FARMER’S MARKET LOCATOR

How to choose a spot for a farmer’s market,

without knowing a thing about them

How do you choose the right location for a new farmer’s market?

- Choosing the right location for a farmer’s market can be the key to its success or failure.
- Existing methods require specialized location and industry knowledge which is not always readily available. Additionally, existing methods suffer from a scalability problem. Would you be able to canvas an entire country in a day if you had to, how about an hour?
- If market organizers had a tool that would help them vet locations, it would allow them to avoid costly mistakes due to locating the market away from important infrastructure.
- We can train a machine learning model to predict good locations for markets by fingerprinting the shops around existing markets.

Data

- Kaggle: Dataset of farmers markets in the US with latitude and longitude
- Self-generated: List of coordinates reflecting areas where there are no farmer markets.
- Foursquare: Dataset of the amenities within 0.33 miles of any given coordinate

Data: Kaggle

	id	name	lon	lat	farmersMarket
37	1006234	4th Street Farmers Market	-105.07300	40.395401	1.0
114	1004070	Alamosa Farmers Market	-105.86523	37.468361	1.0
179	1001367	American National Bank Downtown Farmers Market...	-108.56400	39.068199	1.0
289	1005081	Arvada Farmers Market	-105.08145	39.800137	1.0
290	1009285	Arvada Five Parks Farmers Market	-105.15500	39.848801	1.0

- After removing unnecessary columns this is what was left.

Data: Self Generated

	id	name	lon	lat	farmersMarket
0	2012468	4th Street Farmers Market	-105.07300	40.351984	0.0
1	2008140	Alamosa Farmers Market	-105.86523	37.424944	0.0
2	2002734	American National Bank Downtown Farmers Market...	-108.56400	39.024782	0.0
3	2010162	Arvada Farmers Market	-105.08145	39.756720	0.0
4	2018570	Arvada Five Parks Farmers Market	-105.15500	39.805384	0.0

- I generated negative datapoints by offsetting coordinates from the existing markets.
- To generate these negative datapoints I copied the Kaggle dataset four times. On each copy I transformed the latitude by a set distance. I calculated latitude offsets for 3, 6, 20, and 60 miles south. This made the negative datapoint creation process easy and scalable.
- I multiplied the id number by an incrementing value to generate unique ids for each of the copies

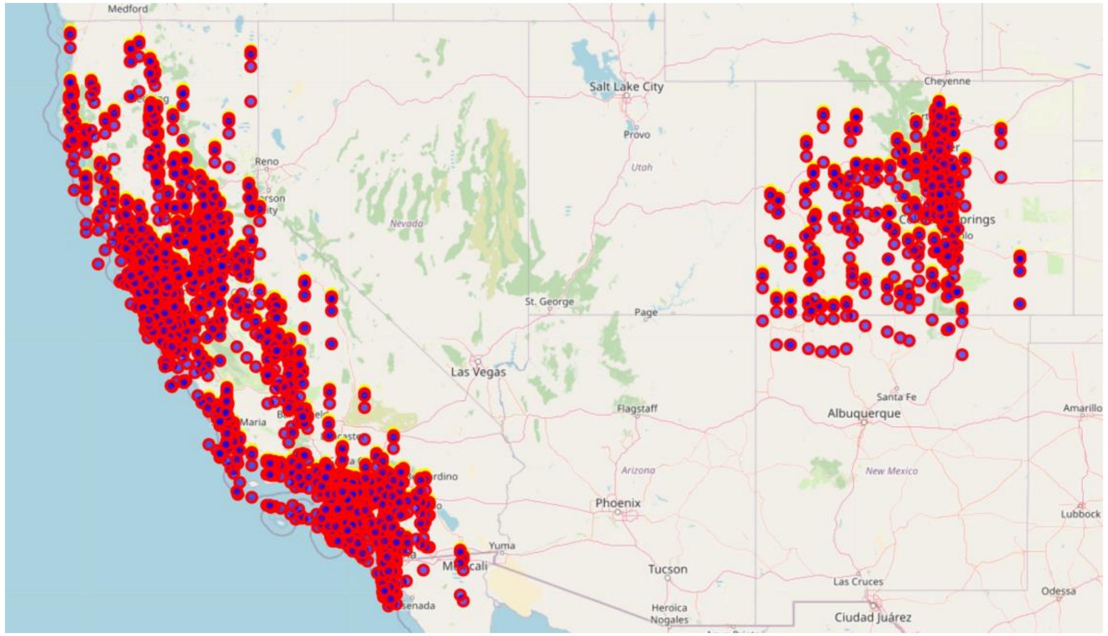
Data: Foursquare

	uniqueId	ATM	Accessories Store	Adult Boutique	Advertising Agency	Airport	American Restaurant	Animal Shelter	Antique Shop	Arcade	...	Vietnamese Restaurant	Vineyard	Warehouse Store	Whi
0	1000015	0.00000	0.0	0.0	0.0	0.0	0.107143	0.0	0.0	0.00000	...	0.0	0.0	0.0	0.00
1	1000133	0.01087	0.0	0.0	0.0	0.0	0.054348	0.0	0.0	0.01087	...	0.0	0.0	0.0	0.01
2	1000315	0.00000	0.0	0.0	0.0	0.0	0.090909	0.0	0.0	0.00000	...	0.0	0.0	0.0	0.00
3	1000417	0.00000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.00000	...	0.0	0.0	0.0	0.00
4	1000418	0.00000	0.0	0.0	0.0	0.0	0.000000	0.0	0.0	0.00000	...	0.0	0.0	0.0	0.00

- The data I pulled from Foursquare was transformed into this dataframe, tying the coordinate ID to a fingerprint of the shops around it.
- Originally the Foursquare api would encounter an error which would require me to repeat the api call from the beginning. After implementing error handling I was able to large amount of data reliably.

Data: Kaggle + Self Generated on Map

- Yellow dots represent existing farmer's markets
- Red dots represent the negative datapoints I generated using the coordinate offsets
- With additional data capacity I could canvas the entire Mountain West region.



Correlations

Positive Correlations

Out[173]:

farmersMarket	1.000000
Coffee Shop	0.324247
Mexican Restaurant	0.288229
Pizza Place	0.285166
New American Restaurant	0.271901
Ice Cream Shop	0.248064
Sandwich Place	0.240424
Farmers Market	0.226280
American Restaurant	0.224972
French Restaurant	0.221197
Bank	0.213920
Café	0.202202
Bakery	0.200001
Italian Restaurant	0.195070
Sushi Restaurant	0.193875
Thai Restaurant	0.185106
Bookstore	0.182806
Juice Bar	0.178401
Chinese Restaurant	0.177920
Grocery Store	0.173717

Negative Correlations

Out[174]:

Government Building	-0.012857
Airport Service	-0.013949
Outdoors & Recreation	-0.014083
Zoo Exhibit	-0.014322
Shabu-Shabu Restaurant	-0.014693
Zoo	-0.014782
Gym Pool	-0.015173
Moving Target	-0.015816
Nature Preserve	-0.016597
Home Service	-0.019875
Mountain	-0.020418
Beach	-0.020672
Boat or Ferry	-0.023672
Farm	-0.024393
Harbor / Marina	-0.025955
Construction & Landscaping	-0.026129
Campground	-0.027490
Scenic Lookout	-0.036005
Trail	-0.036967
uniqueId	-0.707059

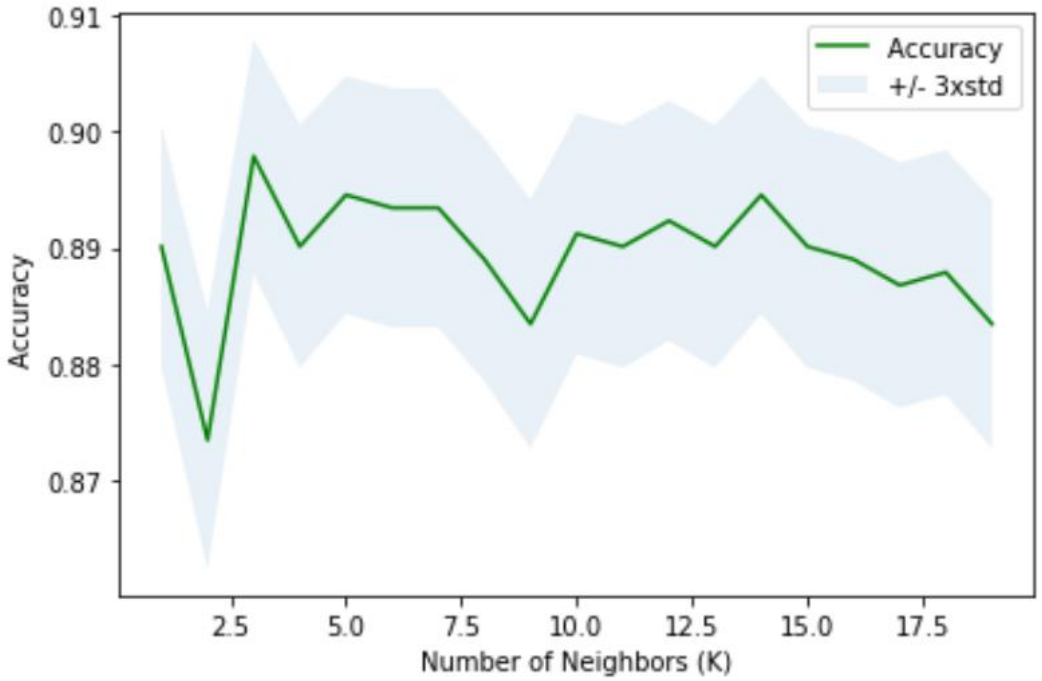
- After calculating the Pearson correlation between our features, and the presence of a farmer's market we can see several interesting things:
    - Farmer's markets tend to locate in neighborhoods with coffee shops, midrange restaurants, bakeries, banks and bookstores (ie Main Street)
    - There are weaker negative correlations locating markets away from infrastructure-less areas, as well as, water, zoos, and airports (ie The Boonies)
- To run the algorithm efficiently we applied a minimum correlation of 0.20, and -0.20 for consideration.
- If we wanted to make the algorithm less selective, we could lower the minimum cutoff point.

K Nearest Neighbor

- I chose to use the K Nearest Neighbor algorithm because I was:
  - Predicting a category
  - Using labeled data
  - I Had less than 100,000 samples
  - The data was numeric vs text
- I tested other algorithms, but KNeighbors scored the best
- I plotted each algorithm on the map and KNeighbors produced half as many coordinate matches as the others.
- The other algorithms were not selective enough, even though their test results are close to the KNeighbor results

	Jaccard (Higher is better)	F1-score (Higher is better)	LogLoss (Lower is better)
Algorithm			
KNN	0.890122	0.890946	3.795096
Decision Tree	0.889012	0.885776	3.833416
SVM	0.870144	0.864361	4.485094
LogisticRegression	0.807991	0.893800	6.631752

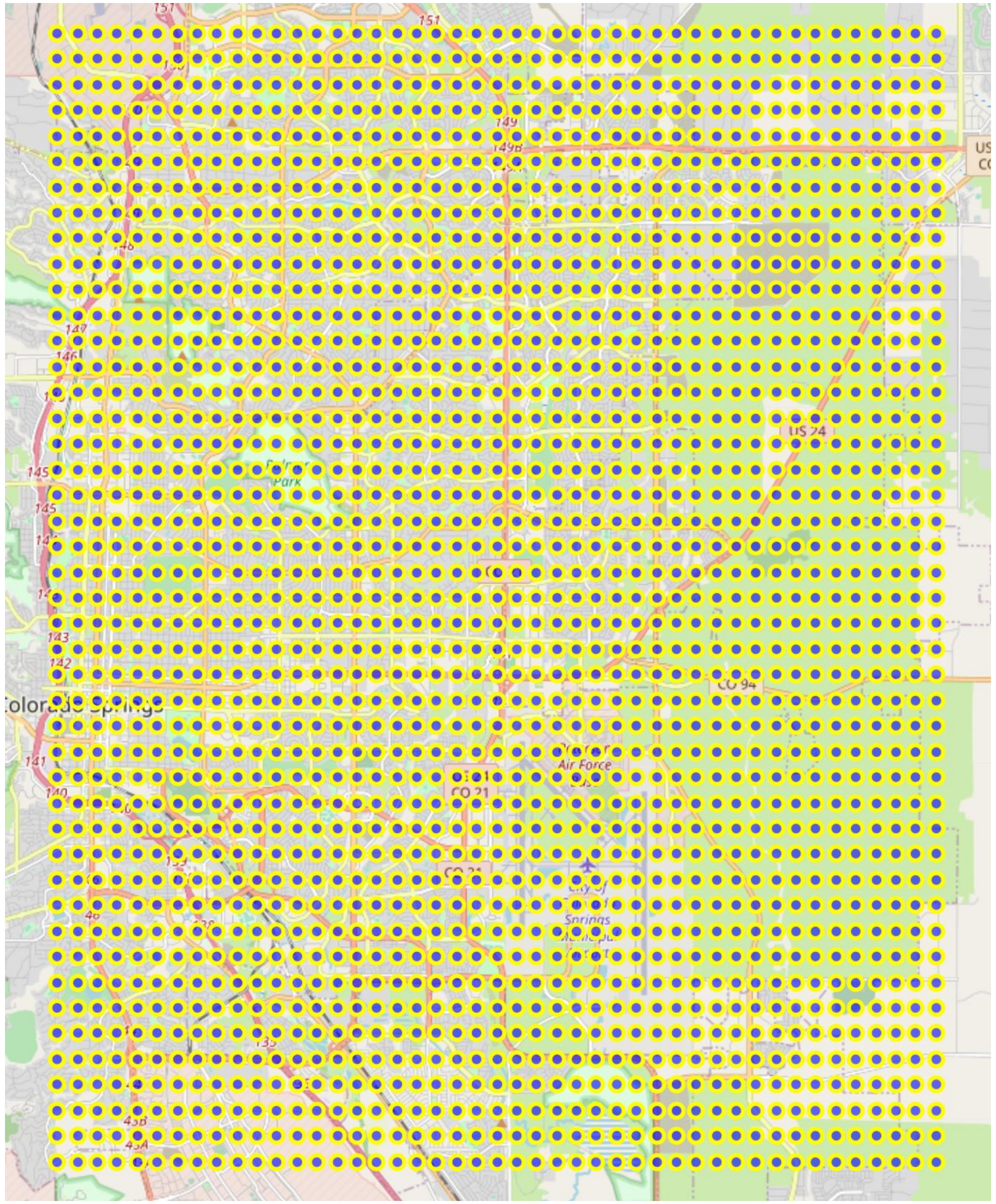
K Nearest Neighbor: Selecting the K Value



- I selected a K value of 5
  - Values higher than 5 that produced similar accuracies
  - Values lower than 5 may result in a program which does not do well filtering out outliers
- This means for each coordinate; the system will make a profile of the surrounding shops and look for five existing farmers markets with a similar local makeup. If it finds five matching datapoints it will assert that this location is a good spot for a market.
- The accuracy of the program is good, but we can always benefit from more training and testing data

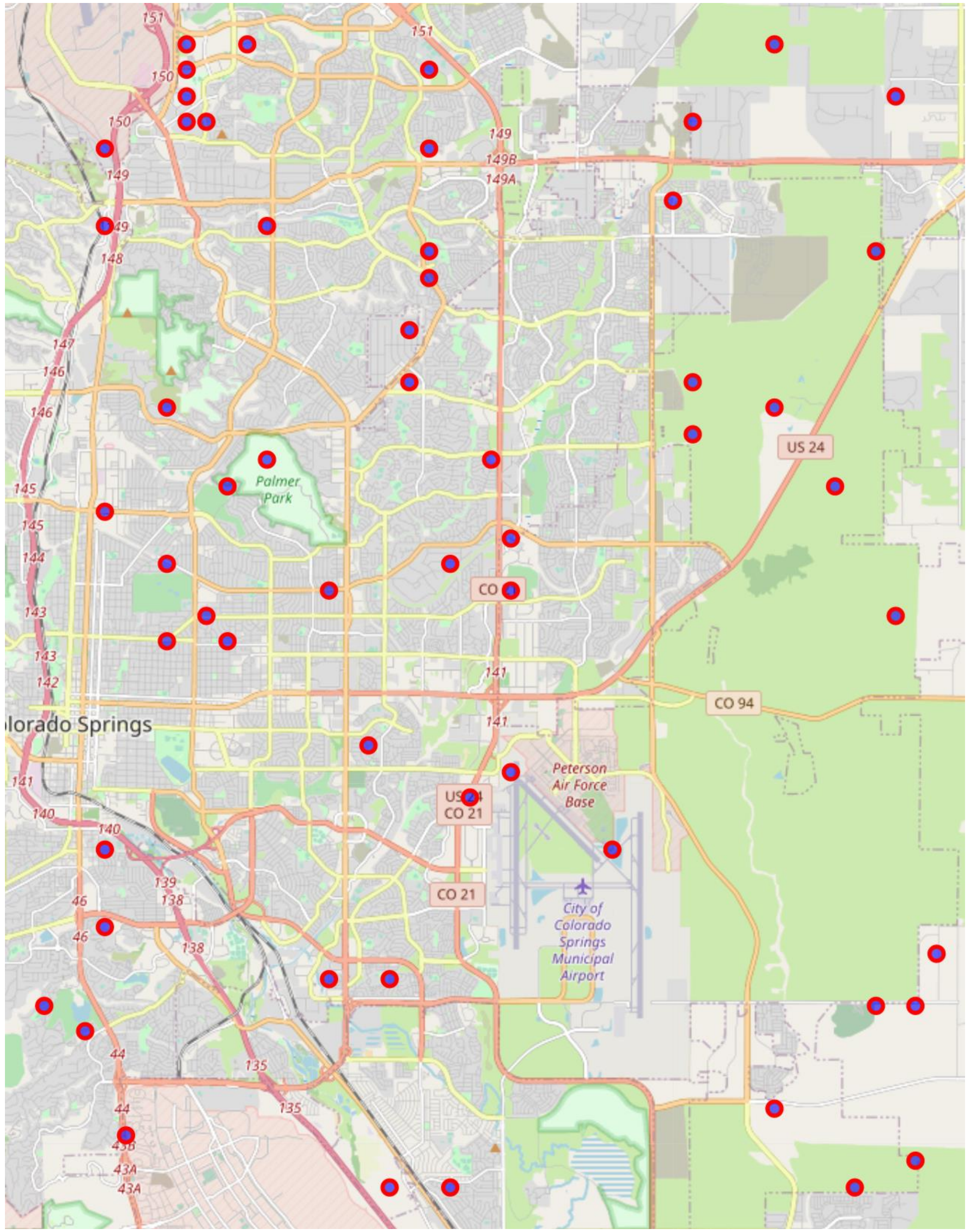
Prediction: Generating the coordinate grid

- In order to test the algorithm I generated a coordinate grid over a section of Colorado Springs with 2025 points.
- Each coordinate represents a bubble with a 0.33-mile diameter where the market should be located.
- The coordinates are then run through the ml program to identify if any are good locations for a farmer's market.
- When the program finds a match it means somewhere within 0.33 miles of that coordinate there is a good location for a market



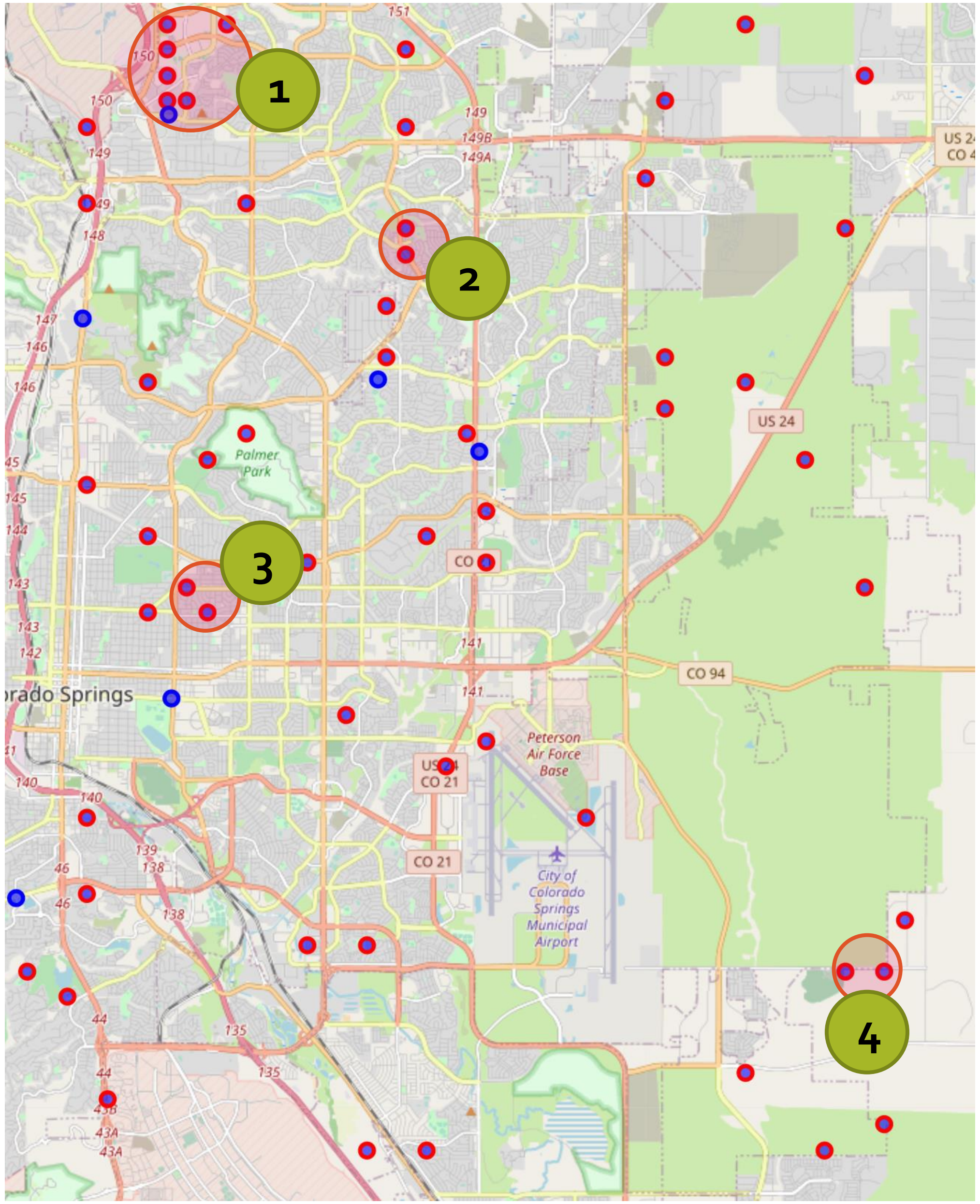
Prediction: Results

- There were 57 farmer's market matches out of 2025 analyzed coordinates.
- This match rate of about 3% makes the algorithm selective enough to be useful for locating new markets.



Prediction: Results

- To simplify our selection process we can cluster using two adjacent matches.
- Requiring two adjacent matches simplifies site selection and guarantees a higher level of local supporting infrastructure than an isolated match would.
- This program and methodology has helped us identify four potential farmer's markets locations.
- This program can also help us analyze positioning of existing markets, featured in blue. As we can see there are three current markets which do not have optimal supporting infrastructure.



Discussion

The results show promise! Without any prior knowledge, we were able to train a system to fingerprint existing markets and canvas an entire city for them.

A limitation that was overcome was the stability of the API, which required a mechanism to handle errors gracefully. This allowed us to max out the Foursquare API's daily call limits to pull as much data as possible.

By increasing our API subscription, or incrementally chunking data over a month, we could improve our correlation metrics and algorithm.

An entire state could be canvassed for locations efficiently if we queried Foursqaure first for all the venues shown to have high correlations. Then we could test only the areas around those coordinates.

The methodology described here could be used to locate other venue types, such as offices, coffee shops, or restaurants.