- Walk score.
- Transit score.

Third Group:

- Bike score
- Single female count

Visulization of Data (under MinMax scaler)

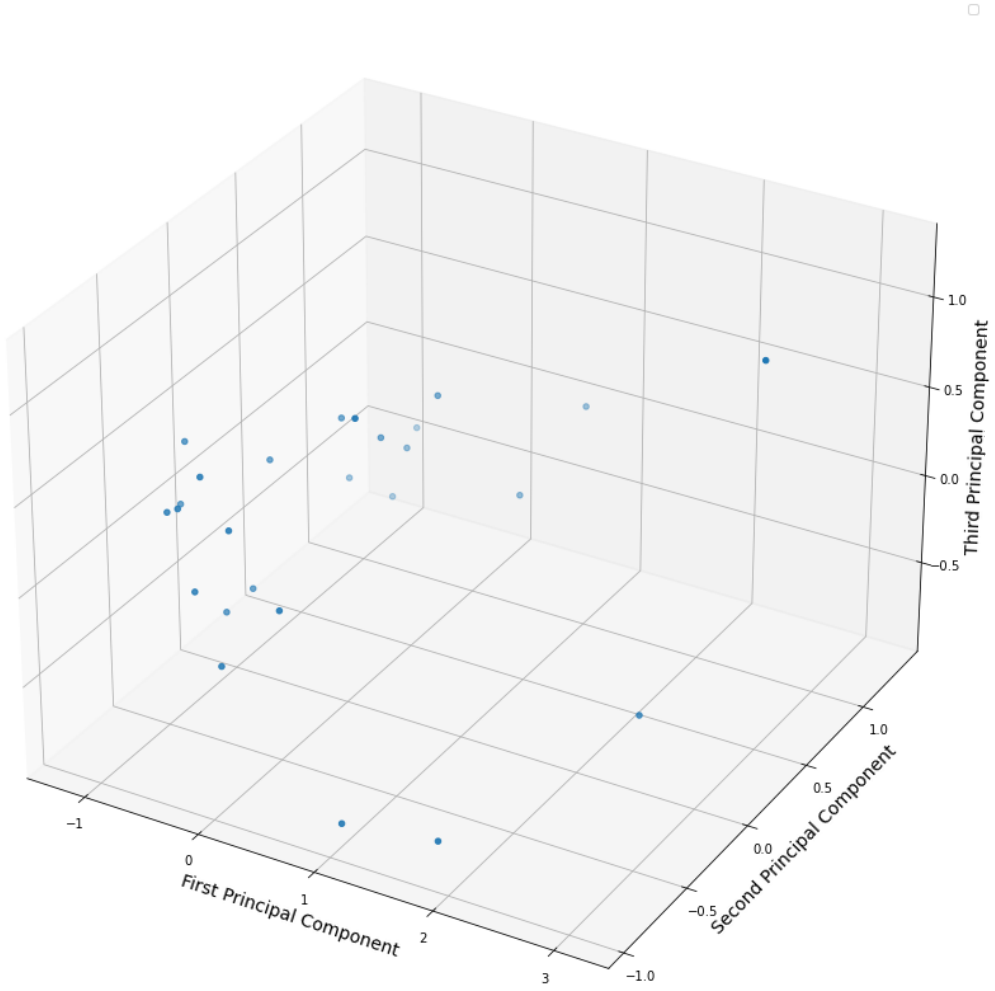


Fig: Similar pattern compared to last case

Implications

1. Married status and advanced degree status are essential in demographic and social data.
2. One person households and the age group 22 to 29 may be correlated somehow. Perhaps they are more likely to be single and living along?

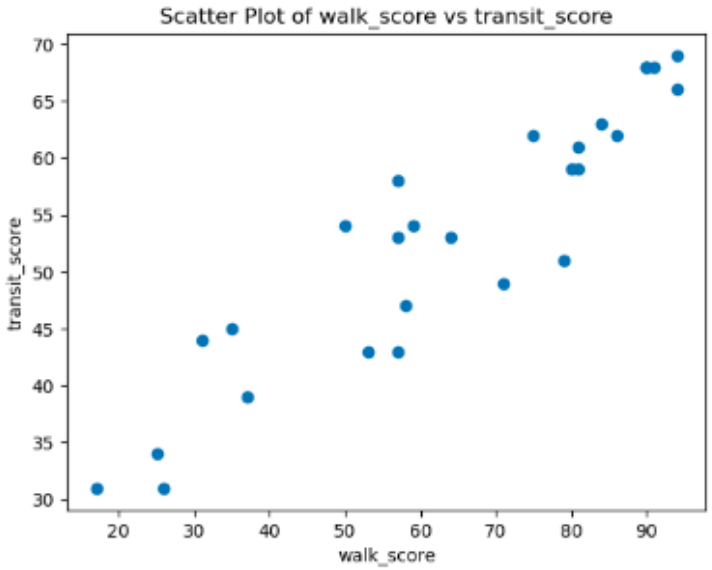
Thank you!

Linear Regression

- What is the regression and so correlation amongst key features?
- Can we predict the marital status and other features based on the population of specific neighborhoods of Albany, NY ?
- Features walk_score and transit_score are correlated together by regresssion, which also matches PCA.

What is the regression and so correlation amongst key features?

```
X=np.array(dataFrame['walk_score'])
Y=np.array(dataFrame['transit_score'])
plt.xlabel('walk_score')
plt.ylabel('transit_score')
plt.title("Scatter Plot of walk_score vs transit_score")
plt.scatter(X,Y)
plt.show()
```




```
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
X=np.array(dataFrame['walk_score']).reshape(-1, 1)
Y=np.array(dataFrame['transit_score']).reshape(-1, 1)
#poly.fit(X, Y)
lin1 = LinearRegression()
lin1.fit(X, Y)
```

```
LinearRegression()
```

```
lin1.coef_
```

```
array([[0.45859024]])
```

```
ypredict=lin1.predict(X)
```

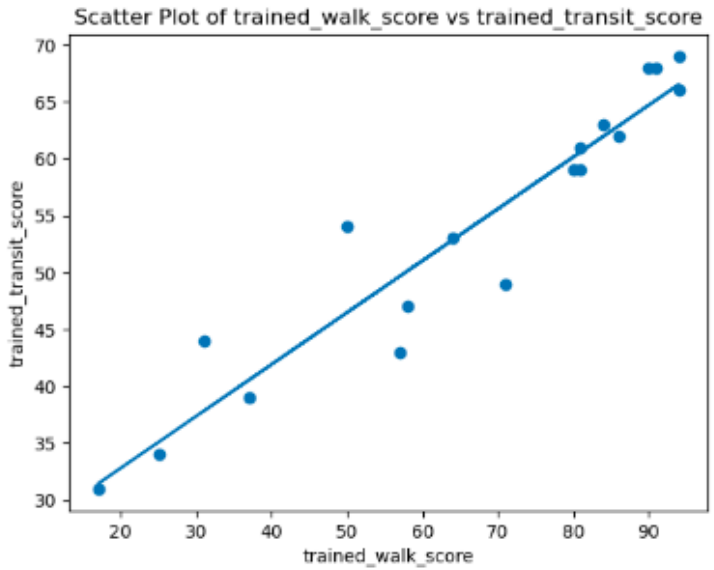
```
plt.plot(X,ypredict)
plt.scatter(X,Y)

plt.xlabel('walk_score')
plt.ylabel('trained_transit_score')

plt.title("Scatter Plot of walk_score vs trained_transit_score")


plt.show()
```


```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.33, random_state=42)
```

```
lin2 = LinearRegression()
lin2.fit(X_train, Y_train)
```

```
LinearRegression()
```

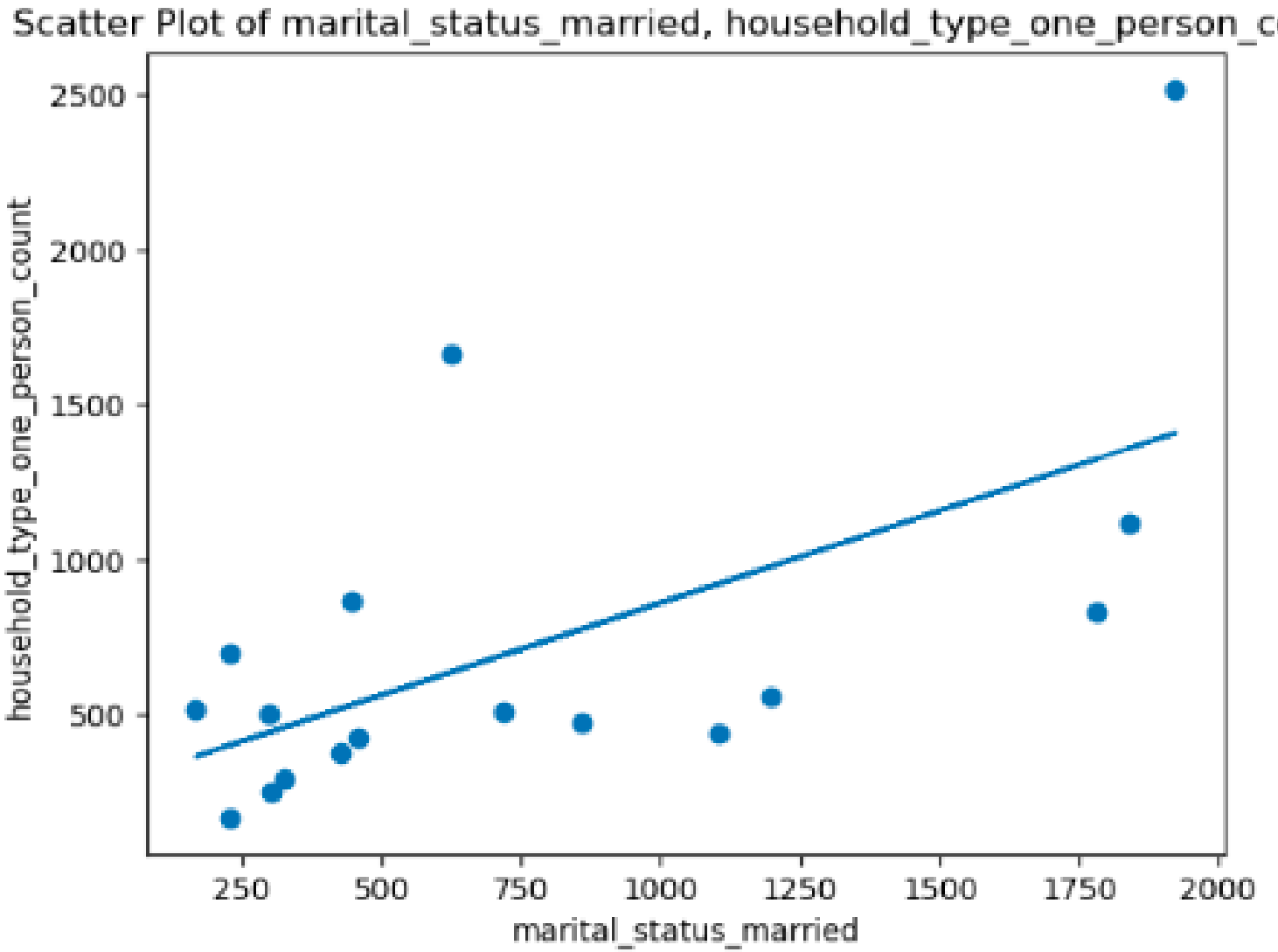
```
lin2 = LinearRegression()
lin2.fit(X_train, Y_train)
y_trainpredict=lin2.predict(X_train)
plt.plot(X_train,y_trainpredict)
plt.scatter(X_train,Y_train)

plt.xlabel('trained_walk_score')
plt.ylabel('trained_transit_score')

plt.title("Scatter Plot of trained_walk_score vs trained_transit_score")


plt.show()
```



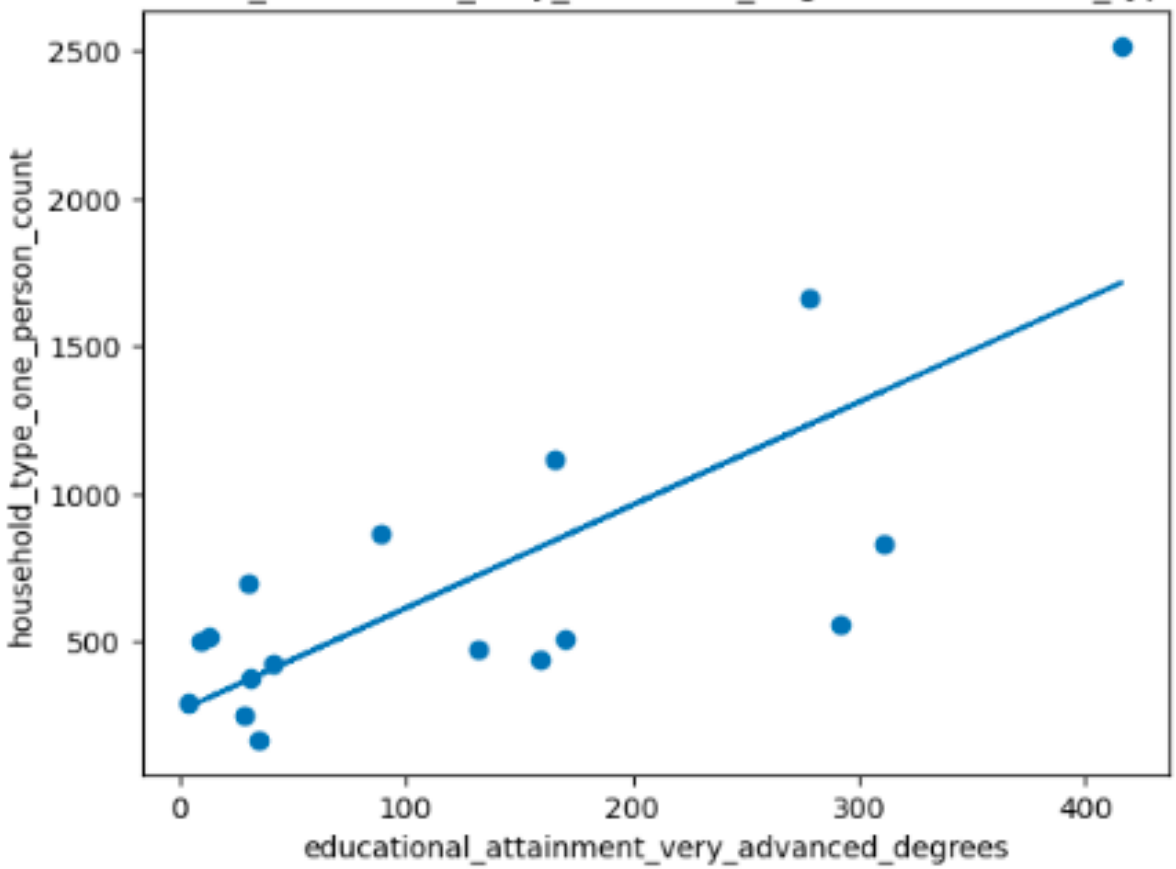
The following features may rather quite not correlate by regression, yet they are extracted by PCA.

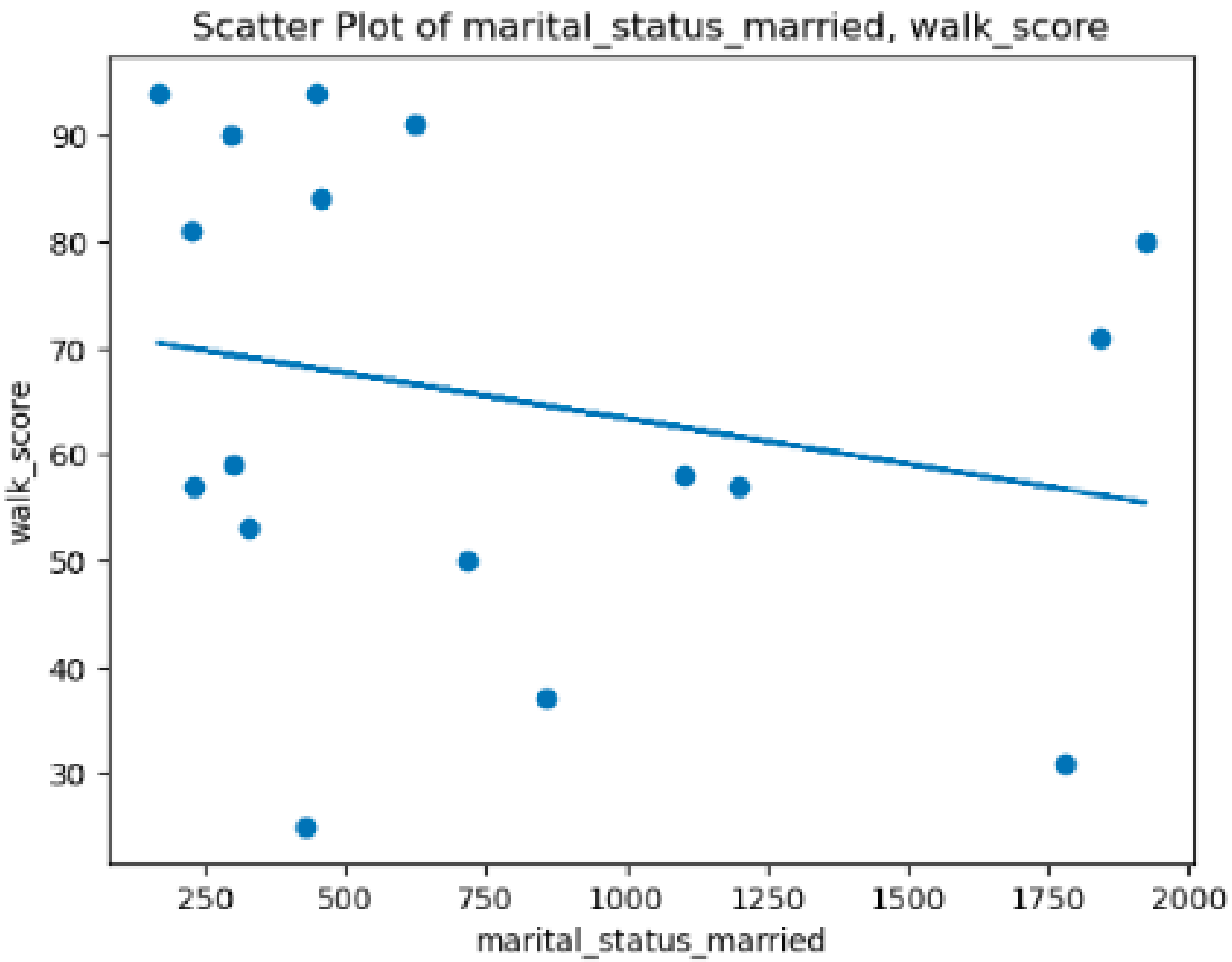
The following features may rather quite not correlate by regression, yet they are extracted by PCA.

```
Z=np.array(df['marital_status_married']).reshape(-1, 1)
U=np.array(df['educational_attainment_very_advanced_degrees']).reshape(-1, 1)
#W=np.array(df['household_type_one_person_count']).reshape(-1, 1)
#poly.fit(X, Y)

#Z_train, Z_test, U_train, U_test, W_train, W_test = train_test_split(Z, U, W, test_size=0.33, random_state=42)
Z_train, Z_test, U_train, U_test = train_test_split(Z, U, test_size=0.33, random_state=42)

s = 0
count = 0
for i in range(len(U)):
    if not np.isnan(U[i]):
        s = s + U[i]
        count += 1
avg = s/count
for i in range(len(U)):
    if np.isnan(U[i]):
        U[i] = avg

Z=np.array(df['marital_status_married']).reshape(26, 1)
U=np.array(df['educational_attainment_very_advanced_degrees']).reshape(26, 1)
#W=np.array(df['household_type_one_person_count']).reshape(26, 1)

lin3 = LinearRegression()

#lin3.fit([Z_train, U_train], W_train)
lin3.fit(Z_train, U_train)

z_trainpredict=lin3.predict(Z_train)
plt.plot(Z_train,z_trainpredict)
#plt.scatter(Z_train,U_train,W_train)

plt.scatter(Z_train,U_train)

plt.xlabel('marital_status_married')
plt.ylabel('educational_attainment_very_advanced_degrees')

#plt.Wlabel('household_type_one_person_count')

plt.title("Scatter Plot of marital_status_married, educational_attainment_very_advanced_degrees")


plt.show()
```


Scatter Plot of marital_status_married, household_type_one_person_count
household_type_one_person_count
2500
2000
1500
1000
500
250
500
750
1000
1250
1500
1750
2000
marital_status_married

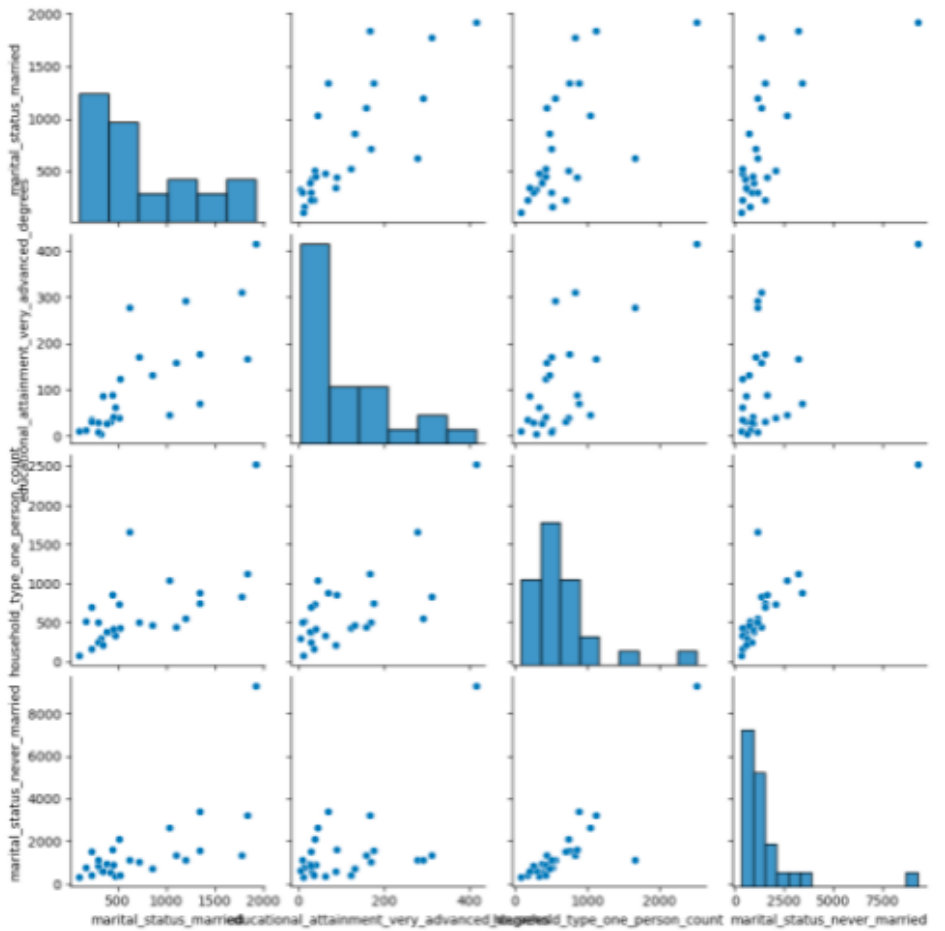
Scatter Plot of educational_attainment_very_advanced_degrees, household_type_one_person_count




Scatter Plot of marital_status_married, walk_score
walk_score
90
80
70
60
50
40
30
250
500
750
1000
1250
1500
1750
2000
marital_status_married

Improved visualizations of pairwise scatter plots with Seaborn

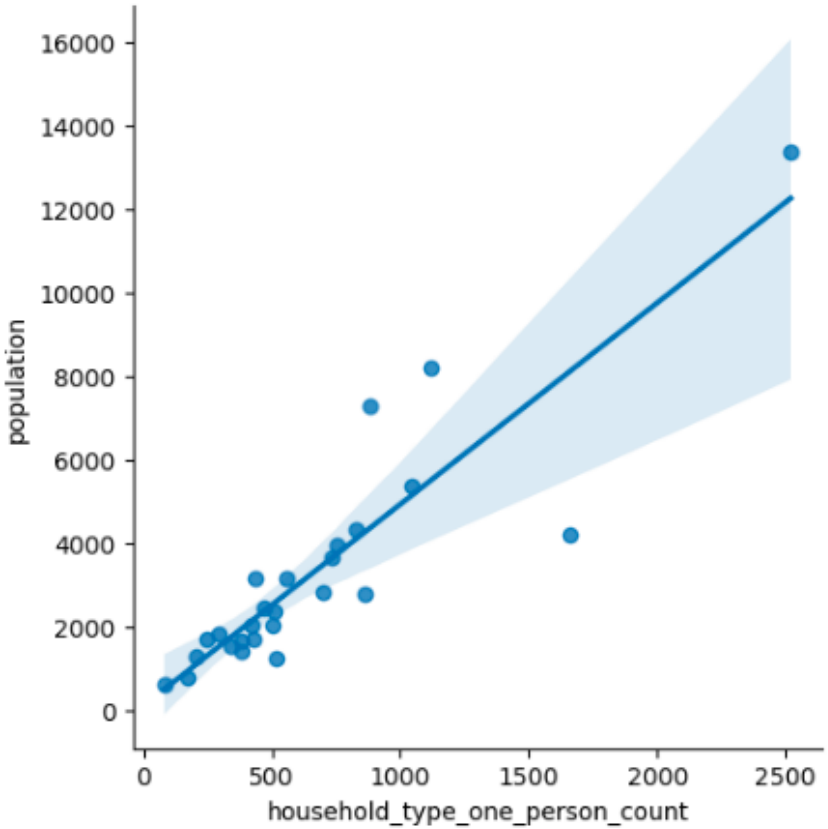
```
import seaborn as sns
sns.pairplot(df[["marital_status_married", "educational_attainment_very_advanced_degrees", "household_type_one_pers
plt.show()
```



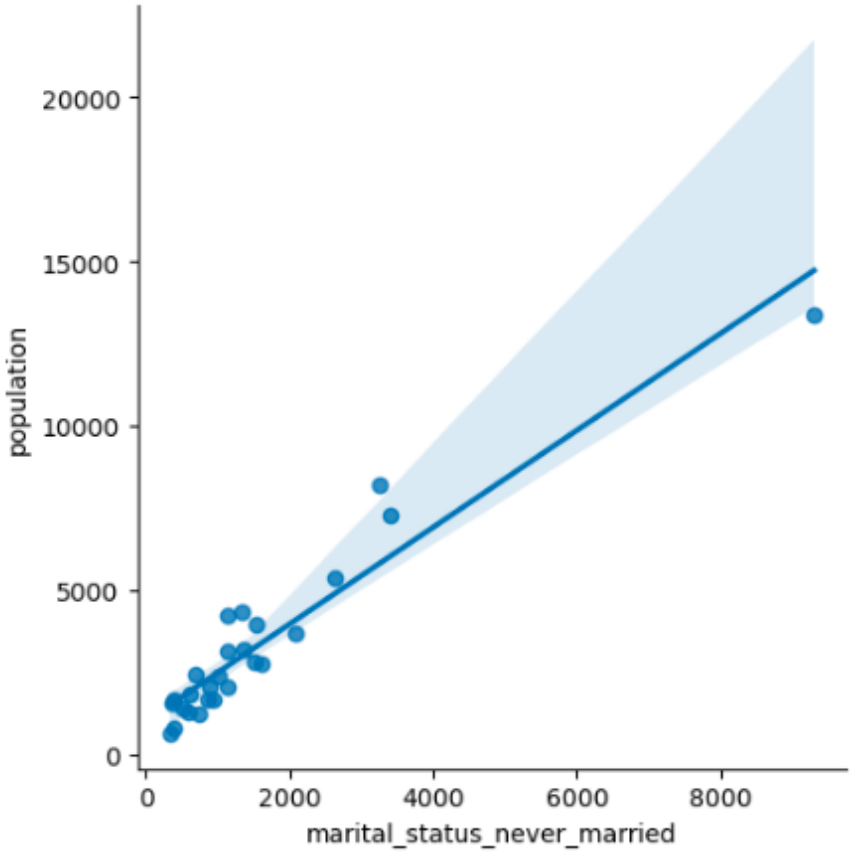
Adding confidence intervals to the correlation plots

Adding confidence intervals to the correlation plots

```
sns.lmplot(x="household_type_one_person_count", y="population", data=df[["household_type_one_person_count", "populat
plt.show()
```



```
sns.lmplot(x="marital_status_never_married", y="population", data=df[["marital_status_never_married", "population"]]
plt.show()
```



Q & A ?
Thanks!