Team House Hunters (#40)

Taylor Gift, Katrina Green, Shelly Kunkle, Melanie Laffin

Final Report

April 20, 2019

Welcome to Casa Connect

Introduction

Problem Definition

Literature Surveys

Proposed Method

- Data Acquisition, Cleaning and Joining
  - Data Acquisition
  - Data Cleaning and Joining
- Clustering
  - K-means
  - Agglomerative
  - DBSCAN
  - Interactive
  - Final Choice
- Visualizations
  - Dataiku DSS Flow
  - Clustering Output/ Observations
- App/UI
- List of Innovations

Design of Experiments/ Evaluation

Distribution of Team Member’s Activities

Conclusion

Appendix A: Survey References

Introduction

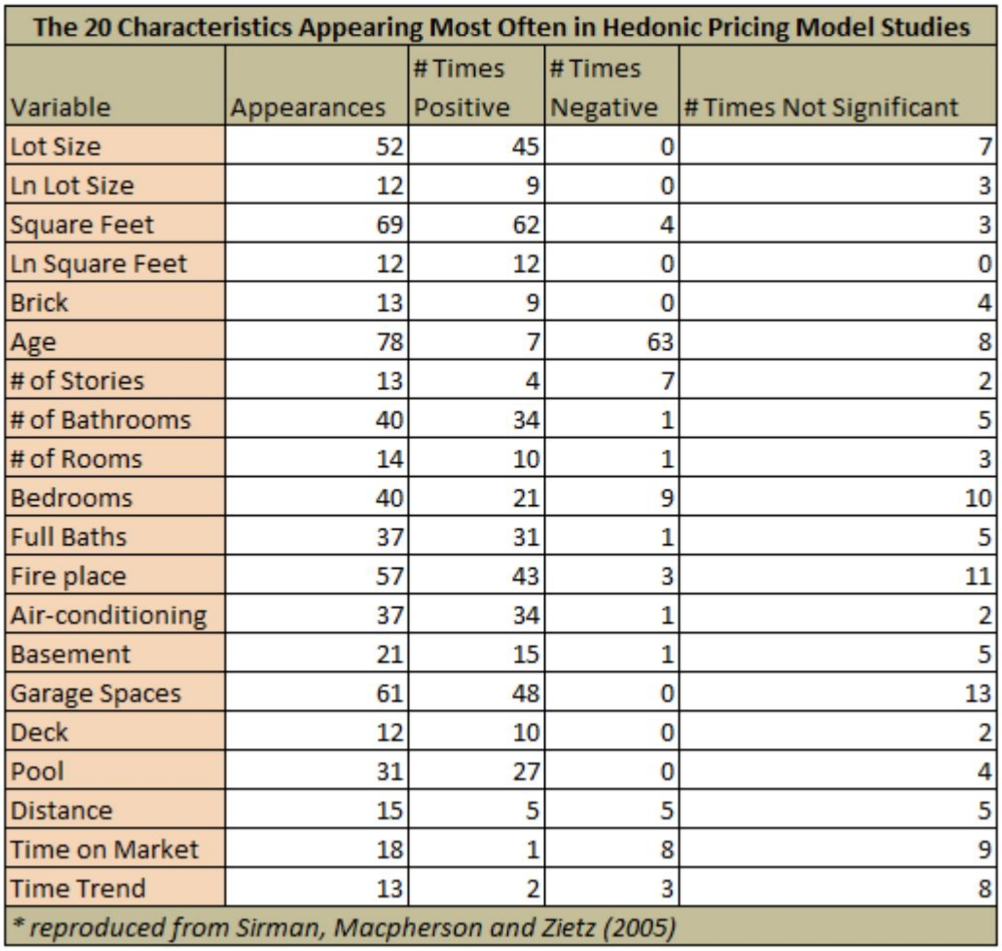
Moving has a lot of uncertainty. Can you find a home and neighborhood that matches your lifestyle? Our app, “Casa Connect,” aims to guide you to life in your new city. We aggregate multiple data points about your current home--beyond just listing details--and then apply clustering algorithms to find a home in the new city most likely to match your lifestyle.

Problem Definition

Existing apps for house shopping don’t compare current lifestyle to target lifestyle. They simply allow a user to search for a home based on basic criteria such as number of bedrooms, bathrooms, square footage and price. This is helpful but misses the mark of truly giving a user a picture into what the new neighborhood is like and if living there will satisfy their expectations. The burden falls on the mover to do their own heavy research or rely on information provided by unfamiliar real estate agents.

Literature Surveys

Buying a home is often an stressful and extensive process. Based on the profile of homebuyers and sellers (SK3), 44% of buyers start the process by looking online at properties for sale. Among those buyers who utilized the internet, 85% found detailed information to be useful in finding the perfect home. In a survey related to the selling side of the process (ML1), it can be found that the buyer may be at an informational disadvantage with agents. We also gained insight from a former Georgia Tech student’s thesis on how houses are valued and what features of a house are most commonly sought after (TG2). We mostly looked at hedonic pricing models when trying to understand how a home is valued.



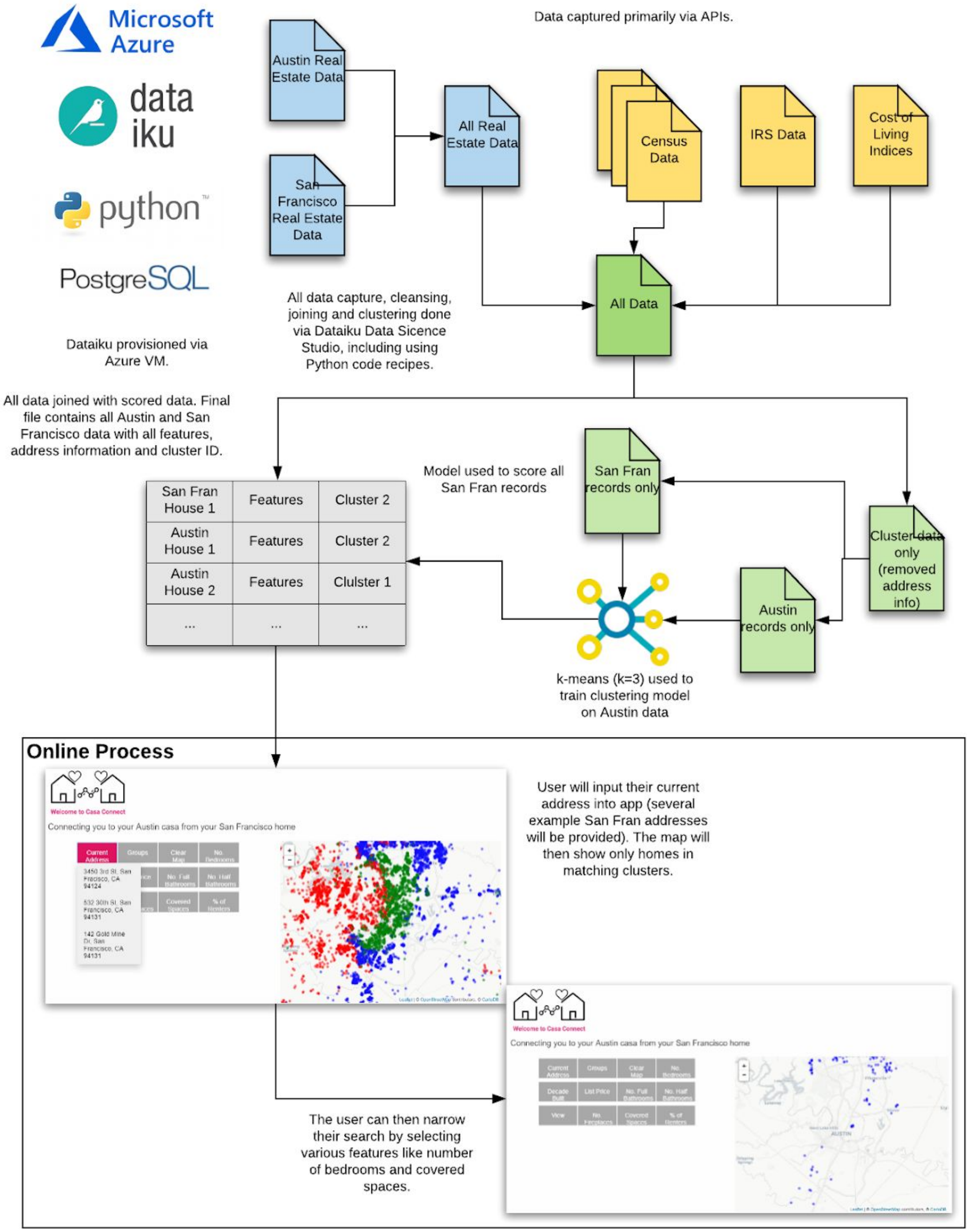
The 20 Characteristics Appearing Most Often in Hedonic Pricing Model Studies				
Variable	Appearances	# Times Positive	# Times Negative	# Times Not Significant
Lot Size	52	45	0	7
Ln Lot Size	12	9	0	3
Square Feet	69	62	4	3
Ln Square Feet	12	12	0	0
Brick	13	9	0	4
Age	78	7	63	8
# of Stories	13	4	7	2
# of Bathrooms	40	34	1	5
# of Rooms	14	10	1	3
Bedrooms	40	21	9	10
Full Baths	37	31	1	5
Fire place	57	43	3	11
Air-conditioning	37	34	1	2
Basement	21	15	1	5
Garage Spaces	61	48	0	13
Deck	12	10	0	2
Pool	31	27	0	4
Distance	15	5	5	5
Time on Market	18	1	8	9
Time Trend	13	2	3	8
** reproduced from Sirman, Macpherson and Zietz (2005)*				

Table 2.1 in TG2 - This table helped us determine the relevant data we should use to build our database.

We referenced Chapter 2 of Urban Morphology and Housing Market (TG3) to derive some of the most commonly applied methods of housing price evaluation. With Casa Connect, we hope to join together the data that would help the buyer be most informed to find the right home and neighborhood. The Zillow case study (ML3) discussed the strategy of combining the right information including in some cases user-contributed facts in order to complete the home listing. While it would have been interesting to include the walkability scores as another way for the buyer to be informed (KG1), the sources available with this information are not free for public use.

Finally to make Casa Connect a reality, we researched the techniques behind unsupervised learning (KG2) and further found surveys on user clustering with recommendation systems which can be useful in finding the perfect home (TG1, ML2). We plan on using these recommendation algorithms and clustering techniques to provide users with a proprietary or standardized similarity score that will return home listings that are most similar to the user's preferences.

Proposed Method



Data captured primarily via APIs.

Dataiku provisioned via Azure VM.

All data capture, cleansing, joining and clustering done via Dataiku Data Sicence Studio, including using Python code recipes.

All data joined with scored data. Final file contains all Austin and San Francisco data with all features, address information and cluster ID.

Model used to score all San Fran records

k-means (k=3) used to train clustering model on Austin data

San Fran House 1	Features	Cluster 2
Austin House 1	Features	Cluster 2
Austin House 2	Features	Clulster 1
...	...	...

User will input their current address into app (several example San Fran addresses will be provided). The map will then show only homes in matching clusters.

The user can then narrow their search by selecting various features like number of bedrooms and covered spaces.

Welcome to Casa Connect

Connecting you to your Austin casa from your San Francisco home

3450 3rd St. San Fracisco, CA 94124

532 30th St. San Francisco, CA 94131

142 Gold Mine Dr, San Francisco, CA 94131

Data Acquisition, Cleaning and Joining

Data Acquisition

We utilized four different types of data for our project: real estate data, census data, IRS data and cost of living index data.

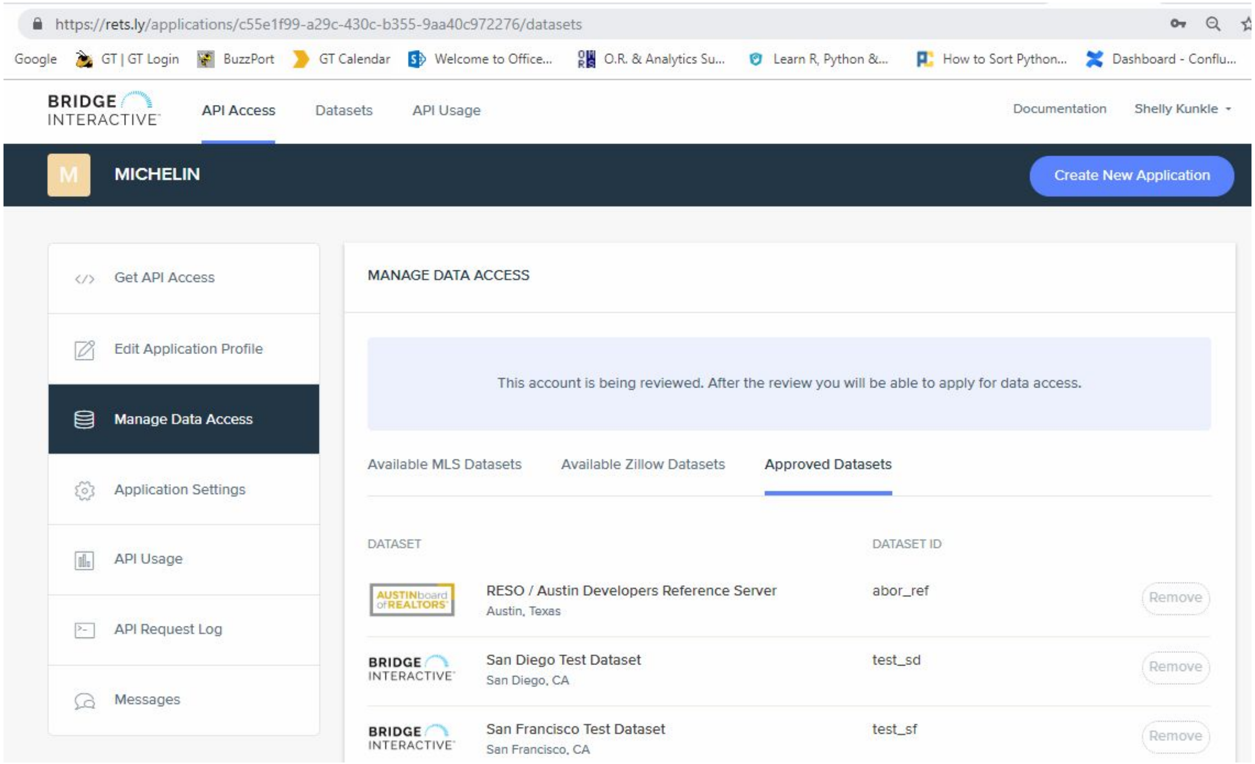
Real Estate Data

We initially tried to utilize Zillow data but realized two major issues with it.

1. No street addresses so this would limit our ability to display the houses in our UI in the way that we wanted.
2. The number of features was minimal, impacting the effectiveness of our clustering algorithm

We researched how to access listing records from the MLS (Multiple Listing Service) system, the US standard for real estate data. However, there are 30 regions with their own databases available only to real estate agents.

We found a set of real estate data in Austin, TX, specifically made available to the developer community. It is accessed through the Bridge API website (https://rets.ly/login). We received access to this data and access to two smaller test datasets in San Diego and San Francisco. See Figure below.



MANAGE DATA ACCESS

This account is being reviewed. After the review you will be able to apply for data access.

DATASET		DATASET ID	
AUSTINboard of REALTORS	RESO / Austin Developers Reference Server Austin, Texas	abor_ref	Remove
BRIDGE INTERACTIVE	San Diego Test Dataset San Diego, CA	test_sd	Remove
BRIDGE INTERACTIVE	San Francisco Test Dataset San Francisco, CA	test_sf	Remove

We accessed data through the API and stored it in a dataframe saved to a database within Dataiku. Because the API only allowed us to retrieve 100 records at a time, we had to find a way to utilize various API parameters to retrieve enough records. Out of a possible 321,175 Austin records, we filtered on Residential properties and retrieved 190,584 records. See file, ‘austin_api.py’ for the code for an API call.

We also pulled the San Diego and San Francisco records, but only San Francisco data was used. After filtering on Residential properties and removing some bad data, we ended up with 3,975 San Francisco properties. Our example is someone moving from San Francisco, CA to Austin, TX.

Census Data

After researching various free government datasets, we found the most relevant data in the Decennial Census Data (https://www.census.gov/data/developers/data-sets/decennial-census.html). We used the latest data, which is from 2010, organized by postal code. We obtained the data as follows:

- Created a listing of unique zip codes in our Austin and San Francisco datasets.
- Queried via the decennial census API for several data points which we translated into metrics usable by our clustering model

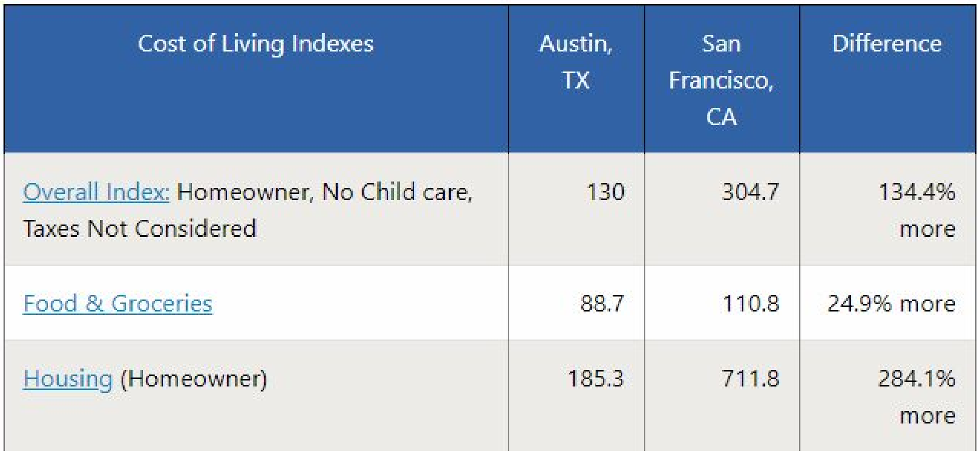
We called the APIs iteratively through all the zip codes and then saved the data to a dataframe and then a database in Dataiku. See ‘census_urban_rural_data.py’ for the code used to retrieve the urban percent metric. Other metrics were obtained similarly.

IRS Data

We also found IRS tax return data available at the zip code level by state for download in csv format. We downloaded CA and TX tax return data and then used the minimum value in the gross taxable income range to calculate the Avg Min Gross Income per return per zip code. This was stored into a database in Dataiku as well.

Cost of Living Index Data

Because we had two fields in our data that were monetary, we knew we need to adjust them if we wanted to properly compare houses in San Francisco with those in Austin. We found cost of living indices for at BestPlaces.net (https://www.bestplaces.net/cost-of-living/austin-tx/san-francisco-ca/50000). We captured the overall cost of living index and the housing index for both cities. See Figure below.

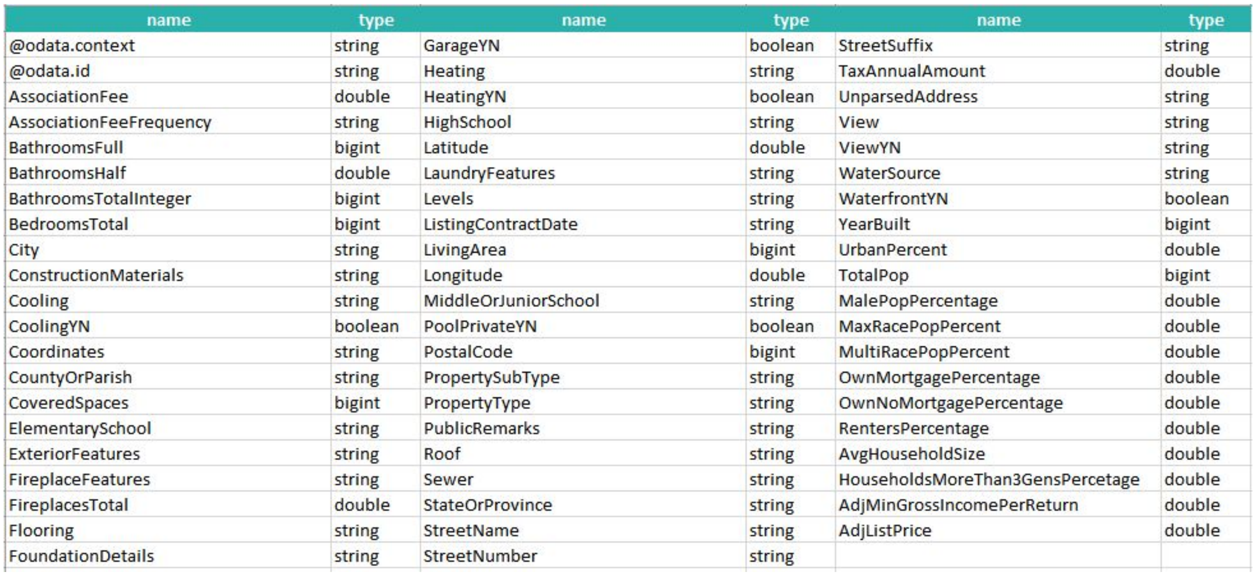


Cost of Living Indexes	Austin, TX	San Francisco, CA	Difference
Overall Index: Homeowner, No Child care, Taxes Not Considered	130	304.7	134.4% more
Food & Groceries	88.7	110.8	24.9% more
Housing (Homeowner)	185.3	711.8	284.1% more

Data Cleaning and Joining

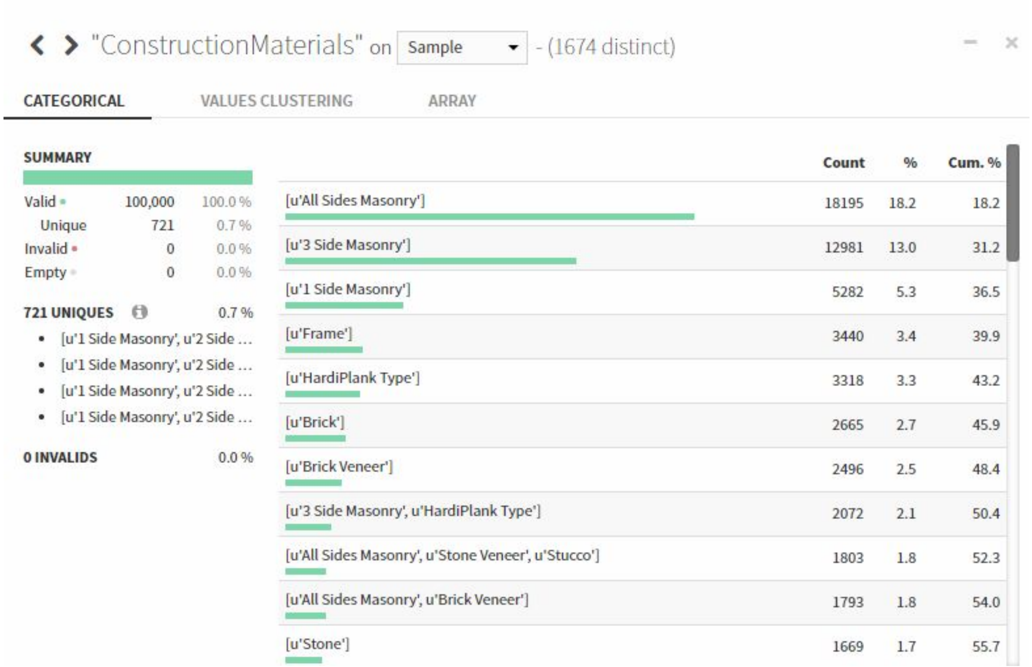
The real estate data required the most cleaning. Initially, we captured the features that seemed interesting for either clustering or display but we had to eliminate some because they were sparse. We only included features that were common to both Austin and San Francisco datasets.

Next, we joined the census, IRS and cost of living index data to the Austin + San Francisco real estate data to create one large dataset, casa_final. Casa_final becomes the source for all possible fields we would need for both clustering and display. It is 194,470 rows and 62 columns. See casa_final.csv in code folder. Here is the schema of casa_final:



name	type	name	type	name	type
@odata.context	string	GarageYN	boolean	StreetSuffix	string
@odata.id	string	Heating	string	TaxAnnualAmount	double
AssociationFee	double	HeatingYN	boolean	UnparsedAddress	string
AssociationFeeFrequency	string	HighSchool	string	View	string
BathroomsFull	bigint	Latitude	double	ViewYN	string
BathroomsHalf	double	LaundryFeatures	string	WaterSource	string
BathroomsTotalInteger	bigint	Levels	string	WaterfrontYN	boolean
BedroomsTotal	bigint	ListingContractDate	string	YearBuilt	bigint
City	string	LivingArea	bigint	UrbanPercent	double
ConstructionMaterials	string	Longitude	double	TotalPop	bigint
Cooling	string	MiddleOrJuniorSchool	string	MalePopPercentage	double
CoolingYN	boolean	PoolPrivateYN	boolean	MaxRacePopPercent	double
Coordinates	string	PostalCode	bigint	MultiRacePopPercent	double
CountyOrParish	string	PropertySubType	string	OwnMortgagePercentage	double
CoveredSpaces	bigint	PropertyType	string	OwnNoMortgagePercentage	double
ElementarySchool	string	PublicRemarks	string	RentersPercentage	double
ExteriorFeatures	string	Roof	string	AvgHouseholdSize	double
FireplaceFeatures	string	Sewer	string	HouseholdsMoreThan3GensPercetage	double
FireplacesTotal	double	StateOrProvince	string	AdjMinGrossIncomePerReturn	double
Flooring	string	StreetName	string	AdjListPrice	double
FoundationDetails	string	StreetNumber	string		

The last step of data preparation was to prepare the dataset needed for clustering. We trimmed casa_final by removing categorical features that were either not differentiating or that added too many dummy variables because of the complexity of the attribute. We also eliminated some rows were sparse. See figure below for the distribution of ‘Construction Materials.’



SUMMARY

Valid	100,000	100.0 %
Unique	721	0.7 %
Invalid	0	0.0 %
Empty	0	0.0 %

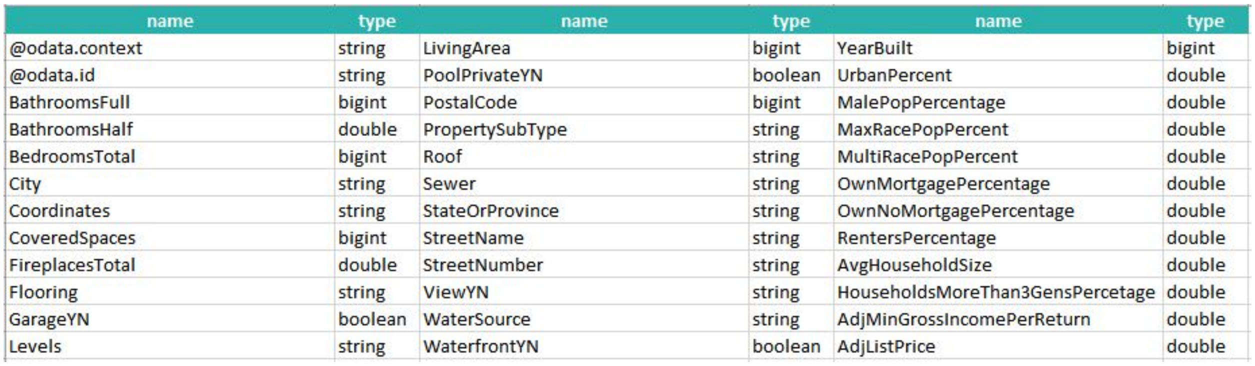
721 UNIQUES 0.7 %

- [u'1 Side Masonry', u'2 Side ...
- [u'1 Side Masonry', u'2 Side ...
- [u'1 Side Masonry', u'2 Side ...
- [u'1 Side Masonry', u'2 Side ...

0 INVALIDS 0.0 %

	Count	%	Cum. %
[u'All Sides Masonry']	18195	18.2	18.2
[u'3 Side Masonry']	12981	13.0	31.2
[u'1 Side Masonry']	5282	5.3	36.5
[u'Frame']	3440	3.4	39.9
[u'HardiPlank Type']	3318	3.3	43.2
[u'Brick']	2665	2.7	45.9
[u'Brick Veneer']	2496	2.5	48.4
[u'3 Side Masonry', u'HardiPlank Type']	2072	2.1	50.4
[u'All Sides Masonry', u'Stone Veneer', u'Stucco']	1803	1.8	52.3
[u'All Sides Masonry', u'Brick Veneer']	1793	1.8	54.0
[u'Stone']	1669	1.7	55.7

This process created the casa_cluster dataset which we then separated between casa_cluster_austin_only and casa_cluster_sf_only. We then performed our clustering on casa_cluster_austin_only and used that model to score the records in casa_cluster_sf_only. Casa_cluster_austin_only contains 190,495 rows and 36 columns. See casa_cluster_austin_only.csv in code folder. Here is a listing of its schema.



name	type	name	type	name	type
@odata.context	string	LivingArea	bigint	YearBuilt	bigint
@odata.id	string	PoolPrivateYN	boolean	UrbanPercent	double
BathroomsFull	bigint	PostalCode	bigint	MalePopPercentage	double
BathroomsHalf	double	PropertySubType	string	MaxRacePopPercent	double
BedroomsTotal	bigint	Roof	string	MultiRacePopPercent	double
City	string	Sewer	string	OwnMortgagePercentage	double
Coordinates	string	StateOrProvince	string	OwnNoMortgagePercentage	double
CoveredSpaces	bigint	StreetName	string	RentersPercentage	double
FireplacesTotal	double	StreetNumber	string	AvgHouseholdSize	double
Flooring	string	ViewYN	string	HouseholdsMoreThan3GensPercetage	double
GarageYN	boolean	WaterSource	string	AdjMinGrossIncomePerReturn	double
Levels	string	WaterfrontYN	boolean	AdjListPrice	double

Clustering

We experimented with the following four clustering methods.

K-means

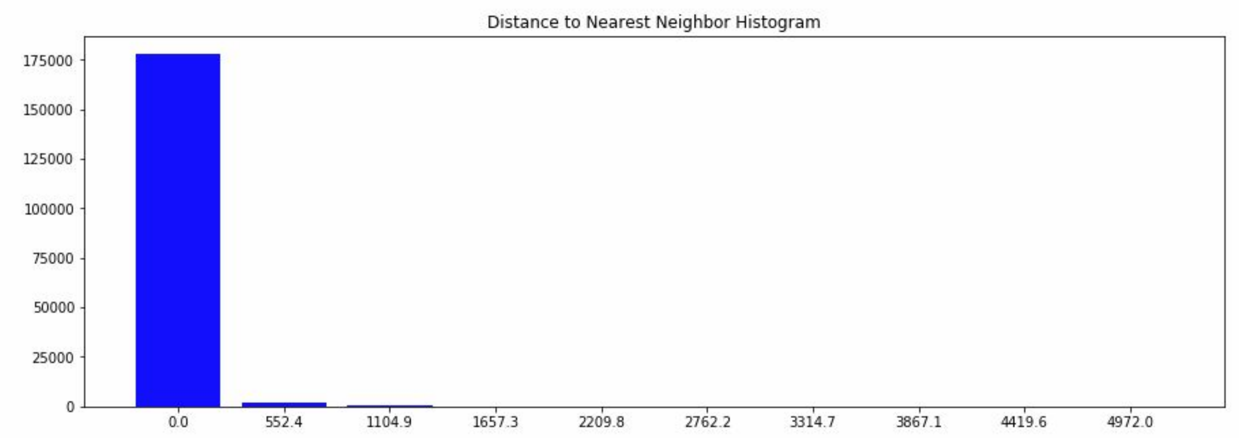
K-means was fast and easy to run. One of the biggest weaknesses of k-means is that you have to pre-select k, the number of clusters. In our case, we knew that we didn't want k to be greater than 5 because of the desired user experience of our app. Showing 5 different clusters on a map would be overwhelming for the user. Also, because k-means is fast, running multiple models with different k values was easy. We ran models for k = 3, 4 and 5 and found the best silhouette value (0.156) for k = 3. The run time for this model was 1 minute and 49 seconds.

Agglomerative

Hierarchical models don't do well with big datasets so we predicted this would not be ideal. Our VM is spec'd with 16 GB of RAM and we didn't have the budget to go higher so this model was out.

DBSCAN

We spent some time trying to run a DBSCAN model but in the end ran into memory errors calculating silhouette. We first had to determine epsilon, the max distance that another point can be away and still be in the same cluster. In order to do this we used a nearest neighbor algorithm and looked at the results in a histogram. We first had to eliminate all the categorical variables for this exercise because of the time it would take to create all the dummy variables (Dataiku did this for us automatically for k-means and interactive). We settled on 1700 for epsilon. See figure below.



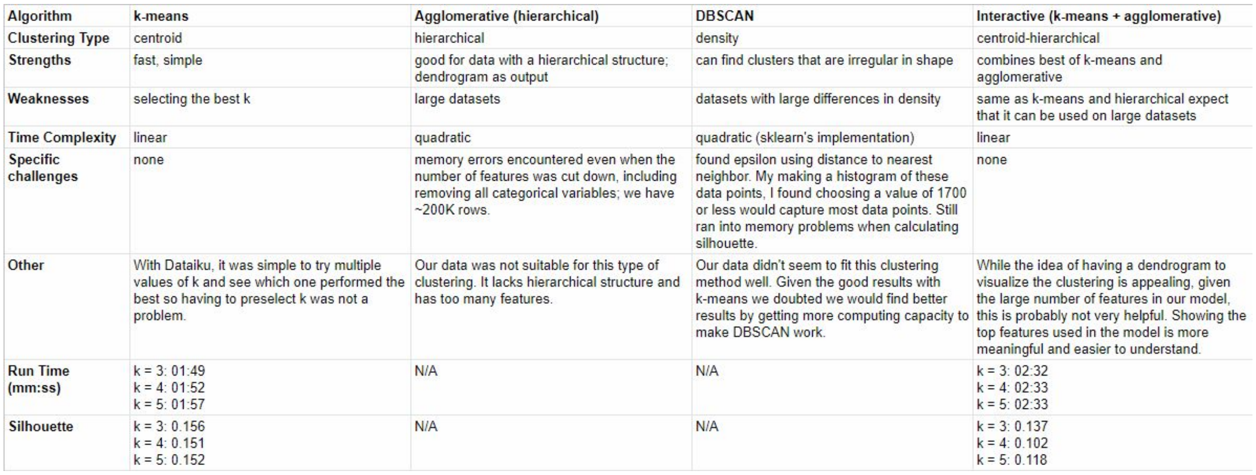
We ran the algorithm with a minimum of 500 data points per cluster and the clustering model ran--but when we tried to calculate silhouette, we encountered memory issues. In the end, we abandoned this model. The features in our dataset have different densities so we suspect this is the underlying issue.

Interactive

Interactive is an option available in Dataiku DSS. It is a way to use hierarchical clustering on large datasets.

Interactive worked and ran pretty quickly. It was a about 35% longer to run than k-means and it's silhouette values were 12% lower than k-means. Again, because having a dendrogram wasn't of importance to us, so while this was interesting, we ended up choosing k-means.

Below is a summary of our findings:



Algorithm	k-means	Agglomerative (hierarchical)	DBSCAN	Interactive (k-means + agglomerative)
Clustering Type	centroid	hierarchical	density	centroid-hierarchical
Strengths	fast, simple	good for data with a hierarchical structure; dendrogram as output	can find clusters that are irregular in shape	combines best of k-means and agglomerative
Weaknesses	selecting the best k	large datasets	datasets with large differences in density	same as k-means and hierarchical expect that it can be used on large datasets
Time Complexity	linear	quadratic	quadratic (sklearn's implementation)	linear
Specific challenges	none	memory errors encountered even when the number of features was cut down, including removing all categorical variables; we have ~200K rows.	found epsilon using distance to nearest neighbor. My making a histogram of these data points, I found choosing a value of 1700 or less would capture most data points. Still ran into memory problems when calculating silhouette.	none
Other	With Dataiku, it was simple to try multiple values of k and see which one performed the best so having to preselect k was not a problem.	Our data was not suitable for this type of clustering. It lacks hierarchical structure and has too many features.	Our data didn't seem to fit this clustering method well. Given the good results with k-means we doubted we would find better results by getting more computing capacity to make DBSCAN work.	While the idea of having a dendrogram to visualize the clustering is appealing, given the large number of features in our model, this is probably not very helpful. Showing the top features used in the model is more meaningful and easier to understand.

Run Time (mm:ss)	k = 3: 01:49
	k = 4: 01:52
	k = 5: 01:57
		N/A	N/A	k = 3: 02:32
				k = 4: 02:33
				k = 5: 02:33

Silhouette	k = 3: 0.156
	k = 4: 0.151
	k = 5: 0.152
		N/A	N/A	k = 3: 0.137
				k = 4: 0.102
				k = 5: 0.118

Final Choice

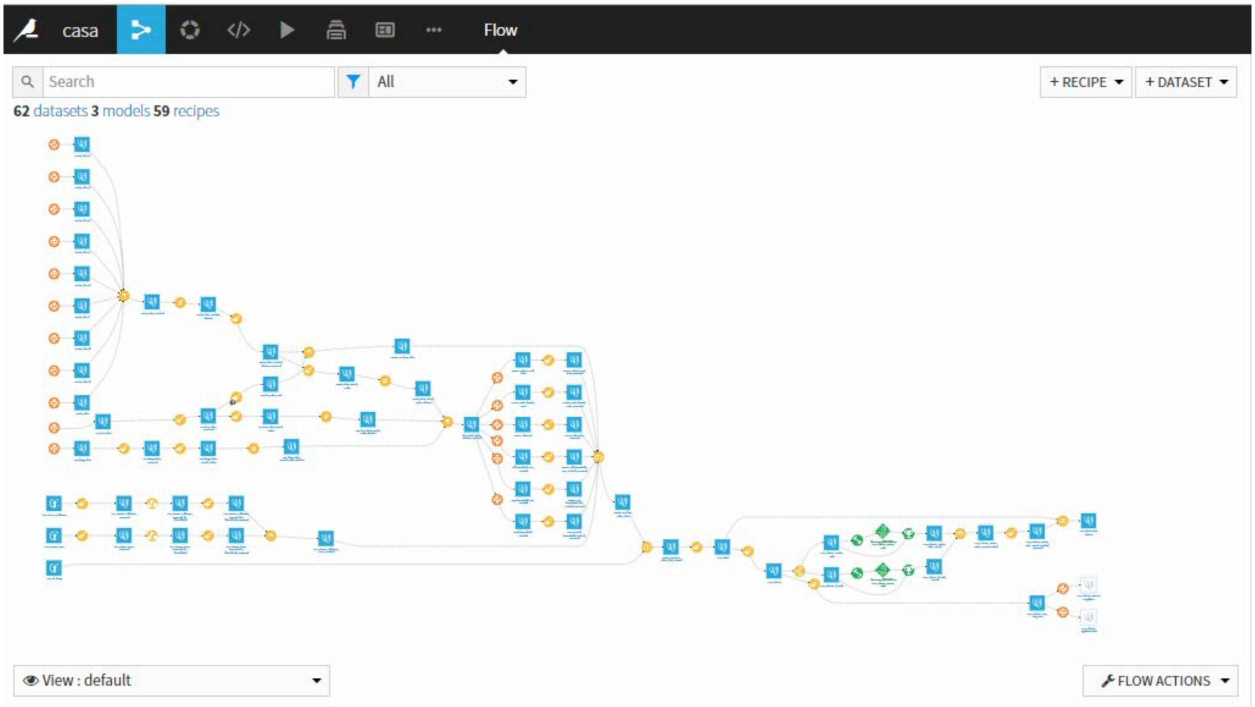
K-means with k = 3 won out in terms of speed and silhouette. It ran in 1 minute, 49 seconds and had a silhouette value of 0.156.

As mentioned previously, Dataiku automatically generates dummy variables for categorical variables so while we started with 36 columns, the k-means model created a total of 171 features. See 'Cluster_casa_cluster_austin_only.ipynb' for code that created the model.

Visualizations

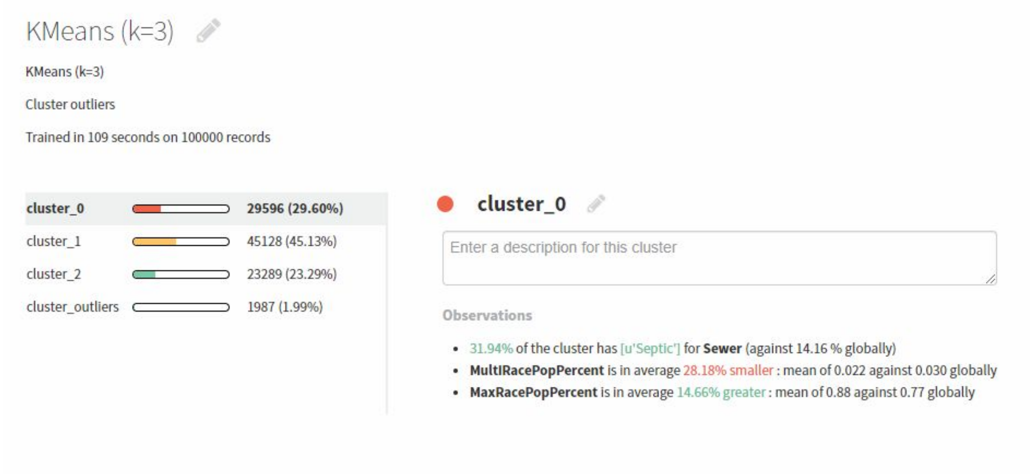
Dataiku DSS Flow

Below is the entire dataflow from data ingestion to model development and scoring. Note that 95% of this is data preparation! The green parts are the model training and scoring.



Clustering Output/Observations

Summary
This is overall summary of our model.



KMeans (k=3)

KMeans (k=3)

Cluster outliers

Trained in 109 seconds on 100000 records

cluster_0	**29596 (29.60%)**
cluster_1	45128 (45.13%)
cluster_2	23289 (23.29%)
cluster_outliers	1987 (1.99%)

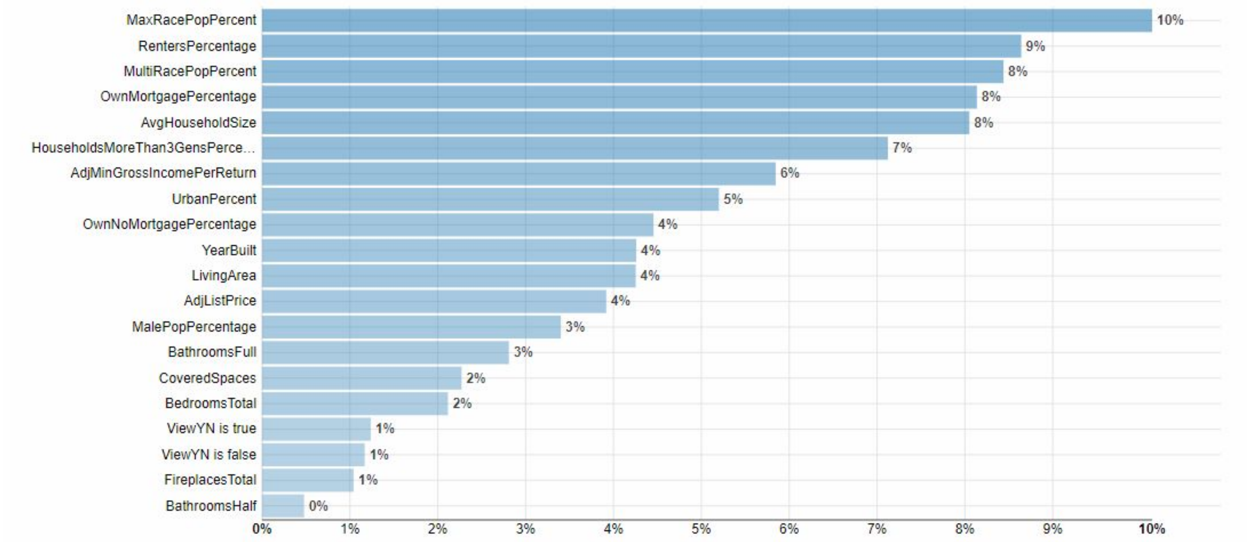
● cluster_0

Enter a description for this cluster

Observations

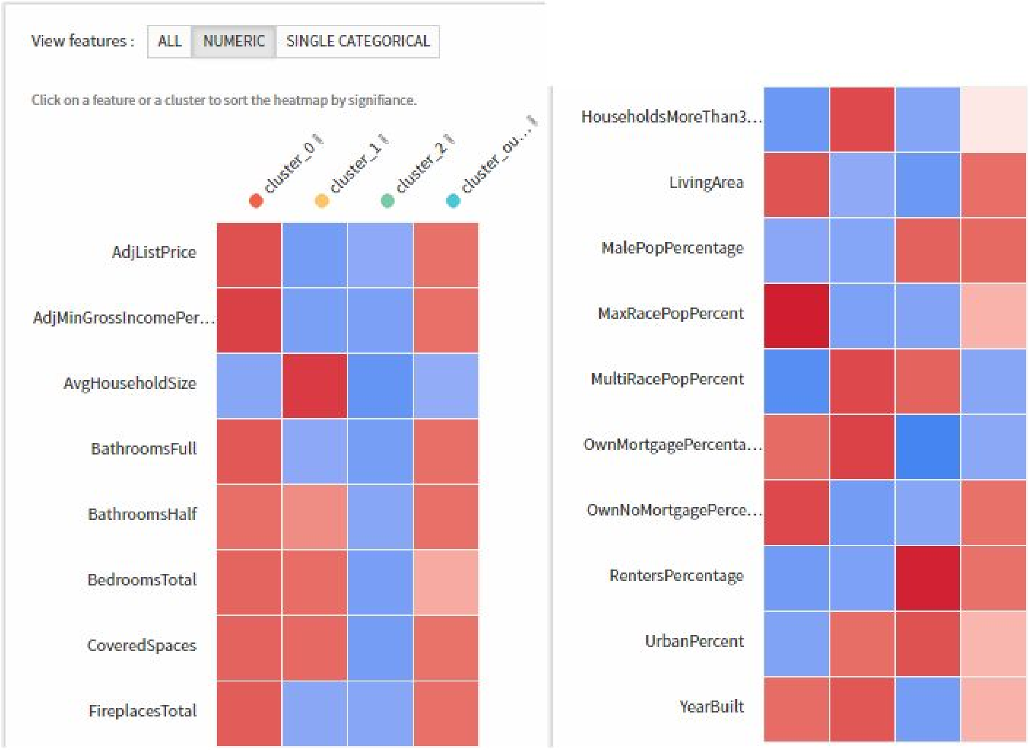
- 31.94% of the cluster has [u'Septic'] for **Sewer** (against 14.16 % globally)
- **MultiRacePopPercent** is in average 28.18% smaller : mean of 0.022 against 0.030 globally
- **MaxRacePopPercent** is in average 14.66% greater : mean of 0.88 against 0.77 globally

Variable Importance
The data we used to enrich the real estate data became the most important features used to differentiate the clusters. This was our hypothesis and it was exciting to see it visualized.



Cluster Heatmaps

This shows the numeric features and their levels (high or low) in each cluster. This allows us to see what makes each cluster unique. Red is high and blue is low.



Click on a feature or a cluster to sort the heatmap by signifiance.

Cluster_0 is made up of houses that are newer, larger and more expensive and are in less urban areas. Their owners have high income, either own with or without a mortgage and the households tend to be of one race.

Cluster_1 houses are smaller, more urban and less expensive with more occupants. They are newer homes and are often owned with a mortgage. Their owners make less money but are more diverse with multiple races and many generations living in the same house.

Cluster_2 are older, small houses in more urban settings with a high number of renters. They have fewer occupants and more women. The households are diverse.

App/UI

The intent of the UI is to allow the user to easily filter and change potential parts of the cluster that their current home is in. The app was created with a custom HTML/CSS/JavaScript package utilizing:

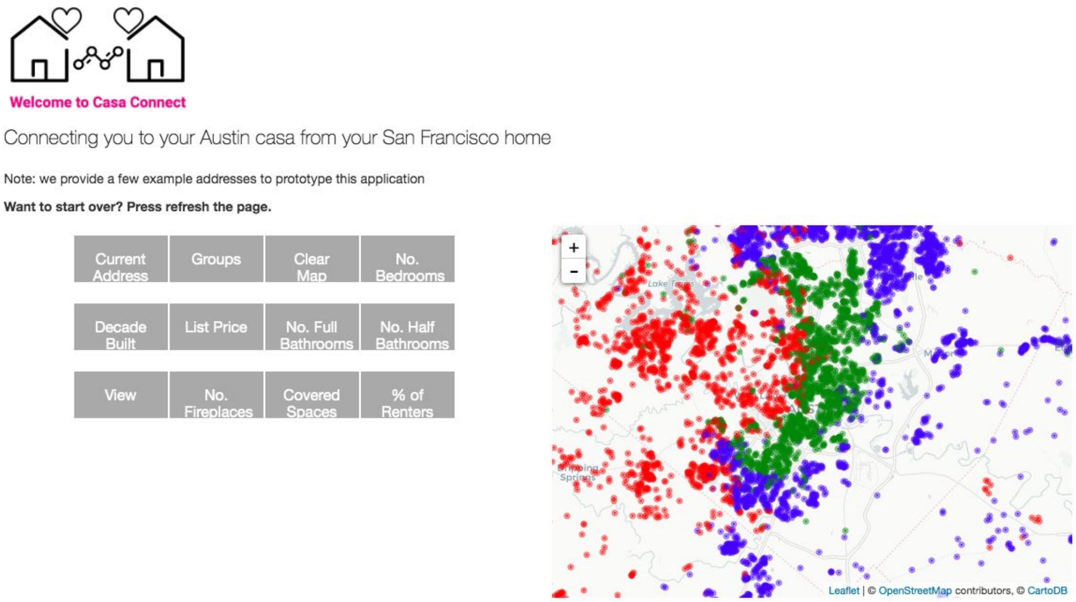
- JQuery, d3.js, Leaflet
- Bootstrap
- Dataiku API

Our files can be found in the code folder are as follows:

1. Casa.html
2. Casa.css
3. Casa.js

Note that the actual application is hosted at:
http://13.68.197.129/projects/CASA/webapps/u3nufJK_casa-connect/view

The UI



Welcome to Casa Connect

Connecting you to your Austin casa from your San Francisco home

Note: we provide a few example addresses to prototype this application

Want to start over? Press refresh the page.

List of Innovations

1. The novelty in our approach is using multiple data points about a person's current home/lifestyle beyond just home listing details, and then applying clustering models to find best matching homes in the target location.

    a. We allow the user to change their important features of the house matching process so they can find homes that fit their new or current lifestyle. This is innovative compared to the standard house search websites because they only allow a user to search for homes with explicit features such as bedrooms, bathrooms and lot size. Sites like Zillow and Trulia miss the nuances of what makes a neighborhood special, although this data can be mined from other data sources that are unrelated to typical real estate data.

    b. CasaConnect also makes the home buying process less stressful and more intuitive than current sites because it allows users to utilize their home as a baseline, and map the features they like from that home onto a home they have never seen.

2. Using Dataiku allows us to quickly and easily run multiple clustering models and select the best one based on silhouette. Because of this flexibility, we don't have to select just one model ahead of time and hope it works well for all scenarios. In theory, this would allow for the best clustering algorithm to be used for each individual search, maximizing the use of the data for each individual user.

Design of Experiments/Evaluation

Our testbed will include unit testing, integration testing and a modified user acceptance testing. **Every test below PASSED.**

Unit Testing (individual parts work)	
Test Description	*Success Criteria*
Does the page load?	Page loads fully (including map rendering).
Algorithm/Database accuracy	Returns appropriate results
Leaflet/ d3.js Map Zoom	Zooming works
Leaflet/ d3.js Map Markers	All markers appear when app is

	loaded
Addresses work as expected	Results appropriately appear
Filtering works as expected	Results appropriately filter
Clear Map	Map is cleared

Functional Testing (individual parts work together)	
Test Description	*Success Criteria*
UI/database integration	Correct dataset is returned
Maps integration	Map updates appropriately
Overall functionality	Does not error out, loads in appropriate amount of time

UAT - each group member has 3 people review the site as defined below	
Test Description	*Success Criteria*
Aesthetic	Visually pleasing
Usability	Results are meaningful and page is easy to use
User interest	Is the app useful

Distribution of Team Member’s Activities

Team Member	Activities	Hours Spent
Taylor Gift	● Built the initial Django web app ● Identified and started implementation of necessary spatial and geographic libraries ● Created shapefiles for coordinate representation ● Built the initial coordinate database in PostgreSQL ● Did technical needfinding on building the app ● Explored API integration within the app ● Helped prepare all reports	30 hours
Katrina Green	● Helped examine datasets San Diego/San Fran. ● Researched Django for our front end development. ● Cleaned up our final datasets to use for clustering. ● Helped with final testing on UI. ● Created our final poster for presentation. ● Helped prepare all reports.	30 hours
Shelly Kunkle	● Obtained license for Dataiku; provisioned Dataiku in Azure ● Found and obtained access to real estate and Census Bureau APIs; wrote Python code to pull in data from APIs ● Built the real estate dataset; pulled census data, IRS data and cost of living data; joined to real estate data ● Experimented with and built various clustering algorithms; chose k-means ● Helped prepare all reports	50 hours

Melanie Laffin	● Cleaned San Fran data ● Obtained appropriate front end template and modified for CasaConnect; ● Utilized Shelly's code to pull more data from API's; ● Designed and implemented front end web app on Dataiku's web app function ● Connected data from result of clustering algorithm to front end ● Helped prepare all reports	45 hours

Conclusion

The original idea behind Casa Connect is that we could provide a better way to find the perfect home that what the current state-of-the-art tools provide. Data sources beyond real estate listing details provide a much better basis for finding a home that truly matches what a buyer is looking for. Casa Connect assumes that a person's current address is the best starting point in finding best-matching houses in a new city.

We successfully validated our data and discovered the most important features for differentiating user preferences from one house to another. Out of the top ten most important variables, only one--Year Built--came from the real estate dataset. The rest are from the census metrics, number one being MaxRacePopPercent. (This shows the level of diversity in the households in the area.)

We found that k-means clustering method won out in terms of silhouette and time to run vs. Interactive, Agglomerative and DBSCAN methods.

Our UI showed that starting with an existing address matched to a cluster in the new city provided a good user experience. The user then filters down to their best recommendation based on the other parameters.

We are happy that if Casa Connect were industrialized would be an improvement over the current state-of-the-art tools (Zillow, e.g.).

Appendix A: Survey References

TG1: A Collaborative Filtering Recommendation Algorithm Based on User Clustering and Item Clustering

TG2: Statistical Analysis of Residential Housing Prices in an up and down Real Estate Market: a General Framework and Study of Cobb County, GA

TG3:Urban Morphology and Housing Market - Chapter 2

KG1: Measuring Neighborhood Walkable Environments: A Comparison of Three Approaches

KG2: The Elements of Statistical Learning Data Mining, Inference, and Prediction

KG3: Perception or Reality, what matters most when it comes to crime in your neighborhood?

SK1: Research on social data by means of cluster analysis

SK2: Unsupervised Learning: Clustering

SK3: Highlights From the Profile of Home Buyers and Sellers

ML1: Market Distortions When Agents Are Better Informed: The Value of Information in Real Estate Transactions

ML2: Exploring semantic content to user profiling for user cluster-based collaborative point-of-interest recommender system

ML3: Combining Structured and Unstructured Information Sources for a Study of Data Quality: A Case Study of Zillow.Com