

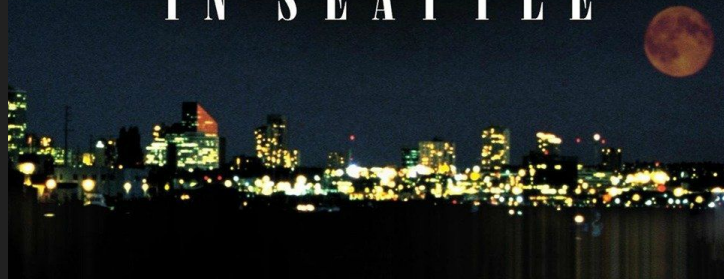
Analyzing the KC House Sales Dataset to Model Sale Price

AKA



BOLSON

IN SEATTLE



Who is Bolson?

- CEO Bolson Construction
- Specializes in:
 - Building Homes
 - Renovating Homes
 - Tearing Down Houses
- Not from around here!
- Needs your business advice!



What does Bolson want to know?



- How much does a house sell for? What drives that price?
- Where are the newest housing developments? Where's the market?
- Is price a function of n bedrooms? N bathrooms? N floors?

To answer these questions Bolson
needs:

Data analysis! A working price model!



Let's get started:

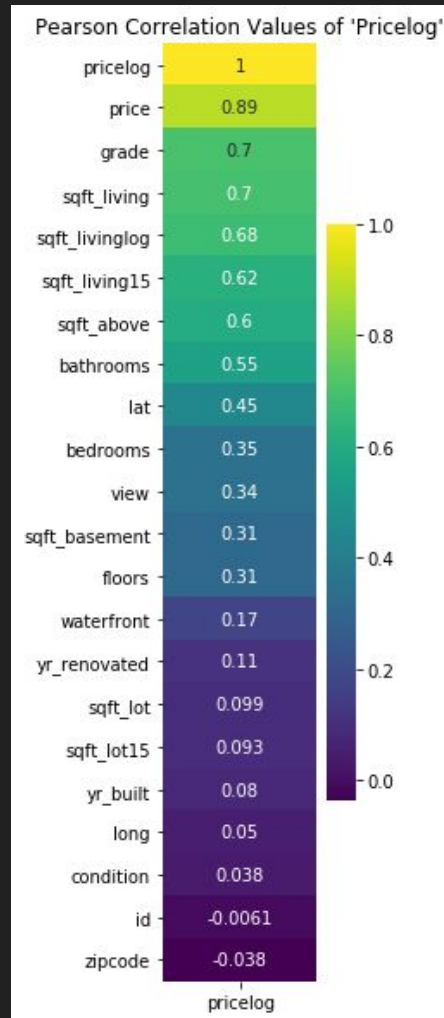
Where are the newest developments?

- Plot all sales on a scatterplot using 'lat' and 'long' data
- Filter all sales for just those that occurred in most recent two years (2014, 2015 for this dataset)
- Plot 'most recent sales' over top all sales, look for trends
- Find mean price:
 - \$687K
- Find median price:
 - \$599K



What drives the price of houses?

- Pricelog is normalized distribution of Price
- Higher correlation implies better predictor
- Looking at Grade, SQFT Living, Bathrooms, LAT, etc
- How do we clarify?
 - Run a test model
 - R^2 w/ only one variable:
 - Grade
- 'Grade' can explain almost 50% of the data on its own! Definitely the biggest driver!



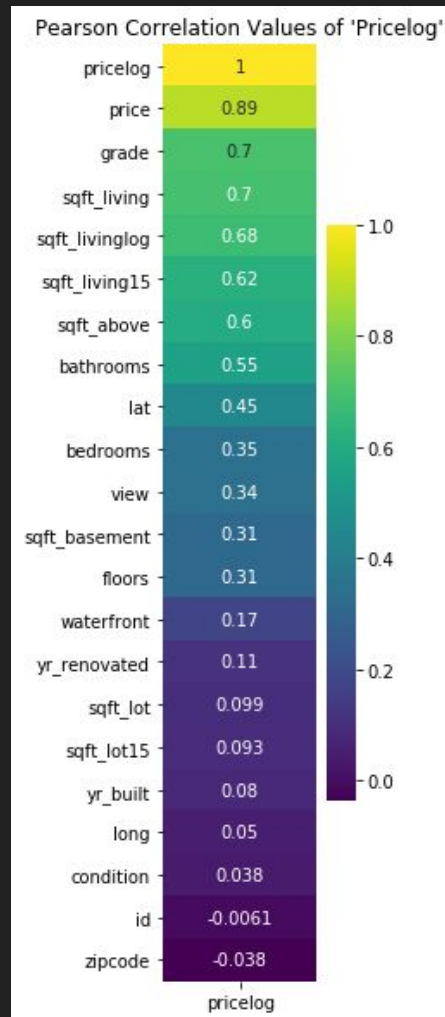
Dep. Variable:	pricelog	R-squared:	0.495
Model:	OLS	Adj. R-squared:	0.495
Method:	Least Squares	F-statistic:	2.066e+04
Date:	Fri, 21 Jun 2019	Prob (F-statistic):	0.00
Time:	18:22:48	Log-Likelihood:	-9172.9
No. Observations:	21054	AIC:	1.835e+04
Df Residuals:	21052	BIC:	1.837e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
Intercept	10.6321	0.017	624.993	0.000	10.599	10.665
grade	0.3156	0.002	143.752	0.000	0.311	0.320

Omnibus:	133.048	Durbin-Watson:	1.965
Prob(Omnibus):	0.000	Jarque-Bera (JB):	136.615
Skew:	0.186	Prob(JB):	2.16e-30
Kurtosis:	3.132	Cond. No.	52.0

What about beds, baths, and extra floors?

- High correlation with n bathrooms helps, but still explains much less of Price - even with three variables!



Dep. Variable:	pricelog	R-squared:	0.311			
Model:	OLS	Adj. R-squared:	0.311			
Method:	Least Squares	F-statistic:	3171.			
Date:	Fri, 21 Jun 2019	Prob (F-statistic):	0.00			
Time:	18:22:48	Log-Likelihood:	-12447.			
No. Observations:	21054	AIC:	2.490e+04			
Df Residuals:	21050	BIC:	2.493e+04			
Df Model:	3					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
Intercept	12.1041	0.013	911.436	0.000	12.078	12.130
bathrooms	0.3268	0.005	61.880	0.000	0.316	0.337
bedrooms	0.0523	0.004	13.230	0.000	0.045	0.060
floors	0.0515	0.007	7.918	0.000	0.039	0.064
Omnibus:	192.663	Durbin-Watson:	1.962			
Prob(Omnibus):	0.000	Jarque-Bera (JB):	197.773			
Skew:	0.236	Prob(JB):	1.13e-43			
Kurtosis:	3.054	Cond. No.	20.7			

Oh no! There's no easy answer!



So what do we do?

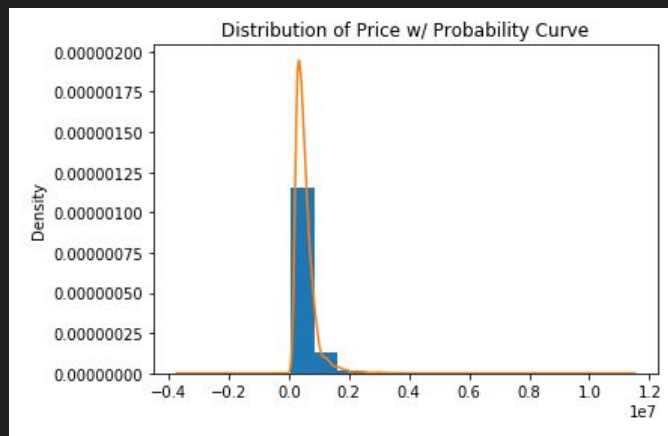
We Build a Model That Examines
Multiple Features!

To get a working model we need:

1. Linear relationships
 - a. The Pearson correlation tells us this, so we've got some options
 - b. We can't include things that strongly correlate to each other!
2. When the model is wrong, it needs to be reliably wrong
 - a. i.e. it's wrongness must be predictable
 - b. We need to 'normalize' some of our data
3. The model has to be fairly reliable
 - a. This is our friendly R^2 variable

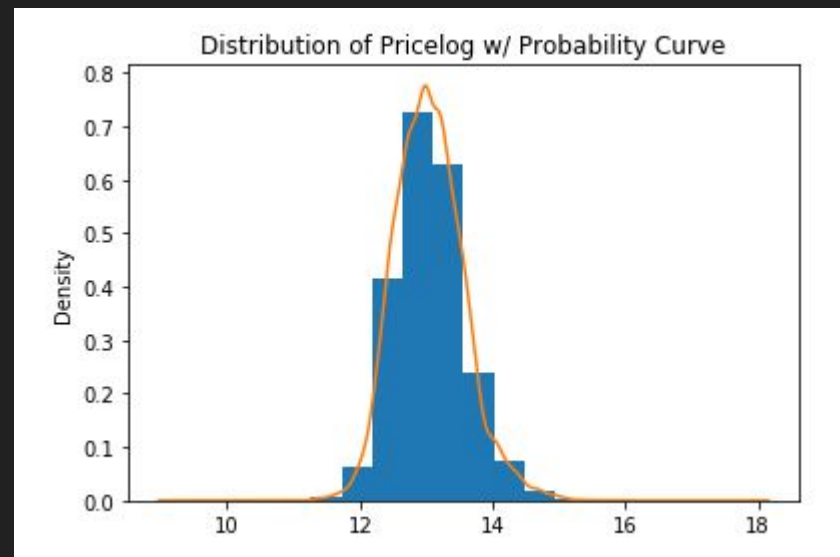
Predictably Unpredictable

- Linear models work best when they are 'normalized' aka 'proportionally distributed'
- Pricelog, as seen earlier, is the normalized version of Price
- Transforming it like this will help make sure that the model's mistakes are ALSO proportionally distributed!



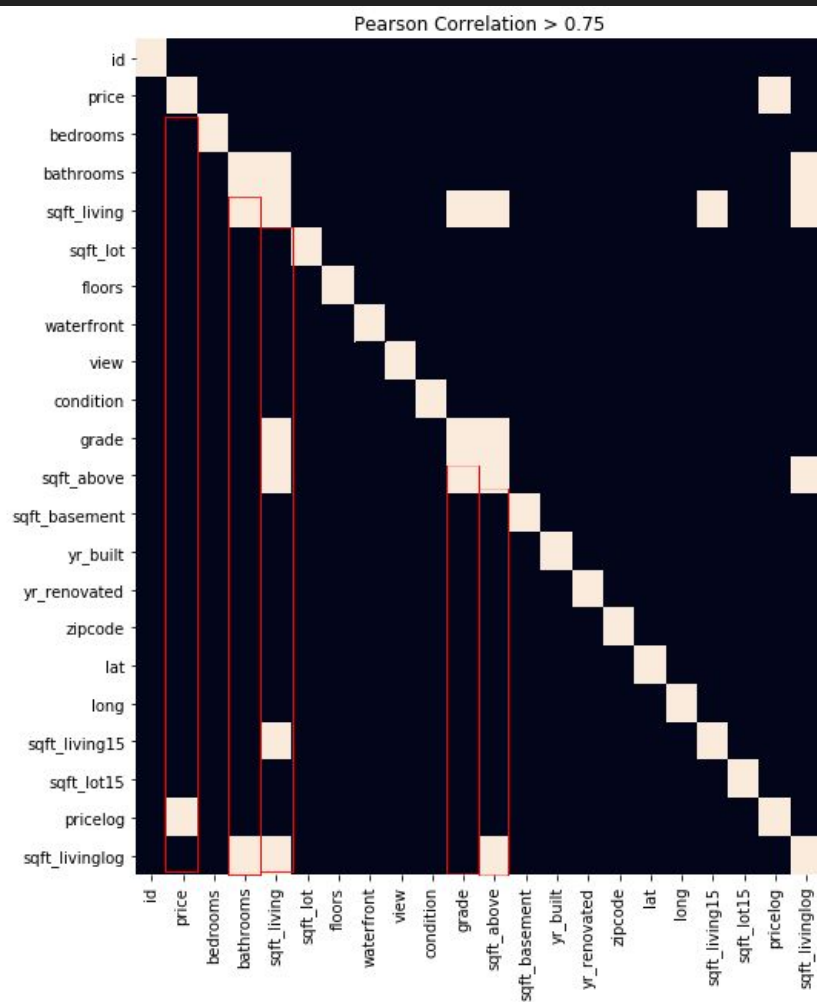
Before

After



Don't Double-Dip

- If we add in features that are closely related to each other, they can disrupt the Pricelog prediction!
- Think of it like paying off one credit card with another: things are moving, but you aren't changing your total amount of debt
- Take out things that talk to each other:
 - e.g. Grade talks to sqft_living and sqft_above

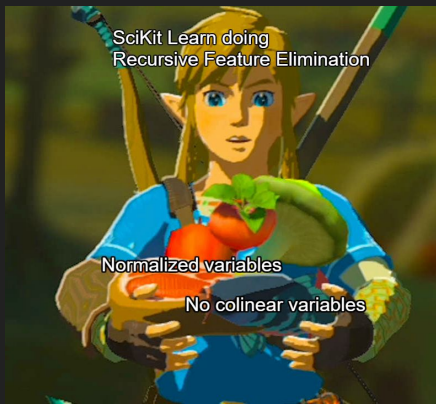


To get a working model we need:

1. Linear relationships
 - a. ✓ The Pearson correlation tells us this, so we've got some options
 - b. ✓ We can't include things that strongly correlate to each other!
2. When the model is wrong, it needs to be reliably wrong
 - a. ✓ i.e. it's wrongness must be predictable
 - b. ✓ We need to 'normalize' some of our data
3. The model has to be fairly reliable
 - a. This is our friendly R^2 variable

Building a Model With What We Have

- After knocking out variables that correlate to each other, we're down to about 15 total
- We run combinations of the 15 into the model, and keep the variables that have the largest coeff (ie, really big impact), discard the rest
- Play with some trial and error, adding some variables back in
- Get a nice R^2



Dep. Variable:	pricelog	R-squared:	0.723
Model:	OLS	Adj. R-squared:	0.723
Method:	Least Squares	F-statistic:	9175.
Date:	Fri, 21 Jun 2019	Prob (F-statistic):	0.00

Time:	18:22:53	coef	std err	t	P> t	[0.025	0.975]
No. Observations:	21054	Intercept	-42.6515	0.733	-58.172	0.000	-44.089 -41.214
Df Residuals:	21047	grade	0.2666	0.002	118.485	0.000	0.262 0.271
Df Model:	6	lat	1.2965	0.014	90.746	0.000	1.268 1.324
Covariance Type:	nonrobust	waterfront	0.6487	0.023	27.993	0.000	0.603 0.694
		condition	0.0617	0.003	19.478	0.000	0.055 0.068
		bathrooms	0.1874	0.004	53.282	0.000	0.181 0.194
		yr_built	-0.0044	8.38e-05	-52.113	0.000	-0.005 -0.004

Spicy Multivariable Fry



* 03:30

It could be the answer!!

To get a working model we need:

1. Linear relationships

- a. The Pearson correlation tells us this, so we've got some options
- b. We can't include things that strongly correlate to each other!

2. When the model is wrong, it needs to be reliably wrong

- a. i.e. it's wrongness must be predictable
- b. We need to 'normalize' some of our data

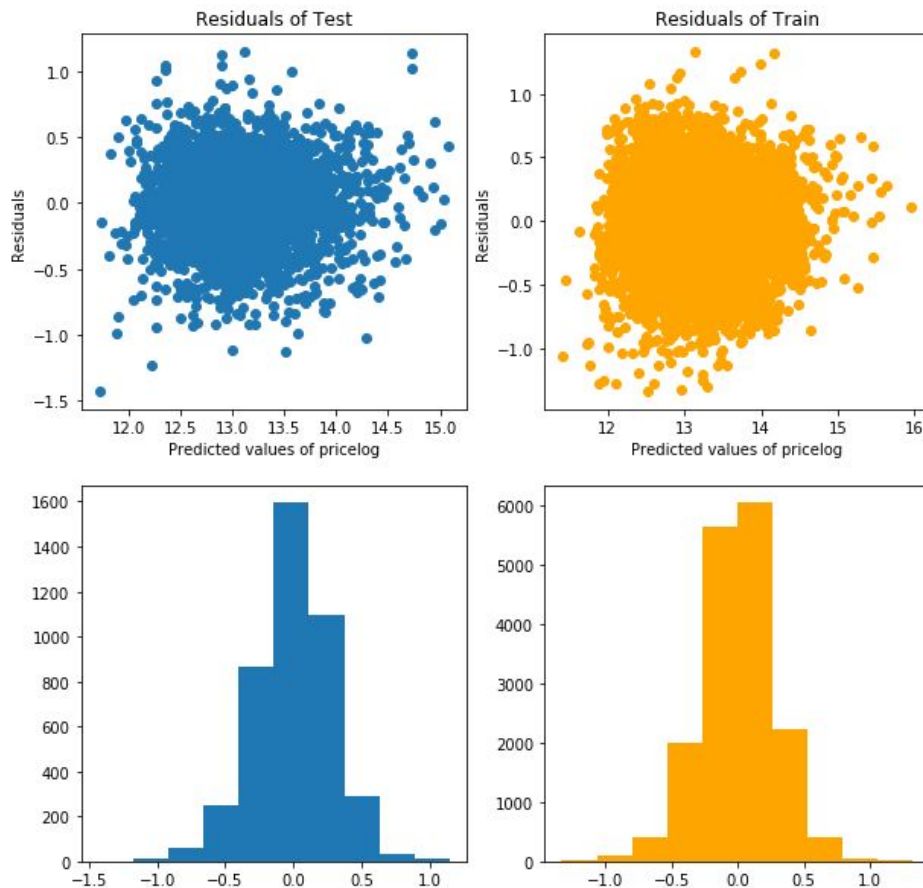
3. The model has to be fairly reliable

- a. This is our friendly R^2 variable

Wrong in the Right Way

- We randomly split the data into two sets:
 - A larger “Training” set
 - A smaller “Testing” set
- We build a model w/ the same variables on the Training set
 - New coeffs, New R^2
- Test that model on the “Testing” set and the “Training” set
- Then, compare ‘wrongness’
- If MUCH more wrong on one set than the other? Model doesn’t work!
- If errors are CONSISTENT and evenly distributed? Model does work!

Distribution of Model Residuals



To get a working model we need:

1. Linear relationships

- a. The Pearson correlation tells us this, so we've got some options
- b. We can't include things that strongly correlate to each other!

2. When the model is wrong, it needs to be reliably wrong

- a. i.e. it's wrongness must be predictable
- b. We need to 'normalize' some of our data

3. The model has to be fairly reliable

- a. This is our friendly R^2 variable

We did it! We can help Bolson predict housing prices!



What our model looks like:

$$\ln(\text{Price}) = \alpha(\text{Grade}) + \beta(\text{Latitude}) + \gamma(\text{Waterfront}) + \delta(\text{Condition}) + \epsilon(\text{Bathrooms}) + \zeta(\text{Year Built}) + \text{Intercept}$$



$$\ln(\text{Price}) = \frac{133}{500} \text{Grade} + \frac{162}{125} \text{Latitude} + \frac{81}{125} \text{Waterfront} + \frac{77}{1250} \text{Condition} + \frac{187}{1000} \text{Bathrooms} - \frac{1}{250} \text{Year Built} - 42.651$$