On KNN Classification

by Antoine Love

December 17, 2021

Contents

1 Abstract

2 Data Explanation

The data set these simulations were run on is the census dataset **??** for Albany, NY. There are 35 features with complete data and 26 neighborhoods from which they come. Relatively speaking this is a small-sample-high-dimension data set. Figure 1 lists these thirty-eight features.

Feature Category	**Feature Names**
Employment/Financial	Household Income, Private Sector, Self Employed Sector, Non Profit Sector
Marital Status	Married, Separated/Divorced, Widowed, Never Married
Race	White, Hispanic, Black, Asian, Mixed, Other
Education	No High School, Bachelors, Very Advanced Degrees
Household Type	Married, Single Female, Single Male, One Person, Non Family, With Children
Age Related	Age 0 to 17, Age 18 to 21, Age 22 to 29, Age 30 to 39, Age 40 to 49, Age 50 to 59, Age 60 to 69, Age 70 to 79, Age 80 or older
Modality	Walk Score, Transit Score, Bike Score
Other	Population, Population Density, Food Stamps
Introduced	*Free Fridge Food Locations*

Figure 1: A listing of the thirty-eight features within the dataset. Most neighborhoods in Albany have data for each, but several are missing many. For example, the neighborhoods 'Pastures' and 'West End' are missing all data on education, food stamps, and households with children. Moreover, the employment features (private, self-employed, non-profit) are missing data for about 21% of the other neighborhoods. The remaining thirty-five features are complete.

3 Classification of Impoverished Neighborhoods

3.1 Overview

Like many large cities Albany New York has a very diverse population with all levels of socioeconomic statuses represented at the individual level. On the coarse level census data [3] provide aggregated snap-shots of the city's socioeconomic status at the level of neighborhoods. The broad topic here concerns one's ability to use these coarse data for the classification of impoverished individuals within neighborhoods.

In the state of New York there are many social welfare initiatives to aid those people who are impoverished, but many of these require those seeking help to traverse bureaucracy's long list of forms before receiving aid. One alternative initiative, Free Food Fridges in Albany [1], seeks to provide free food to those who need it without the work and registration normally required.

The overarching question we are interested in is, "How accurately can we mimic the classification of impoverished neighborhoods by the Free Food Fridges in Albany organization with available census data?" Perhaps, we can learn how to sort through coarse measurements of a city in order to identify the impoverished people-groups within neighborhoods. Results from a suite of K-Nearest Neighbor classifications and an analysis of their quality follows.

3.2 Accuracy Stabilizing Process

The basic process to mimic the classification is to choose a set of census features, some training set from the data set, some k-value, and construct a K-Nearest Neighbor classifier to predict if the neighborhoods in the remaining neighborhoods should or should not classify as needing a free food fridge. The accuracy of any classifier was judged based on its agreement with the actual location of the fridges; i.e. did it predict correctly where fridges are actually found?

Despite the question and process being well-posed, it was ill-answered in initial classification testing. With only twenty-six neighborhoods having a complete set of census data and five having a free food fridge, there was a high amount of variance in classification accuracy. For a given set of features and any two given samples of neighborhoods and some fixed k-value, accuracies could be as different as 33% and 100%. A method of accuracy validation was needed to provide a more stable measure of classifier accuracy. Algorithm 1 was constructed in order to create classifiers which would a) provide a stable accuracy measure and b) be independent of each other.

Algorithm 1: Algorithm for Resampling and Classifier Construction

```
for n ≤ResampleCount do
    Randomly split neighborhoods into training and testing sets.;
    for k ≤MaxK do
        Construct KNN Classifier with k-value.;
        Train Classifier on training set.;
        Obtain accuracy on testing set.;
    end
end
Average accuracies for each value of k.
```

By their very nature, the average accuracies obtained from Algorithm 1 allow one to compare accuracies from various feature-sets through the Central Limit Theorem. With a choice of ResampleCount, one obtains corresponding significance levels and confidence intervals which can be used

for hypothesis testing. As an example and because most of the best accuracies found are near 0.83, the above process provides a confidence interval of $(.8203, 0.8396)$ for ResampleCount$= 10,000$ at the 1% significance level for a two-tailed test.

3.3 Preliminary Results

Before continuing with the preliminary results, it would be wise to first step back and decide, "what is our target accuracy, or benchmark?" Certainly, one might be pleased if they obtained an accuracy of 81%. However, this accuracy could be obtained by simply guessing 'no fridge' in every neighborhood no matter what. So it is that we use $21/26 \approx 80.77\%$ as our benchmark and its 99% confidence interval on 10,000 trials of $(0.79754, 0.81784)$. Any average accuracy found outside this interval after 10,000 trials by Algorithm 1 can safely be considered better or worse than the benchmark classifier.

Algorithm 1 was run on all complete thirty-five one-feature and all 595 two-feature sets with ResampleCount$= 10,000$ and for k-values from zero to seven and zero to five, respectively. Additionally, 1,694 three-feature sets of the 6,545 possible were tested.

Accurate classifiers were found in each set of tests and for various values of k. For the single-feature classifiers, only the number of widows in a neighborhood for $k = 1$ was found to provide a significant accuracy above the benchmark. As the number of features included in the classifier set increased, so did the number of significantly accurate classifiers. Eight of the 595 were significantly more accurate than the benchmark. Seventeen of the 1,694 three-feature classifiers were significantly as well. Plots of these classifier accuracies can be found in figures 2 and 3. These plots show the dependence of classifier accuracy on k-value. The precise accuracies, k-values, and feature-sets of the significantly accurate classifiers can be found in figure 6 at the end of this paper.

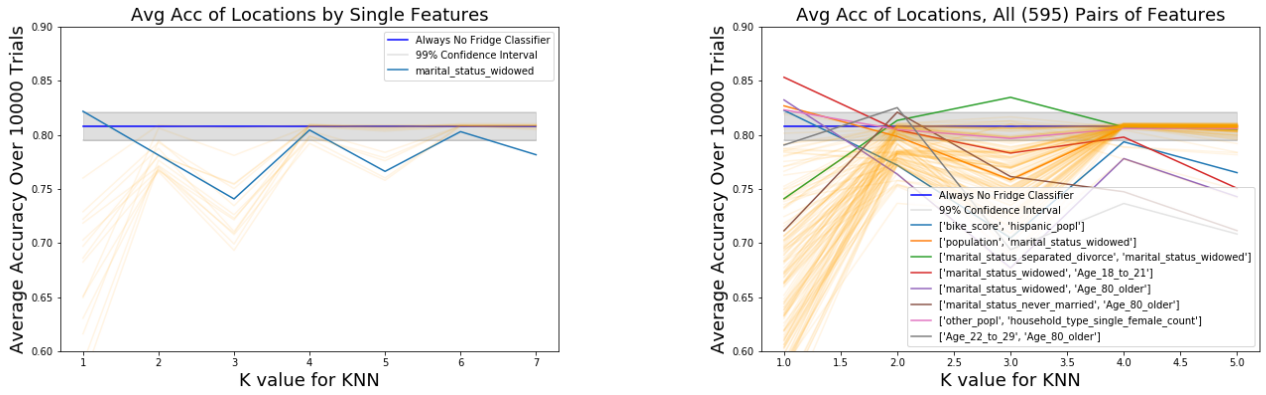

Figure 2: Plot of average accuracies for single-feature and two-feature sets with ResampleCount$=$ $10,000$ and various k-values. Feint yellow lines are sequences of accuracies for classifiers which performed as well as the benchmark. Classifiers which performed more poorly than the benchmark are not shown.

An analysis of errors from the classifiers in the triple-feature classifier set shows a tendency for classifiers to err on the side of 'no fridge', with some notable exceptions where the errors are balanced. This examples how classifiers can become heavily biased in the face of imbalanced labels in data; almost always giving one response despite empirical evidence. Confusion matrices for the top three triple-feature classifiers are shown in figure 4 for confirmation of this phenomenon.

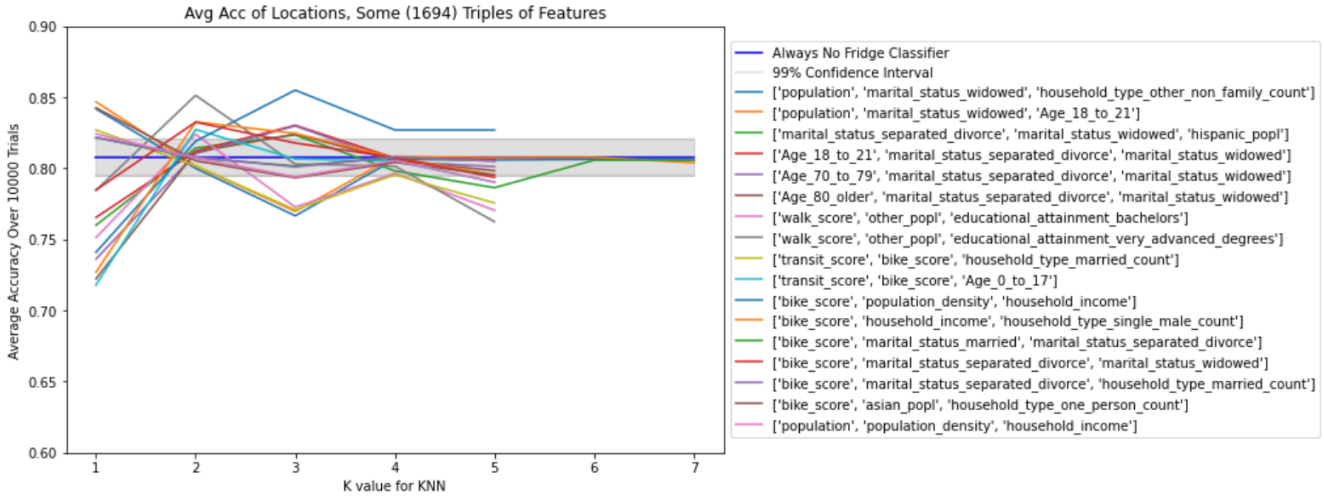

Figure 3: Plot of high-average-accuracy triple-feature sets with ResampleCount= 10,000 and various k-values. Classifiers which performed more poorly or as well as the benchmark are not shown for clarity.

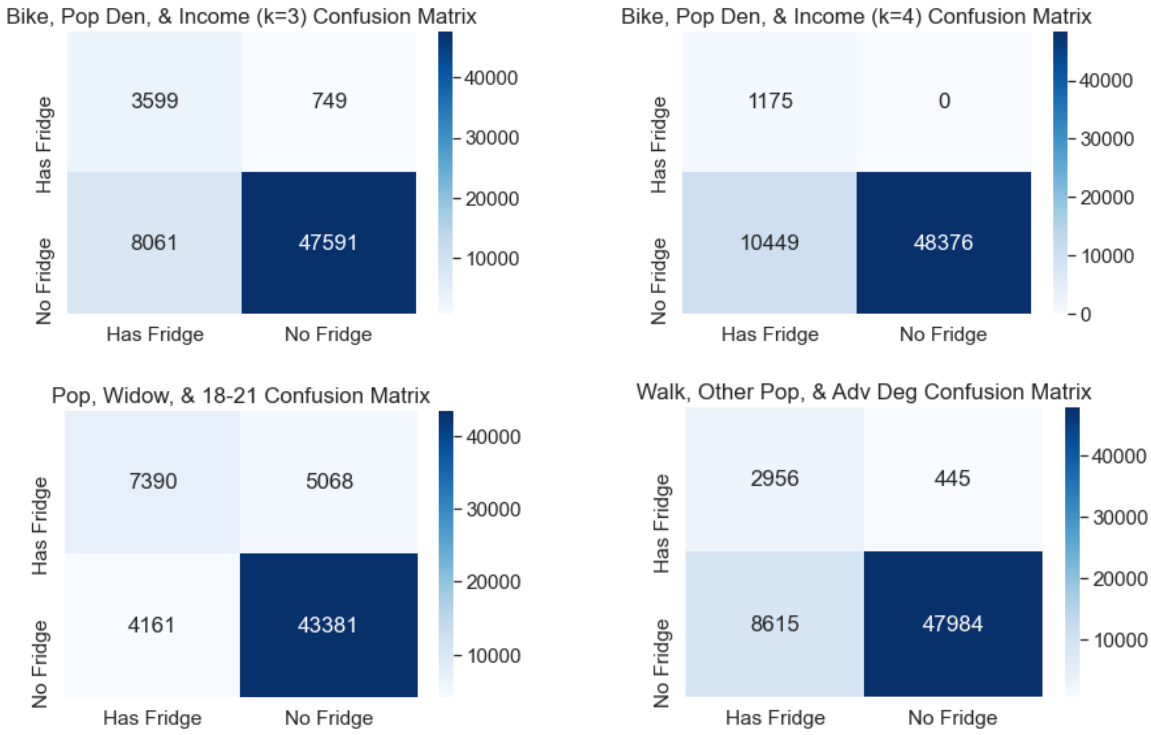

Figure 4: Confusion matrices for top three triple-feature classifiers. Top row: Bike score, population density, and median income features for $k = 3$ (left) and $k = 4$ (right).

4 Computation Cost

All of the code, data, and accuracy results can be found in a GitHub respository found at [2]. The accuracy tests from this section were run on two machines: an AMD Ryzen 5 1600 six-core processor at 3.3 GHz with 16.0 GB of 2400 MHz DDR4 RAM (most of the triple feature sets) and an AMD Ryzen 7 5700U eight-core processor at 4.3 GHz with 16.0 GB of 3200 MHz DDR4 RAM

(all single and two-feature sets, some triple feature sets). Total computational time exceeded one week for all tests combined.

For this amount of time and such a small dataset, it would have been much nicer to have searched further afield in the possible set of features from which to classify.

4.1 Faster KNN Feature Selection

While working with KNN I realized how difficult it was to select good features to use as classifiers. Moreover, with small data samples there was a need to run long simulations with bootstrapping to smooth out high variances in accuracy values. Hence, the need to quickly check for possible high-accuracy-featuresets was imperitive. The following method was initially an attempt to speed up the SKLearn KNN algorithms which I suspected had a lot of overhead for handling arbitrary labels and not just binary 1s and 0s, repetitive computation of distance matrices which do not change, and general overhead required for setting up the KNN class. After initial testing it was apparent that this method quickly picks out high-accuracy-featuresets based on confirmation by the high accuracies found with the much slower simulations. However, this method always appears to overestimate the accuracy by 1-3%, leading me to believe that this is some sort of upper bound computation. Though I have no proof yet of this claim. See Algorithm 2 for the heuristic as well as GitHub **??** for the code and results.

Algorithm 2: Fast KNN Feature Selection (FKFS)

Let $i, j \leq |X|$, where X is our data set. Compute the distance matrix $D = \big(d(x_i, x_j)\big)$;
Construct the label matrix $L = \big(l_j\big)$;
Sort each row of L according to how rows of D would be sorted.
Construct a prediction matrix P through the following: **for** *Row in L* **do**
    Column 1 is column 1 of L.
    Column 2 is column 2 of L.
    for *Columns up to* $kMax - 1$ **do**
        The column has the label of the mode of the labels in
        columns 2 through the current column.
    end
end
Construct an agreement matrix A as follows:

$$A_{i,j} = \begin{cases} 1 & \text{if } P_{i,0} = P_{i,j} \\ 0 & \text{Otherwise} \end{cases}$$

return Vector of averages of columns of A.

4.2 FKFS Validation

Two feature-sets were used initially to test the validity of FKFS. These were the two-feature sets 'marital status widowed' with 'age 18 to 21' and 'marital status widowed' with 'marital status separated divorced'. The former was known to have an average classifier accuracy of 0.85314 after 10,000 trials for $k = 1$. The latter was known to have an average accuracy of 0.834 for $k = 3$. FKFS predicted an accuracy of 0.8846 and 0.846 for each respectively.

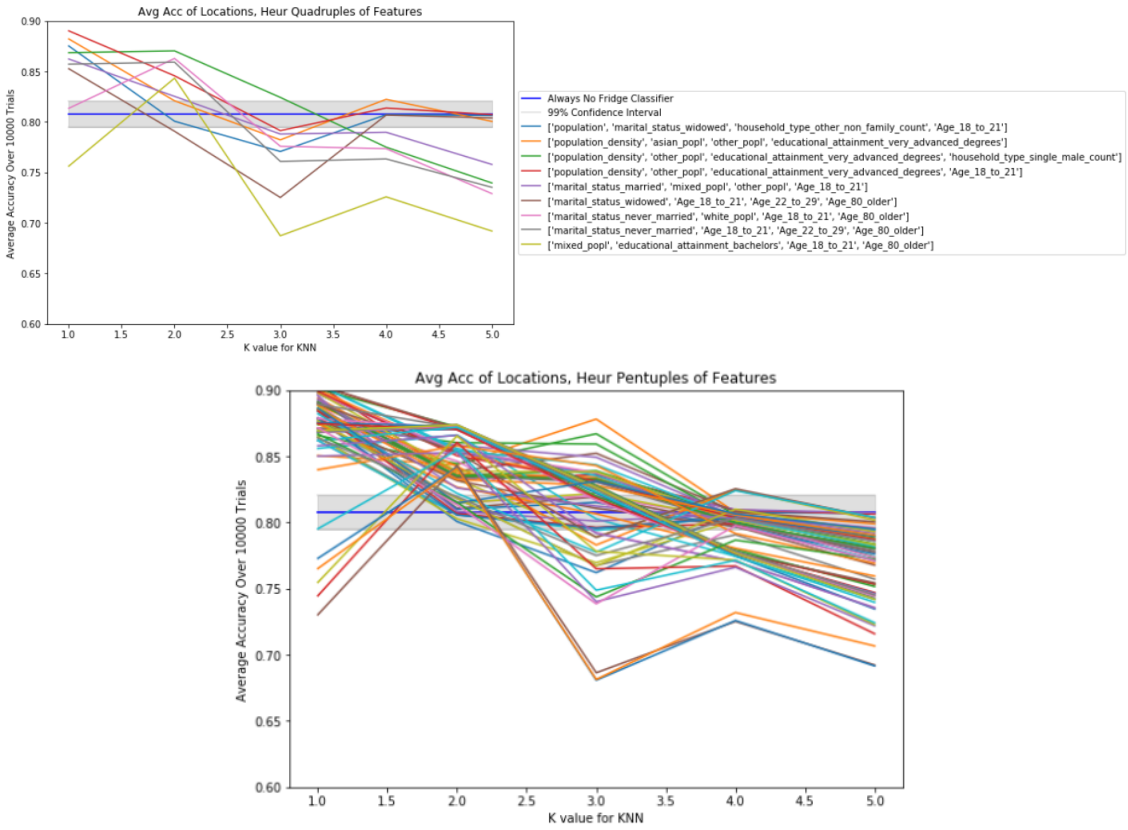

Figure 5: Validation of many four-feature and five-feature sets suggested by FKFS.

Much validation then ensued. FKFS was used to suggest three-, four-, and five-feature sets for classication. The suggestions were done on various minimum accuracy requirements to reduce the number of suggested feature sets.

4.2.1 Suggested Three-Feature Sets

Only 1,964 of the 6,545 possible three-feature sets were tested before FKFS. These sets were found after a week of computational time. In 67 seconds FKFS had searched all 6,545 and suggested many sets above the 0.82 accuracy mark and two above the 0.92 accuracy mark. See figure 7 at the end of the paper for the accuracy testing results and a list of the feature sets.

4.2.2 Suggested Four-Feature Sets

All of these were undiscovered prior to the heuristic. Note also that nine of these (counting multiples for different k-values) outperform the best feature-set discovered prior. These were suggested from the heuristic inspecting all 52,360 four-feature sets possible in 544 seconds. See figure 5 for accuracy testing results and feature sets.

4.2.3 Suggested Five-Feature Sets

All of these were undiscovered prior to the heuristic. These were suggested from the heuristic inspecting all 324,632 five-feature sets possible in one hour and one second. See figure 5 for the accuracy testing results and figure 8 for the list of feature sets at the end of this paper.

5 Future Work

This is a fantastic result. It allows us to much more quickly discover possible high-accuracy feature-sets. Questions still exist:

- Is this a general result or is it specific to these data?
- How can we further improve this result?
- What other results can we use to suggest high-accuracy feature sets?
- How inaccurate is this method? i.e. Are there bounds on how far off the predicted accuracy is from the true accuracy? Usually this method overestimates the accuracy of the random resampling. By how much does it do so? Are there instances where it underestimates?

6 Additional Tables and Figures

	Features	**Accuracy**	**k-Value**
1-Feature	Widowed	0.82171	1
2-Feature	Widowed, Age 18 to 21	0.85314	1
	Widowed, Separated/Divorced	0.83463	3
	Widowed, Age 80 or Older	0.83222	1
	Widowed, Population	0.82674	1
	Age 22 to 29, Age 80 or Older	0.82509	2
	Other Pop, Single Female	0.82308	1
	Bike Score, Hispanic Pop	0.82264	1
	Never Married, Age 80 or Older	0.82091	2
3-Feature	Bike Score, Household Income, and Pop Density	0.85495	3
		0.82693	4
		0.82693	5
	Walk Score, Other Pop, and Very Adv Degrees,	0.85129	2
	Population, Widowed, and Age 18 to 21	0.84672	1
	Bike Score, Asian Pop, and One Person	0.84253	1
	Population, Widowed, and Non-Family	0.84179	1
	Bike Score, Household Income, and Single Male	0.83262	2
		0.82453	3
	Bike Score, Separated/Divorced, and Household Married	0.83243	2
	Separated/Divorced, Widowed, and Bike Score	0.83078	2
	Separated/Divorced, Widowed, and Age 18 to 21	0.83023	3
	Separated/Divorced, Widowed, and Age 70 to 79	0.82978	3
	Transit Score, Bike Score, and Age 0 to 17	0.82725	2
	Transit Score, Bike Score, and Household Married	0.82697	1
	Population, Pop Density and Household Income	0.82418	1
	Separated/Divorced, Widowed, and Age 80 or Older	0.82396	3
	Walk Score, Other Pop, and Bachelors	0.82390	2
	Separated/Divorced, Widowed, and Hispanic Pop	0.82338	3
	Bike Score, Marital Status Married, and Separated/Divorced	0.82178	1

Figure 6: Table of discovered classifiers with average accuracies significantly above $a = 80.77\%$.

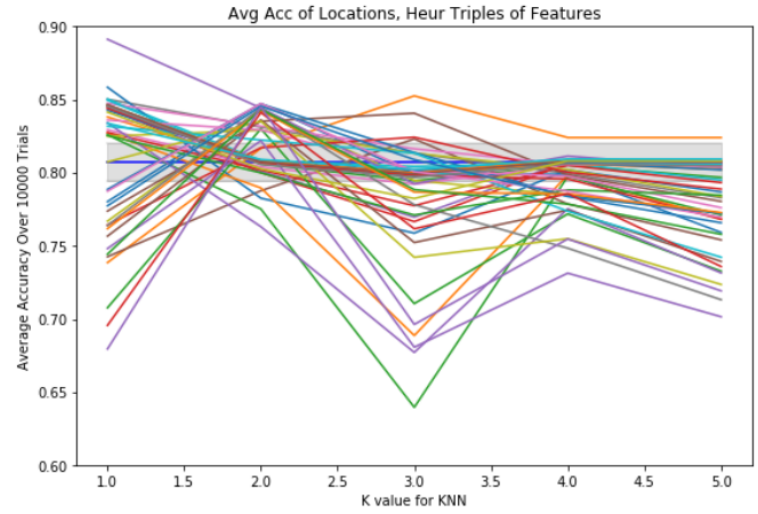

Figure 7: Validation of many three-features sets suggested by FKFS.

Figure 8: Feature-set list for five feature classifiers in figure 5.

References

[1] J Anderson. *Free Food Fridge Albany*. Acc. Nov 16, 2021. URL: https://freefoodfridgealbany.com/.

[2] A Love. *SE Data from Albany, NY - Analysis*. Dec. 8, 2021. URL: https://github.com/AntoineLove/SEDataFromAlbanyNYAnalysis.

[3] Christopher Yong et al. *An Integrated Dataset of Civil Issue Reports and Neighborhood Characteristics for Computational Social Science Research*. Harvard Dataverse, 2019. DOI: 10.7910/DVN/WQ2M1H. URL: https://doi.org/10.7910/DVN/WQ2M1H.