Anthony Gray
EN.605.649.81
Intro to Machine Learning
Project 3

Abstract
The *k*-nearest neighbor algorithm was used to perform classification and regression on two datasets each. Permutations of the algorithm involving the distance measurement used, as well as the construction of the training data were evaluated. The regression algorithm was error prone, though it performed best using Manhattan distance on one dataset, and comparably with Manhattan and Euclidean distance on the other, though Euclidean distance provided a small advantage. The classification algorithm had the best classification rate using Manhattan distance and stratified training data on both datasets. Unexpectedly, condensing the training data before classification lead to a performance drop in both datasets.

Problem Statement and Algorithms
The *k*-nearest neighbor algorithm (1) was implemented twice, once for classification and once for regression. In both cases, the function is passed a training dataset, a test dataset, a value for *k* and a method for computing distance. The function iterates over the test dataset and computes the distances from each test item to all the training items. It then orders those distances and chooses the *k* lowest. The classification algorithm then assigns the most common class among those *k* examples, while the regression algorithm takes their average. Both Euclidean and Manhattan distance were implemented.
Condensed *k*-Nearest (2) was implemented as a pre-processing algorithm for the classification algorithm. It accepts the training data and a distance measurement as parameters and works by shuffling the training data, choosing the first item in the training data to initialize subset *Z*, then iterating over each example. For each example, it finds the closest point and tests for accurate classification using only the subset of examples *Z.* If *Z* is able to correctly classify the point, nothing happens, but if the classification is incorrect, the point is then added to *Z*. After each iteration over the training set, the points in *Z* are subtracted from it. This process is repeated until no more points are added to *Z*, implying that *Z* is a sufficient subset for accurate classification. *Z* is then returned to use as a training dataset.
Performance was evaluated using 5-fold cross-validation. The dataset was split into five subsets of equal size, and the algorithms were run with each subset serving as the test set in turn. The other four were combined to act as the training set. For the classification algorithm, the data was also stratified, or split into fifths by class, with the remainder being assigned to each subset randomly. The error measures of each of the five runs were stored and averaged in order to evaluate the algorithm.
The two error measures used were mean squared error for the regression algorithm, and classification accuracy for the classification problems.
These algorithms implemented in python and were run on the Ecoli (3), Image Segmentation (4), Computer Hardware (5) and Forest Fires (6) datasets.
The regression algorithm will be tested with both distance functions to see if one results in better performance than the other. The classification algorithm will also be tested with both distance functions, as well as with stratified or randomly split data and with condensed or unfiltered training data to determine which combination of factors results in the best performance. It is likely that stratifying and condensing the data will improve performance relative to random splitting and unfiltered data, as these methods ensure that at least one representative point for each class is present, provided there are enough examples in the overall dataset. It is also likely that the runs using Manhattan distance will be more accurate than the ones using Euclidean distance, particularly in those datasets with higher attribute counts.

Procedure/Tuning

Dataset	Instances	Attributes	Classes
Ecoli	336	7	8
Image	210	19	7
Hardware	209	6	Regression
Fires	517	12	Regression

All three datasets were obtained from UCI Machine Learning Repository (7). The class values in the Image dataset were moved from the row labels to the last column in the dataset. The ecoli and hardware datasets had unlearnable values removed (sequence, model and vendor names). The fires dataset contained weekday and month data represented as strings of common three letter abbreviations. The strings were replaced with numbers, with 1 and corresponding to January and February and Monday and Tuesday, respectively. All four datasets contained no missing values.

K values were tuned for each dataset and each combination of algorithms by running each combination 25 times and constructing plots of the results. The two *k* values that returned the most consistently high results were selected. *K* values were selected both for low error rates, but also for clustering. In several cases, the *k* value selected was the one that performed consistently well, rather than one that performed very well, but inconsistently. Some datasets produced a variety of *k* values that all performed well over a small accuracy range, and in these cases the most representative one was chosen. Plots of the *k* values can be found in the appendix.

Using the best combination of measurement function, splitting strategy, condensed or uncondensed data and *k* value for each dataset, the algorithms were run on their respective datasets 20 more times and the results of the error function were averaged.

Results

Tuning Data:

Dataset	Distance	Stratified	Condensed	Best K	2nd Best K	Loss Min	Loss Max	Error Range
Hardware	Manhattan	n/a	n/a	4	1	4000	18000	14000
Hardware	Euclidean	n/a	n/a	3	2	5000	21000	16000
Fires	Manhattan	n/a	n/a	4	3	4400	5300	900
Fires	Euclidean	n/a	n/a	4	3	2100	5100	3000
Image	Euclidean	Y	Y	1	n/a	0.31	0.41	0.1
Image	Euclidean	Y	N	1	n/a	0.735	0.805	0.07
Image	Euclidean	N	Y	1	n/a	0.275	0.425	0.15
Image	Euclidean	N	N	1	n/a	0.72	0.8	0.08
Image	Manhattan	Y	Y	1	n/a	0.34	0.44	0.1
Image	Manhattan	Y	N	1	2	0.78	0.835	0.055
Image	Manhattan	N	Y	1	n/a	0.35	0.45	0.1
Image	Manhattan	N	N	2	1	0.765	0.815	0.05
Ecoli	Euclidean	Y	Y	4	8	0.765	0.812	0.047
Ecoli	Euclidean	Y	N	7	8	0.825	0.86	0.035
Ecoli	Euclidean	N	Y	8	6	0.765	0.815	0.05
Ecoli	Euclidean	N	N	6	7	0.825	0.855	0.03
Ecoli	Manhattan	Y	Y	7	6	0.765	0.829	0.064
Ecoli	Manhattan	Y	N	8	6	0.863	0.865	0.002
Ecoli	Manhattan	N	Y	8	6	0.765	0.81	0.045
Ecoli	Manhattan	N	N	8	n/a	0.83	0.855	0.025

Best Performance Data:

Dataset	Performance Measure	Average Performance
Hardware	Mean Squared Error	7500.3
Fires	Mean Squared Error	4587.29
Image	Classification Percent Correct	87%
Ecoli	Classification Percent Correct	86%

The Tuning Data table above contains the *k* values and error ranges for each combination of algorithm and dataset. The combination determined to show the best results for each dataset is underlined and was used to produce the performance averages in the Best Performance Data table.

The hardware regression was performed using Manhattan distance, as it produced lower minimum and maximum mean squared errors across a smaller range.

Both distance measures produced similar results with Fire dataset, with the Euclidian distance measure producing lower minimum error, but with a greater range. The Manhattan distance measure produced a maximum error comparable to that of Euclidean distance, but with a much smaller range. Average performance with both distance measurements was evaluated on the Fires dataset, and the errors were similar, at 4587 and 4809. However, Euclidean distance produced the lower average error of 4587.

Manhattan distance produced average accuracies of a few percentage points higher than the same tests with Euclidian distances in both classification datasets. Stratification also had a similar effect, increasing the minimum and maximum accuracy percent by a few points, as well as narrowing the accuracy range in most cases. However, condensing the data before classification had very different effects on each dataset. The Ecoli dataset, when condensed, lost around 5% accuracy and increased the accuracy range in most cases. The Image dataset however, halved its accuracy percent when the training data was condensed. Minimum and maximum accuracies were roughly half of what was produced by the full dataset, and the accuracy range roughly double in all cases. *K* values were consistent within each dataset and remained stable with algorithm variant.

Discussion

Overall, the k-nearest neighbor classification algorithm performed well, with average accuracy between 85 and 90 percent. The regression algorithm did not perform as well however, with average mean square error often reaching several thousand. As expected, use of Manhattan distance provided better results than Euclidian distance most of the time, though the results were very similar in the case where it was did not. This was regression of the Fires dataset, which is the largest in terms of examples and attributes of the datasets examined. I could be that with 19 dimensions, both distance measures struggled to return accurate results.

Stratification of the dataset splits increased classification accuracy in both the Ecoli and Image datasets, though the increase was more noticeable in the Image set. This could be due to the fact that the Image set has equal numbers of every class, while the Ecoli set only has one or two examples of some of its classes. Without enough examples to represent each class in each subset during five-fold validation, stratifying the subsets by class wouldn't help with the underrepresented classes, as assignment to a subset would still be effectively random. It would have still helped with the more common classes, however.

A major surprise was that condensing the training set reduced performance in both classification problems, and substantially so in the case of the Image dataset. This is even more surprising considering how balanced the classes were in that set. One explanation is that the Image data is well clustered by class, with several outliers in each. In the uncondensed set, a vote of neighbors would be enough to accurately classify a test example, but in a smaller set, the chances of including a misleading point could be very high.

What is interesting is that the *k* values for the classification algorithms followed the same pattern whether the training set was condensed. The Ecoli data generally classified best with a *k* of about 7, showing clusters of *k* values between 5 and 9. The Image set consistently performed best with a *k* value of only 1,

with occasional outliers. Even with the poor performance of the condensed Image dataset, the *k* value did not increase to compensate.
The performance of the regression algorithm was also lower than expected. When testing the algorithm, it was observed to make guess very close to the actual value of the majority of instances, and often guessed the actual value exactly. However, when it guessed poorly, the answer was very wrong, and often off by several orders of magnitude. This may help explain the large mean-squared error, though the use of a more sophisticated kernel function would have also helped.

Summary

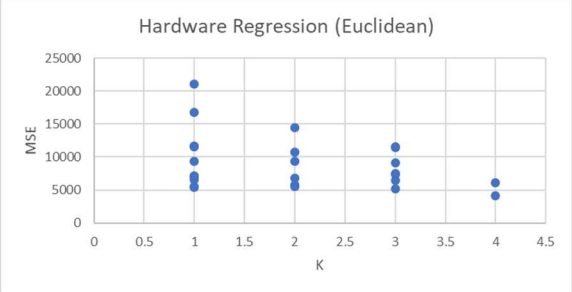
Overall, the algorithms performed well, though not as well as one would like, and in one case, not nearly as well as expected. An 85% classification accuracy certainly isn't bad, but it's also not particularly reliable, especially for real-world applications. The extremely poor performance of the condensed k-nearest neighbor algorithm was also a surprise, and much more work would be needed to investigate why it performs so poorly. The regression algorithm was also error prone, which is not itself a problem with the regression problem in general, but the magnitude of those errors was large enough to warrant better tuning and optimization. It is notable that it performed better on the Fire dataset than the Hardware dataset, considering that the Fire set is larger and contains a note about the difficulty of using it for regression. In general, the regression function needs more tuning and an upgrade in the form of a real kernel function. Additionally, there were a few methodological errors. The first was that the categorical time data present in the Fires dataset should have been handled differently or removed altogether. Simply replacing a month's abbreviation with its corresponding number allows the algorithm to run, but likely introduces some error. For example, January and December are only 1 month apart, the math should not imply that they are 12 months apart. There also wasn't a rigorous method in use to tune the *k* parameters. Several of the best-looking values were selected from the plots and tested, and that selection process could have been better controlled. It likely wouldn't have influenced the larger sources of error like the regression implementation, but it still could have been controlled better. Finally, there were some definite optimization issues. Running the algorithm took a long time, especially at the tuning stage, and illustrates one of the main drawbacks of these non-parametric techniques.

References

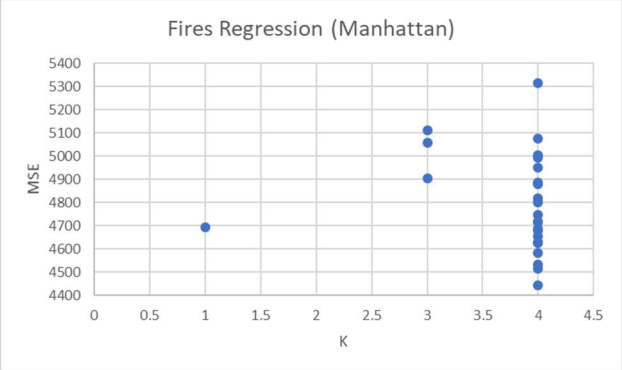
1) Altman, N. S. (2019). An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression, The American Statistician, Vol .46, No .3 (Aug, 1992), pp.175-185
2) Hart, P. E., "The Condensed Nearest Neighbor Rule," IEEE Trans. on Information Theory, Vol. IT-14, No. 3, pp 515-516 (May 1968)
3) Protein Localization Sites Creator and Maintainer: Kenta Nakai, Institue of Molecular and Cellular Biology, Osaka, University, 1-3 Yamada-oka, Suita 565 Japan, nakai@imcb.osaka-u.ac.jp, http://www.imcb.osaka-u.ac.jp/nakai/psort.html, Donor: Paul Horton (paulh@cs.berkeley.edu), Date: September, 1996, See also: yeast database
4) Image Segmentation data, Source Information, Creators: Vision Group, University of Massachusetts, Donor: Vision Group (Carla Brodley, brodley@cs.umass.edu), Date: November, 1990
5) Relative CPU Performance Data, Source Information, Creators: Phillip Ein-Dor and Jacob Feldmesser, Ein-Dor: Faculty of Management; Tel Aviv University; Ramat-Aviv; Tel Aviv, 69978; Israel, Donor: David W. Aha (aha@ics.uci.edu) (714) 856-8779, Date: October, 1987
6) P. Cortez and A. Morais. A Data Mining Approach to Predict Forest Fires using Meteorological Data. In J. Neves, M. F. Santos and J. Machado Eds., New Trends in Artificial Intelligence, Proceedings of the 13th EPIA 2007 - Portuguese Conference on Artificial Intelligence, December, Guimaraes, Portugal, pp. 512-523, 2007. APPIA, ISBN-13 978-989-95618-0-9. Available at: http://www.dsi.uminho.pt/~pcortez/fires.pdf
7) Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science

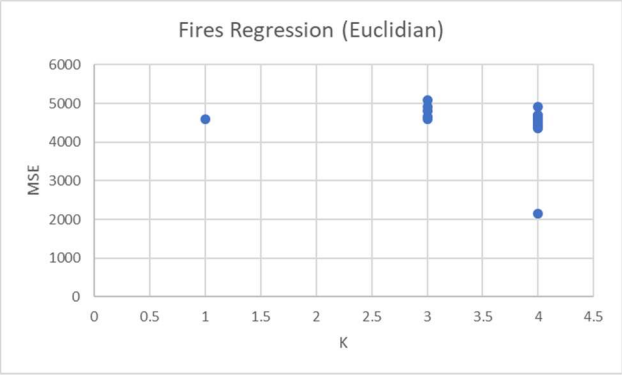
Appendix:

Regression:
Hardware:

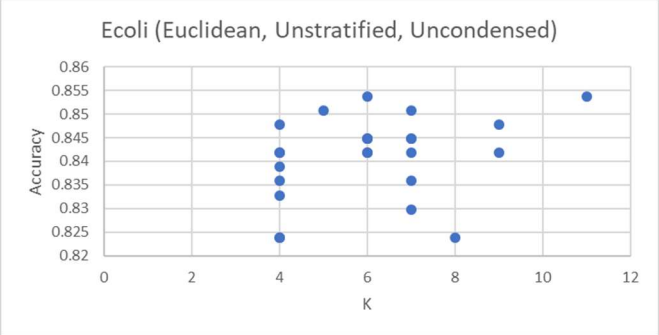


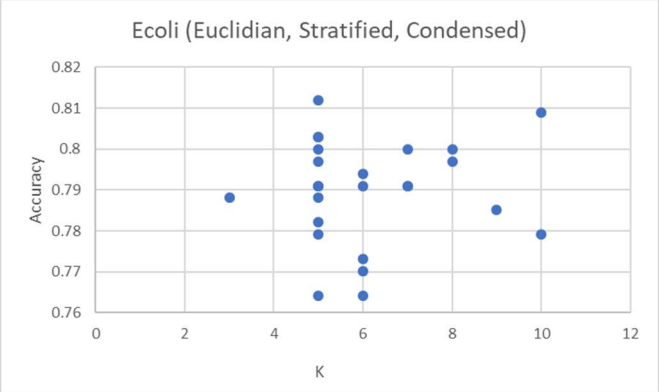
Fires:

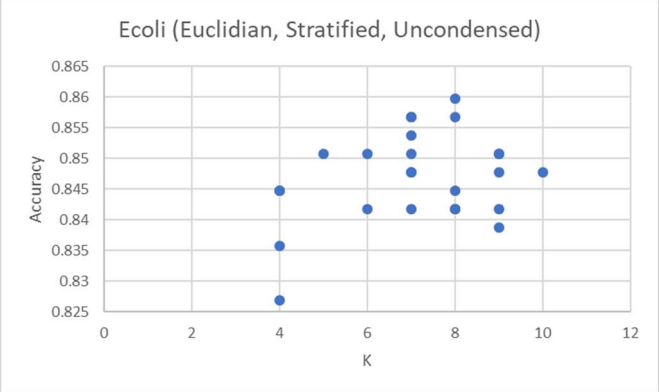


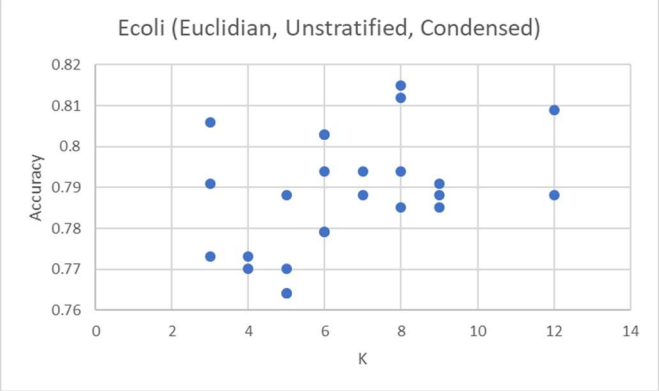


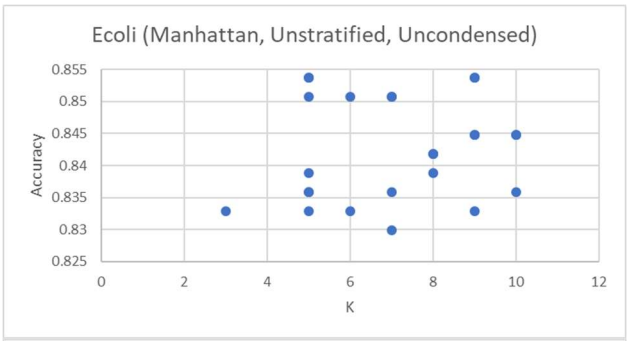
Classification:
Ecoli:

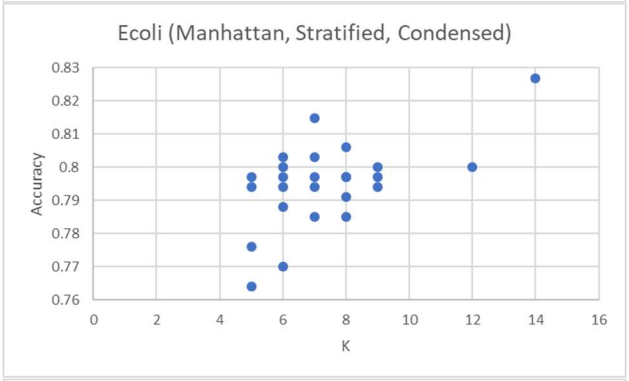


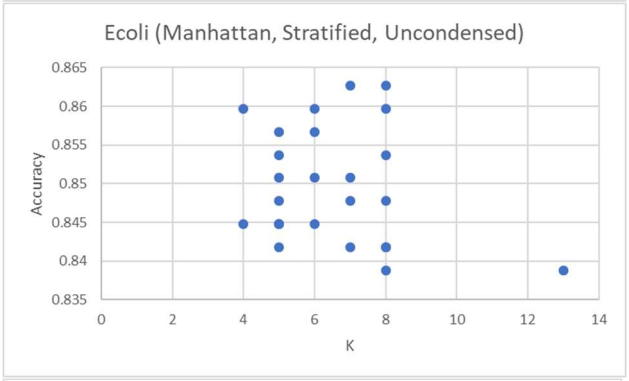







Ecoli (Manhattan, Unstratified, Uncondensed)
Accuracy
K


Ecoli (Manhattan, Stratified, Condensed)
Accuracy
K


Ecoli (Manhattan, Stratified, Uncondensed)
Accuracy
K

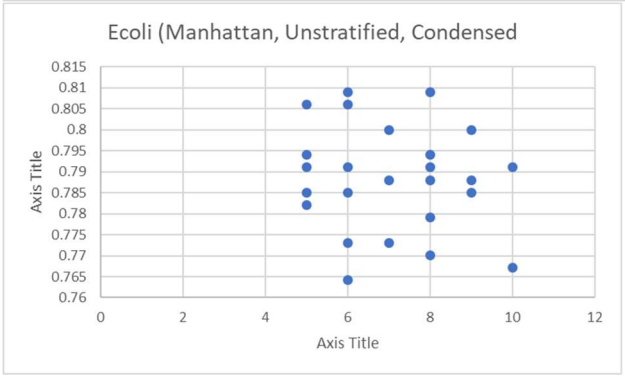

Ecoli (Manhattan, Unstratified, Condensed
Axis Title
Axis Title

Image:

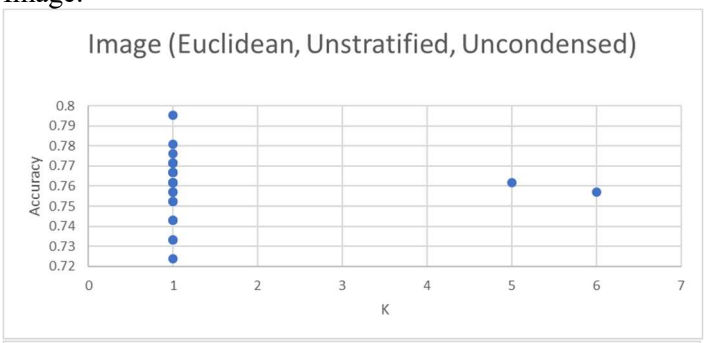


Image (Euclidean, Unstratified, Uncondensed)
Accuracy
K

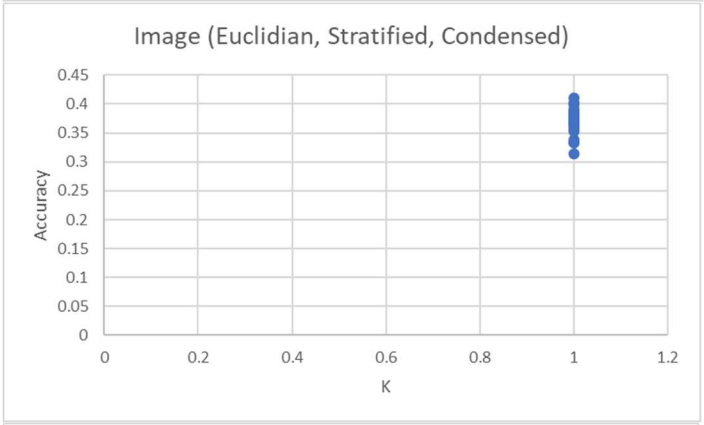


Image (Euclidian, Stratified, Condensed)
Accuracy
K

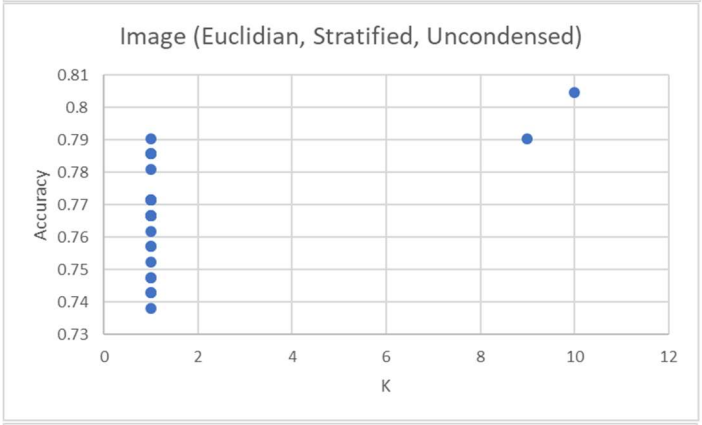


Image (Euclidian, Stratified, Uncondensed)
Accuracy
K

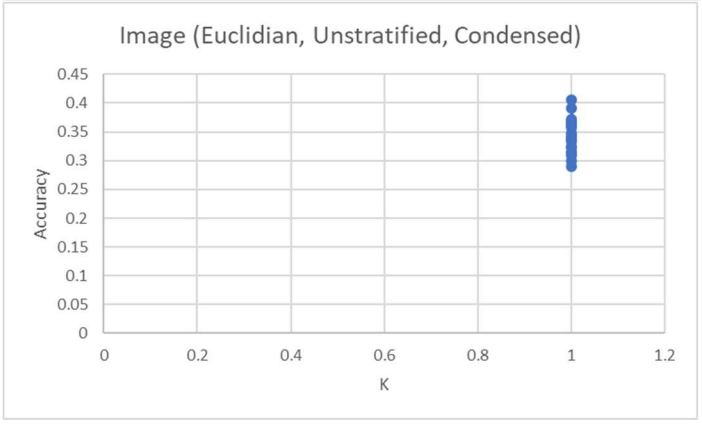


Image (Euclidian, Unstratified, Condensed)
Accuracy
K

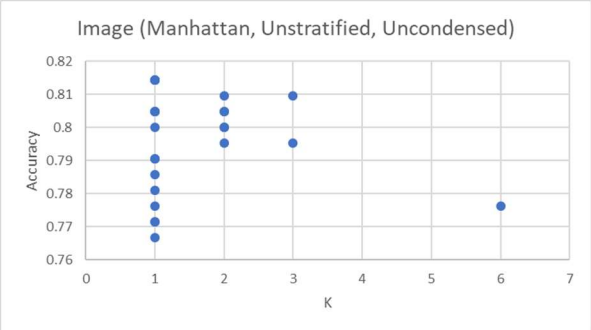


Image (Manhattan, Unstratified, Uncondensed)
Accuracy
K
0.82
0.81
0.8
0.79
0.78
0.77
0.76
0
1
2
3
4
5
6
7

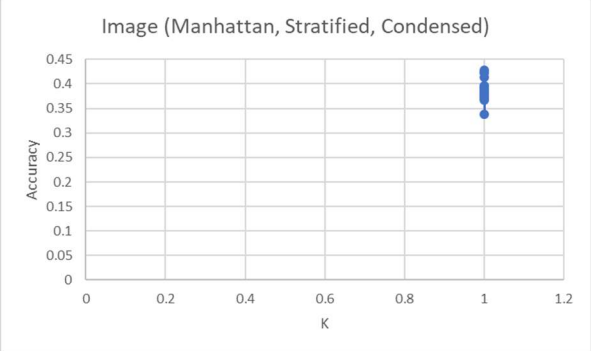


Image (Manhattan, Stratified, Condensed)
Accuracy
K
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.2
0.4
0.6
0.8
1
1.2

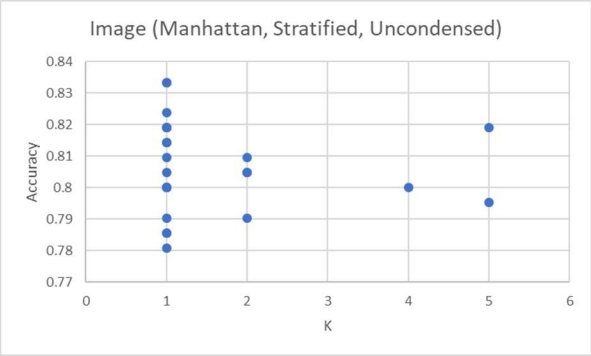


Image (Manhattan, Stratified, Uncondensed)
Accuracy
K
0.84
0.83
0.82
0.81
0.8
0.79
0.78
0.77
0
1
2
3
4
5
6

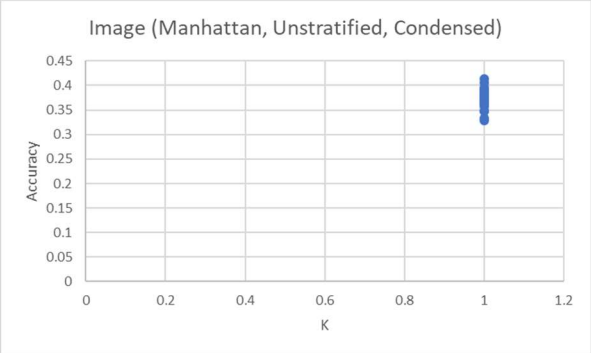


Image (Manhattan, Unstratified, Condensed)
Accuracy
K
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.2
0.4
0.6
0.8
1
1.2