

# Predicting Real Estate Sale Prices

...

A Lecture on Linear Regression  
James Pecore, Columbia University Data Scientist

# Overview

- I. Problem Statement
- II. Executive Summary
- III. Data Cleaning: Null Values
- IV. Data Cleaning: One-Hot-Encoding/Mapping
- V. Exploratory Data Analysis: Outlier Management
- VI. Feature Engineering
- VII. Feature Selection
- VIII. Ridge and LASSO Regularization
- IX. RMSE and Evaluating Predictions
- X. Recommendations for Real Estate Developers
- XI. Data Source and Python 3 Libraries

# I. Problem Statement

For housing in AMES, Iowa:

- How can a house's information be used by Real Estate developers and clients to predict or increase a property's value?
- Moreover, how can a data scientist use a model to process all of this data?

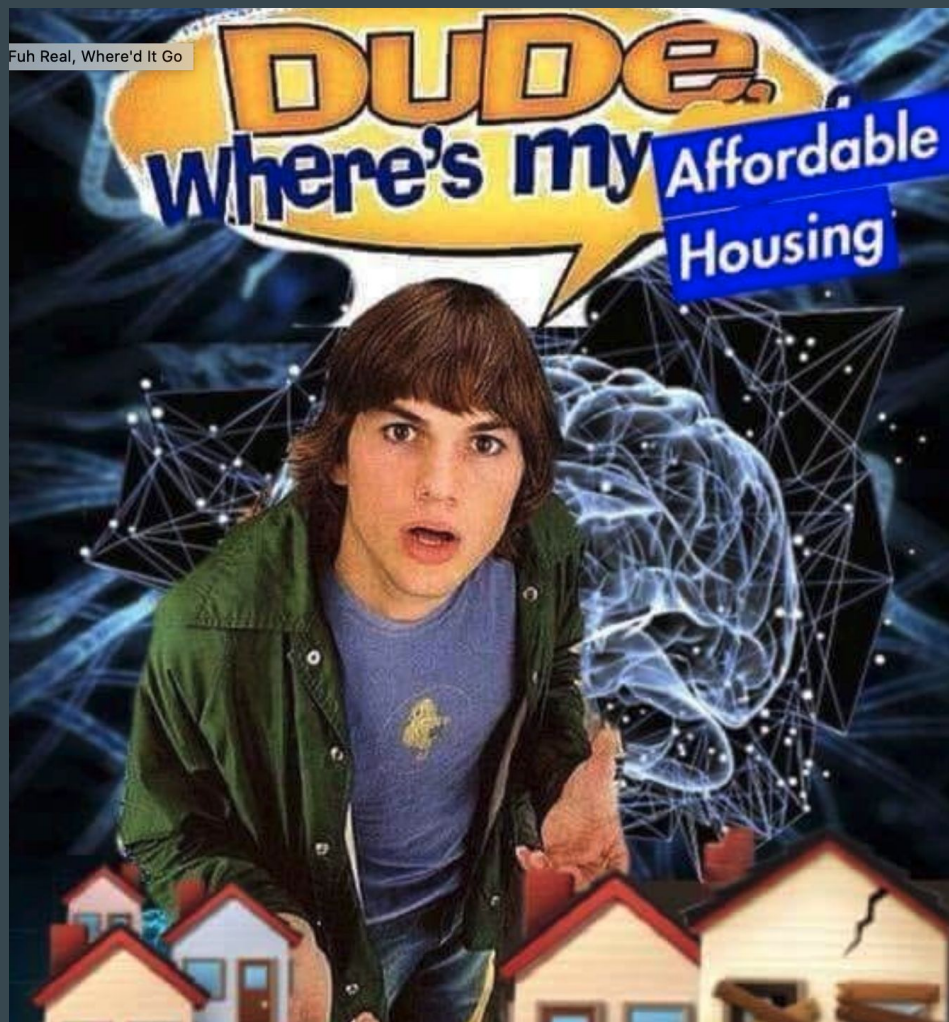
## II. Executive Summary

By using the form of Machine Learning called **Linear Regression**, this project processes through a database of AMES Housing training data.

Then, linear regression predicts a house's **sale price** and **value** using the trends and properties of this training data.

Additionally, **regression techniques** (such as **feature engineering** and **selection**) enhance the quality of our predictions.

Fuh Real, Where'd It Go



# III. Data Cleaning: Null Values

Replace **Null Values (NaNs)** with appropriate replacements using `.replace`

1. **Categorical variables** need ('NA') replacements

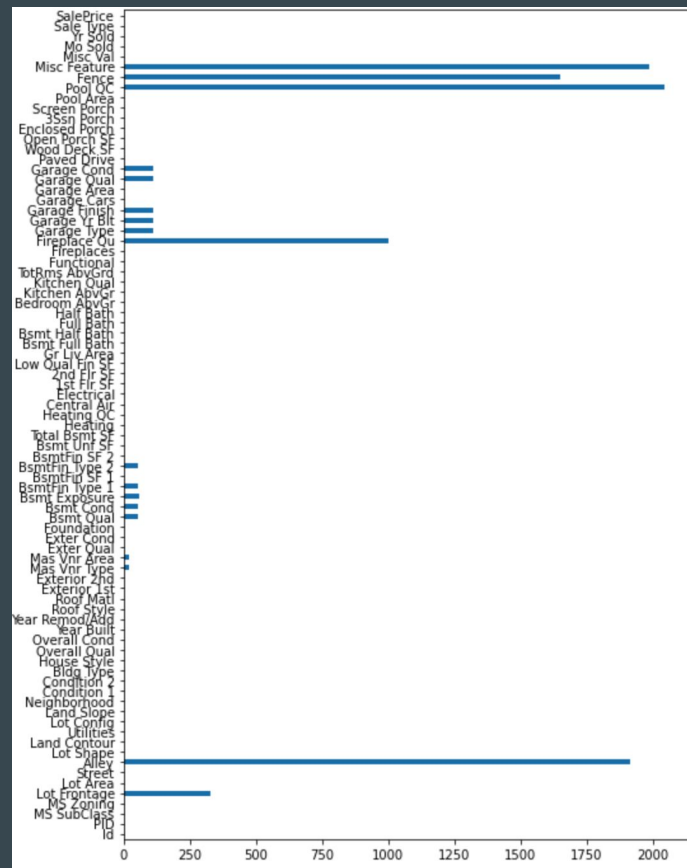
a. These will later be one-hot-encoded (dummified)

2. **Continuous variables** need (Float) replacements

3. **Discrete variables** need (Int) replacements

4. **Ordinal variables** need ('NA') replacements

a. These will later be mapped with hierarchical values



## IV. Data Cleaning: One-Hot-Encoding/Mapping

Categorical variables (like 'Neighborhood') possess a lot of information but need to be made quantitative.

- One-hot-encoding gives each Neighborhood a column of binary (0/1) values for each neighborhood:
  - 0 value properties are not in that Neighborhood.
  - 1 properties are in that Neighborhood.
- We can interpret this data numerically now.



## IV. Data Cleaning: One-Hot-Encoding/Mapping

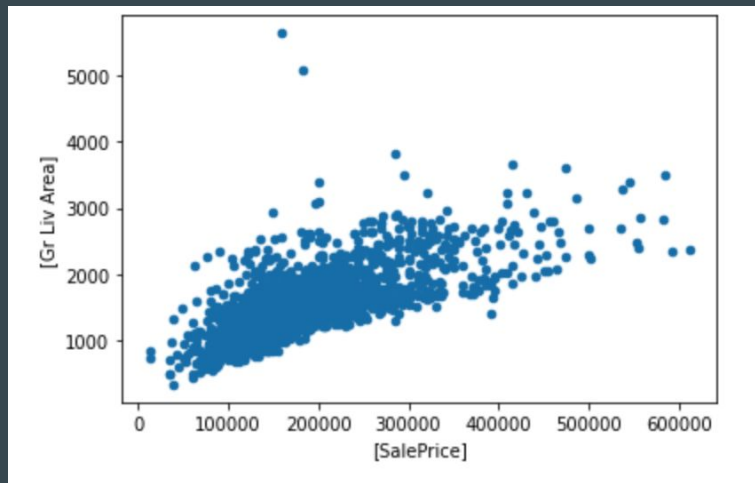
Ordinal variables (like 'Garage Quality') possess a lot of information but need to be made quantitative. Additionally, these quantities need to have a hierarchy (e.g. '5 is more valuable than 2').



- {'Lot Shape' : {'IR3' : 1, 'IR2' : 2, 'IR1' : 3, 'Reg' : 4},
- 'Utilities': {'ELO' : 0, 'NoSeWa' : 1, 'NoSewr' : 2, 'AllPub' : 3},
- Etc. }
- We can interpret this data numerically now.

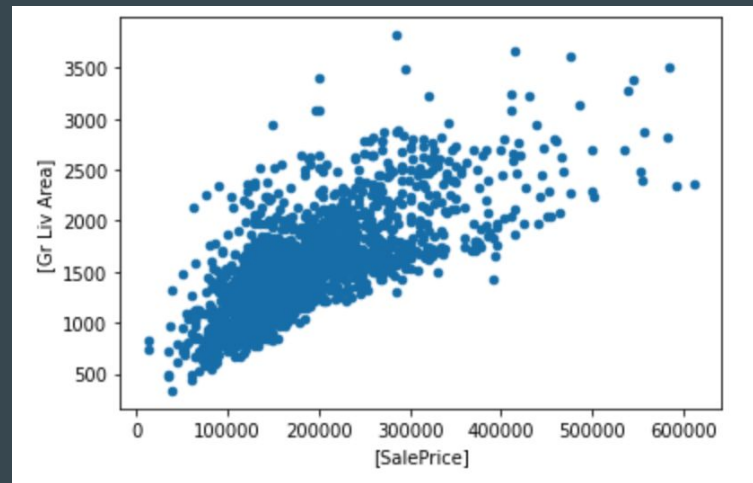


# V: Exploratory Data Analysis: Outlier Management



Gross Living Area vs Sale Price

**BEFORE**



Gross Living Area vs Sale Price

**AFTER**

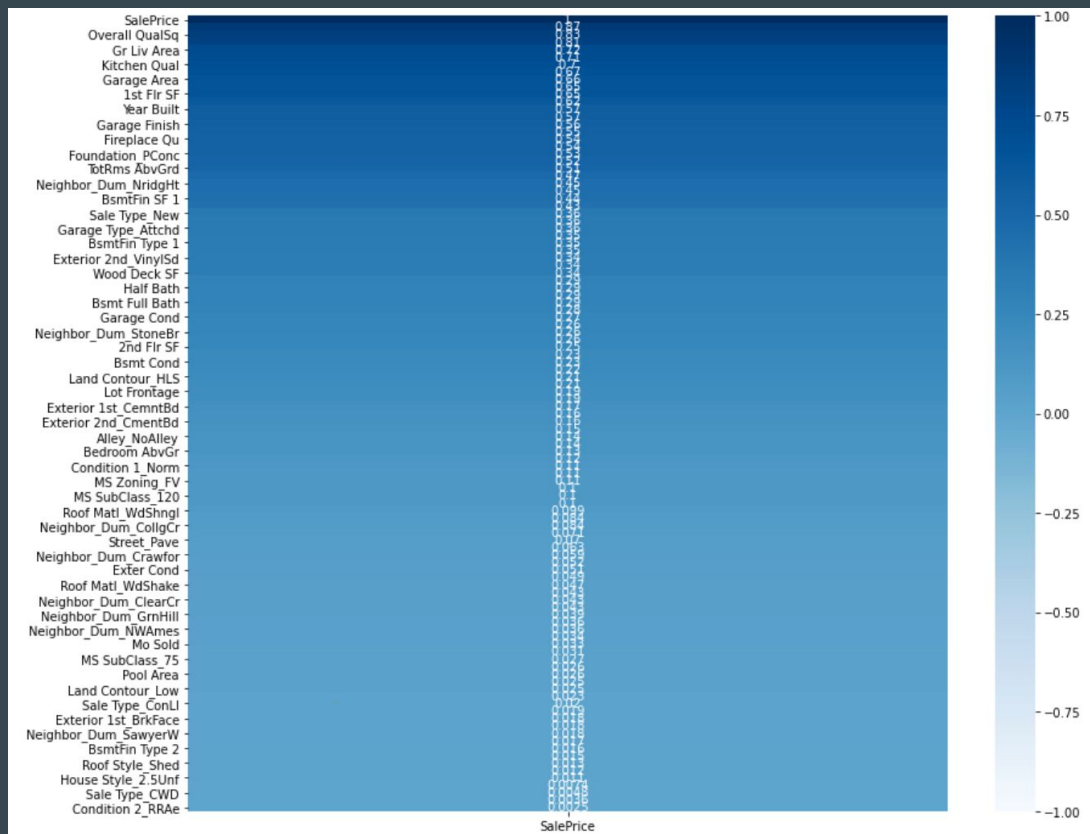
## VI. Feature Engineering

- # This code conglomerates the correlated data 'Full Bath' and 'Half Bath'
- $\text{train\_df}[\text{'Bathroom Total'}] = \text{train\_df}[\text{'Full Bath'}] + (0.5 * \text{train\_df}[\text{'Half Bath'}])$

Feature engineering lets us **express correlations** between features and **highlight** them in the model, increasing the accuracy of the model's predictions for a house's Sale Price.



# VII. Feature Selection



## VIII. Ridge and LASSO Regularization

- First, I scaled the dataset with **StandardScaler**.
- Then, both Ridge and LASSO regularized the data set.
- Regularization:
  - reduces magnitude of coefficients
  - optimizes datasets
  - often increases  $R^2$  Values
  - improves accuracy

## IX. RMSE and Evaluating Predictions

Root Mean Squared Error:

‘27,979’

- Meaning: Each prediction for Sale Price may be at most \$27,979 from the actual Sale Price of each property.
- $R^2 = 90\%$

## X. Recommendations for Real Estate Developers

One can use linear regression to greatly reduce errors and improve the performance of predicting sale price based on a variety of highly correlated factors.

Although, it seems that knowing which factors influence a house's price (such as Kitchen Quality and Bathroom Space) can often be more valuable than the predictive model itself.

As a result, even the basic method of Linear Regression can be incredibly useful for a firm's Real Estate pricing decisions.

# X. Recommendations for Real Estate Developers

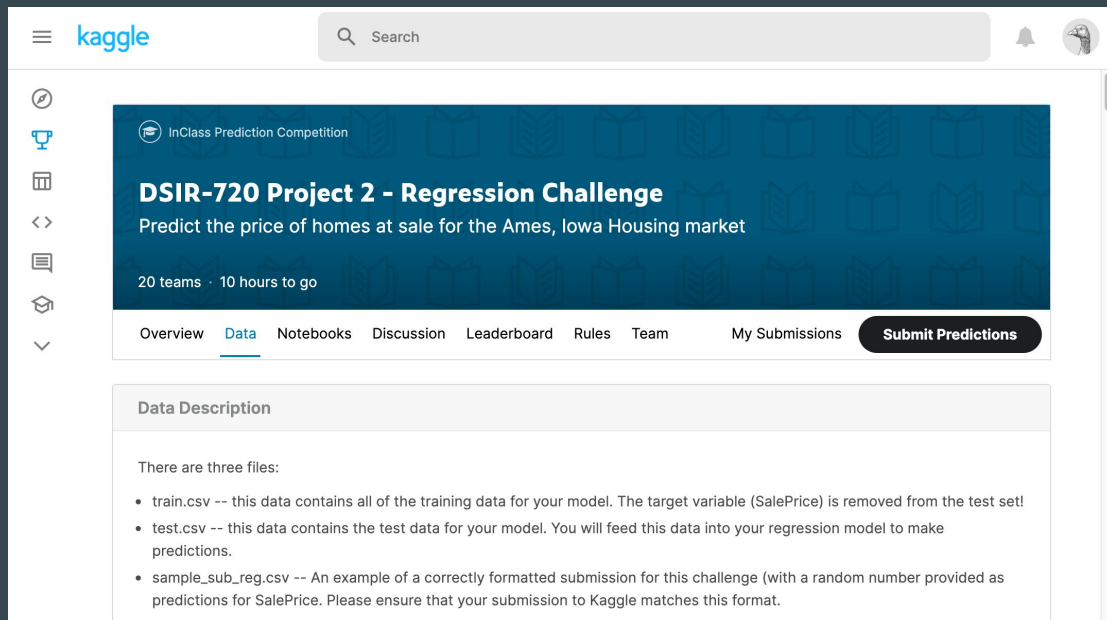
## Important Factors (Top 7):

1. Overall Quality
2. Gross Living Area
3. Kitchen Quality
4. Garage Area
5. 1ST Floor Square Footage
6. Year Built
7. Finished Garage

Questions?



# XI. Data Source and Python Libraries



The screenshot shows the Kaggle website interface for the 'DSIR-720 Project 2 - Regression Challenge'. The header includes the Kaggle logo, a search bar, and user icons. The left sidebar contains navigation icons. The main content area features a blue banner for the 'InClass Prediction Competition' with the title 'DSIR-720 Project 2 - Regression Challenge' and the description 'Predict the price of homes at sale for the Ames, Iowa Housing market'. It also indicates '20 teams · 10 hours to go'. Below the banner is a navigation bar with tabs: Overview, Data (selected), Notebooks, Discussion, Leaderboard, Rules, Team, My Submissions, and a 'Submit Predictions' button. The 'Data Description' section states: 'There are three files:' followed by a bulleted list: 'train.csv -- this data contains all of the training data for your model. The target variable (SalePrice) is removed from the test set!', 'test.csv -- this data contains the test data for your model. You will feed this data into your regression model to make predictions.', and 'sample\_sub\_reg.csv -- An example of a correctly formatted submission for this challenge (with a random number provided as predictions for SalePrice. Please ensure that your submission to Kaggle matches this format.'

<https://www.kaggle.com/c/dsir-720-project-2-regression-challenge/data>  
<http://jse.amstat.org/v19n3/decock/DataDocumentation.txt>

# XI. Data Source and Python Libraries

- import **pandas** as **pd**
- import **numpy** as **np**
- import **matplotlib.pyplot** as **plt**
- import **seaborn** as **sns**
  
- from **sklearn.linear\_model** import **LinearRegression**
- from **sklearn** import **metrics**
- from **sklearn.model\_selection** import **train\_test\_split**, **cross\_val\_score**
- from **sklearn.preprocessing** import **PolynomialFeatures**, **StandardScaler**
- from **sklearn.linear\_model** import **Ridge**, **RidgeCV**
- from **sklearn.linear\_model** import **Lasso**, **LassoCV**