

Project Name: Zillow Clustering Project

Project Type: Required solo project

Project Goals:

Use clustering ML algorithm to identify clusters within the data

Construct a ML regression model that improves logerror prediction

Deliverables: Main Notebook for Pipeline process, readme.md, acquire.py, prepare.py, wrangle_zillow.py
explore.py, model.py

5 min notebook walkthrough

Stage	Tools	Brief Description of Process	Challenge Resolution
Plan	<ul style="list-style-type: none">Visual Studio	<ul style="list-style-type: none">In Visual Studio create a new readme.md file to outline project plan, create GitHub repoImport key elements and deliverables from curriculum requirements	<ul style="list-style-type: none">No unusual challenges in this section
Acquire	<ul style="list-style-type: none">Visual StudioMySQL.py scriptSQL queryQuery function	<ul style="list-style-type: none">In Visual Studio create a new acquire.py file to obtain data from Codeup Zillow databaseDetermine “single housing unit” definition from domain research and filter dataQuery required to have only transaction dates from 2017 and only most recent transaction date for property	<ul style="list-style-type: none">mySQL query required multiple subqueries to resolve, ended up finding out you can join on 2 columns which resolved problem
Prepare	<ul style="list-style-type: none">Visual Studio.py scriptJupyter Notebook	<ul style="list-style-type: none">Needed to handle/clean lots of null values and make decisions on outliersAdded multiple additional functions to my prepare.py file for future useCreated functions for identifying and	<ul style="list-style-type: none">Original removal of outliers with IQR reduced dataset to far, changed method for calculating outliers based on domain research

	<ul style="list-style-type: none"> • Matplotlib 	<ul style="list-style-type: none"> removing outliers with IQR • Created functions to determine % of missing data in row or columns and remove based on exceeding set threshold • 	
Explore	<ul style="list-style-type: none"> • Jupyter Notebook • Seaborn • Scipy.stats • Matplotlib • KMeans 	<ul style="list-style-type: none"> • Successfully used loop functions to auto generate scatterplots for interaction of independent variables with target • Also used loops effectively for statistical testing to produce a dataframe of summarized results from stats tests • Used clustering unsupervised ML algorithm KMeans to engineer features for use in modeling • Added functions to summarize.py and explore.py for use in future projects 	<ul style="list-style-type: none"> • Getting loops to work is one of biggest achievements for this project for me • Successfully move a lot of the code from the notebook to functions in the .py files
Model	<ul style="list-style-type: none"> • Jupyter Notebook • Sklearn • Multiple ML models tested 	<ul style="list-style-type: none"> • Created model.py file with multiple modeling functions for use in future projects • Used RFE (recursive feature engineering) to evaluate best features for modeling • Created baseline using mean of training data for predictions • Used same regression ML algorithms from previous project to test models • Created many iterations of models varying features, quantity of features, and adjusting hyperparameters • Create loop function to iterate through multiple models and produce summary dataframe of results 	<ul style="list-style-type: none"> • Feel very accomplished that I could get the loop and dataframe summary working • RFE features did not have high correlations with results from statistical analysis

		<ul style="list-style-type: none"> • Made multiple iterations of models using LinearRegression, LassoLars, Polynomial Features, and TweedieRegression 	
Evaluate	<ul style="list-style-type: none"> • Jupyter Notebook • Sklearn 	<ul style="list-style-type: none"> • Used RMSE (root mean squared error) to determine if model produced less RMSE than baseline • Many models performed well on training data, but all models performed worse on unseen data • Best result was worse than baseline by 3.6% • 	<ul style="list-style-type: none"> • Unable to return a model that did better than baseline on unseen data, all models appear to be overfit
Model Explanation	How does your algorithm work?	<ul style="list-style-type: none"> • Linear regression worked best • LinearRegression fits a linear model with coefficients $w = (w_1, \dots, w_p)$ to minimize the residual sum of squares between the observed targets in the dataset, and the targets predicted by the linear approximation • Also known as Ordinary Least Squares 	<ul style="list-style-type: none"> • No unusual challenges in this section
Delivery	<ul style="list-style-type: none"> • Jupyter Notebook 	<ul style="list-style-type: none"> • Used slides feature I hadn't known was available in Jupyter Notebook until this project • Created slides from notebook cells for presentation this helped keep me on track timewise 	<ul style="list-style-type: none"> • HTML syntax error created issues in GitHub preview