Linear Regression

Geena Kim

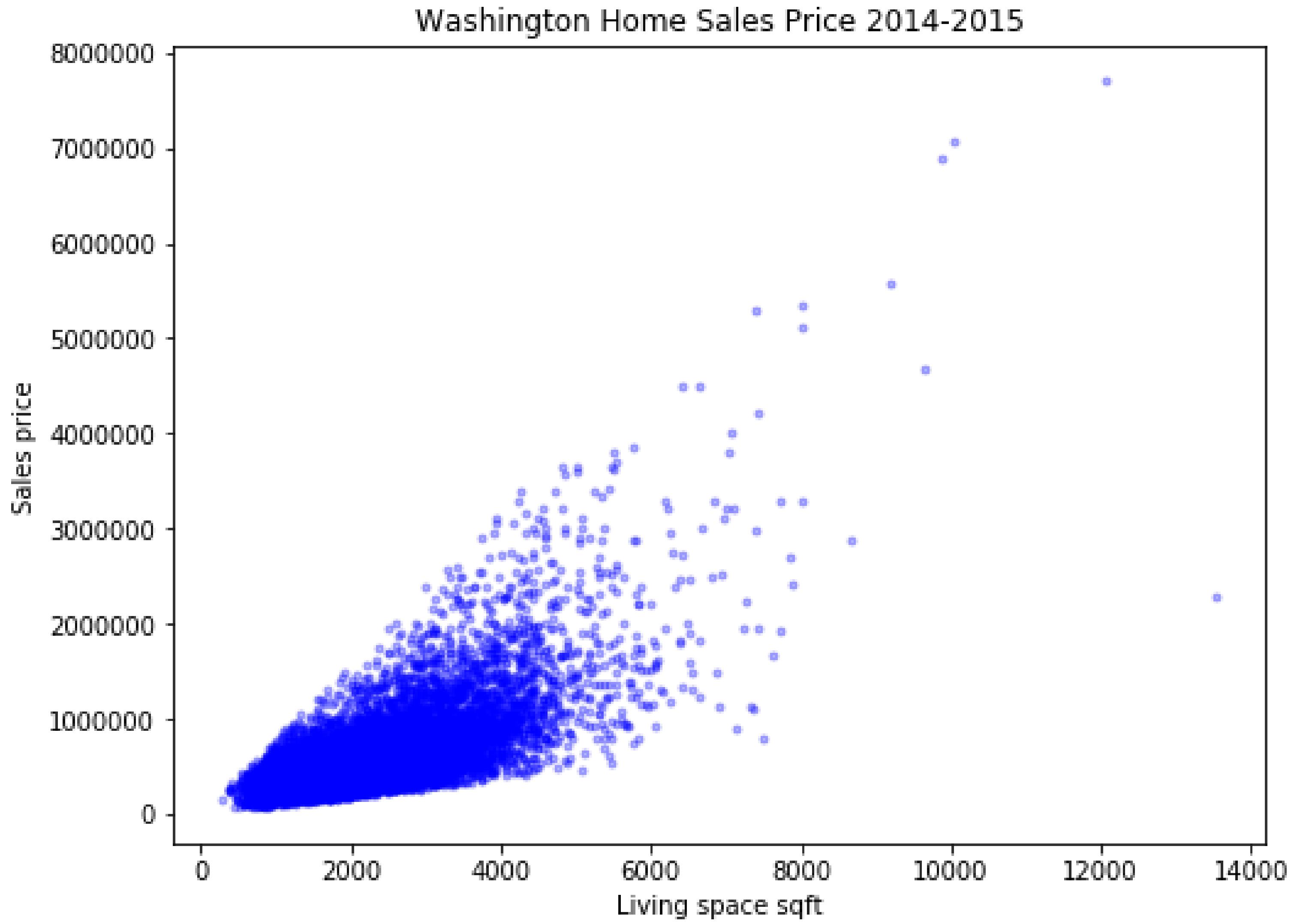
University of Colorado Boulder

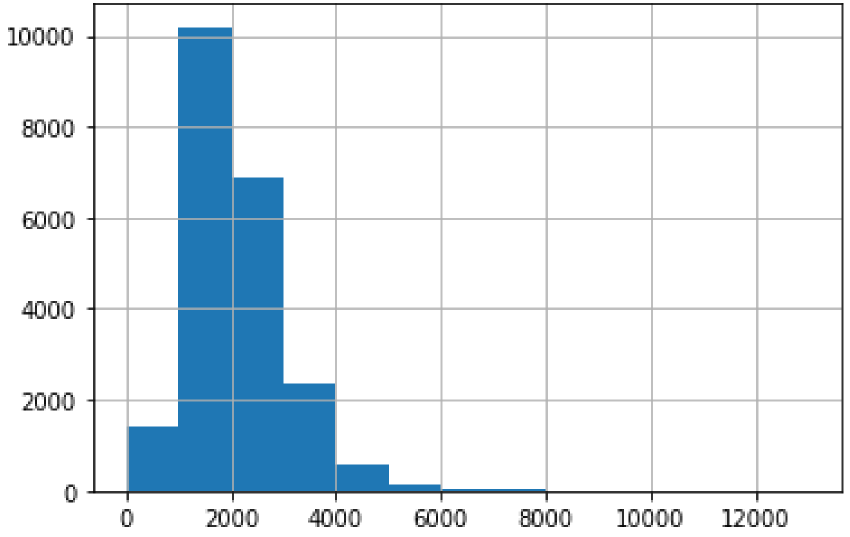
Example

	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors
	221900	3	1.00	1180	5650	1.0
	538000	3	2.25	2570	7242	2.0
	180000	2	1.00	770	10000	1.0
	604000	4	3.00	1960	5000	1.0
	510000	3	2.00	1680	8080	1.0

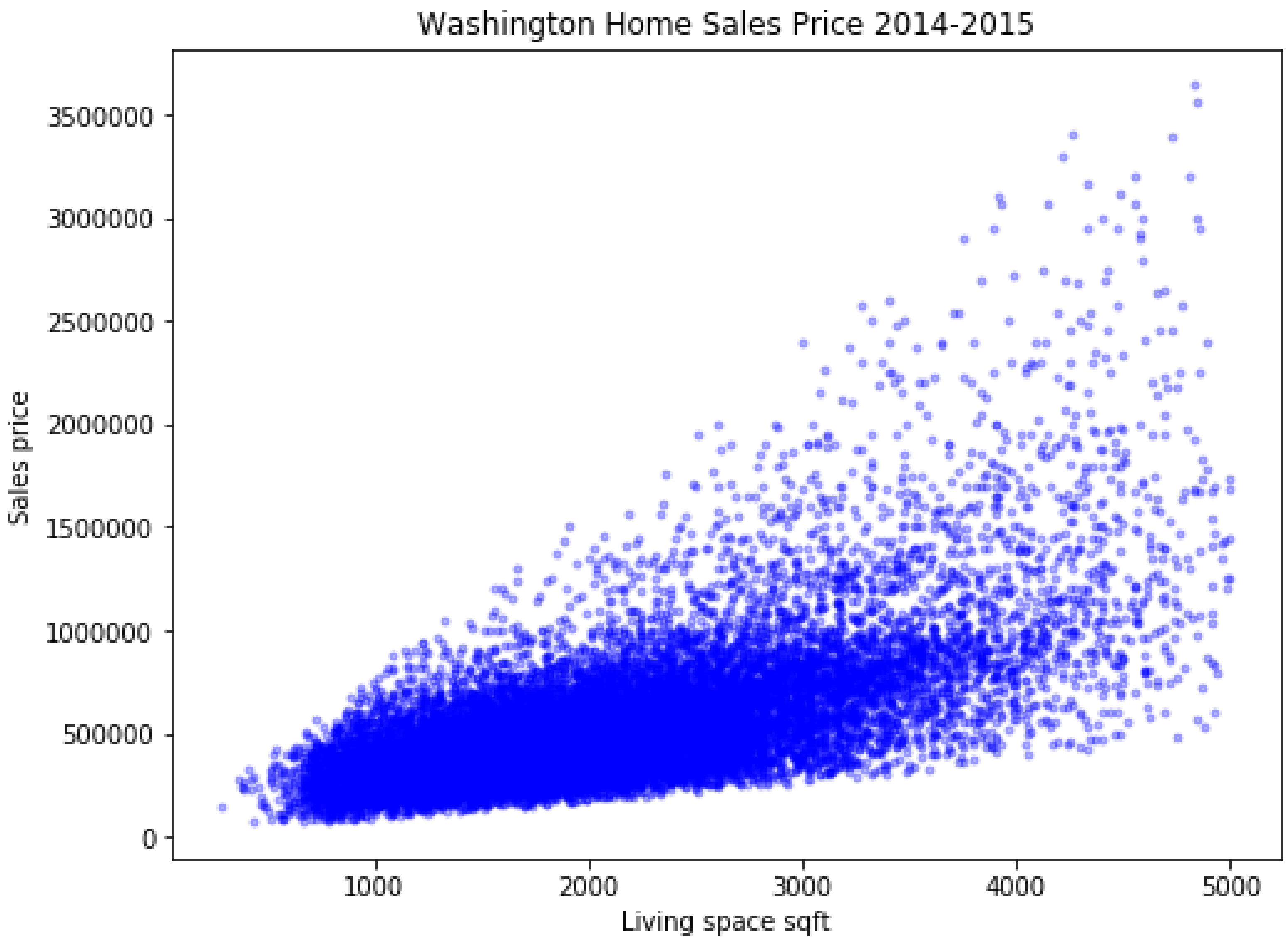
```
Data columns (total 21 columns):
id               21613 non-null int64
date             21613 non-null object
price            21613 non-null int64
bedrooms         21613 non-null int64
bathrooms        21613 non-null float64
sqft_living      21613 non-null int64
sqft_lot         21613 non-null int64
floors           21613 non-null float64
waterfront       21613 non-null int64
view             21613 non-null int64
condition        21613 non-null int64
grade            21613 non-null int64
sqft_above       21613 non-null int64
sqft_basement    21613 non-null int64
yr_built         21613 non-null int64
yr_renovated     21613 non-null int64
zipcode          21613 non-null int64
lat              21613 non-null float64
long             21613 non-null float64
sqft_living15    21613 non-null int64
sqft_lot15       21613 non-null int64
```

Example


Washington Home Sales Price 2014-2015
Sales price
8000000
7000000
6000000
5000000
4000000
3000000
2000000
1000000
0
0
2000
4000
6000
8000
10000
12000
14000
Living space sqft


10000
8000
6000
4000
2000
0
0
2000
4000
6000
8000
10000
12000

Example


Washington Home Sales Price 2014-2015
Sales price
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
1000
2000
3000
4000
5000
Living space sqft

Simple Linear Regression

Intercept Slope Residual

$$Y = \underbrace{\beta_0}_{} + \underbrace{\beta_1}_{} X + \underbrace{\epsilon}_{}$$

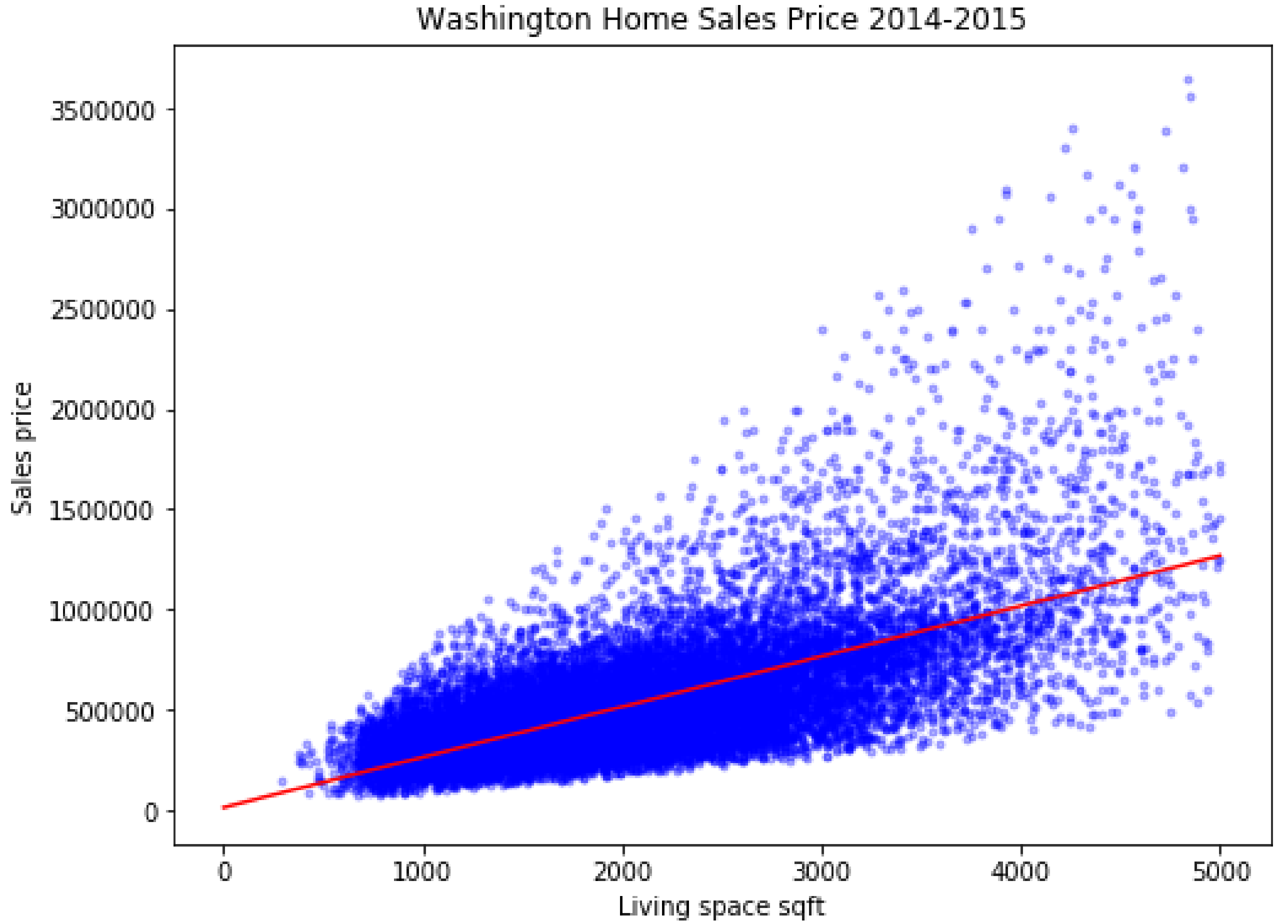
Coefficients, or Parameters

Estimated values

Model $$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

Example

```
import statsmodels.formula.api as smf
model = smf.ols(formula='price~sqft_living',data=sub).fit()
model.summary()
```

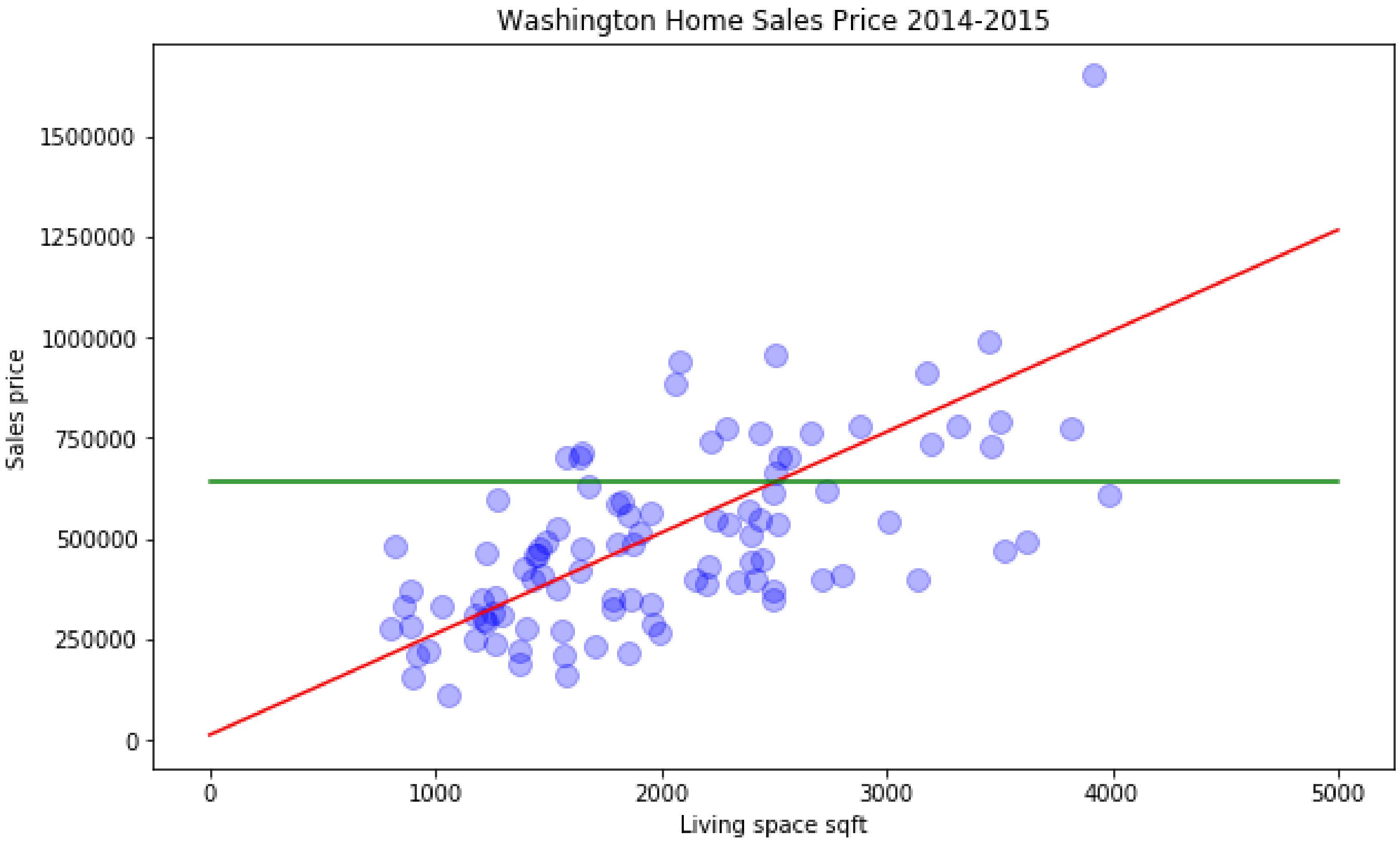


OLS Regression Results

Dep. Variable:	price	**R-squared:**	0.436
Model:	OLS	**Adj. R-squared:**	0.436
Method:	Least Squares	**F-statistic:**	1.651e+04
Date:	Tue, 14 Jan 2020	**Prob (F-statistic):**	0.00
Time:	19:04:13	**Log-Likelihood:**	-2.9523e+05
No. Observations:	21402	**AIC:**	5.905e+05
Df Residuals:	21400	**BIC:**	5.905e+05
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	1.24e+04	4305.280	2.881	0.004	3964.466	2.08e+04
sqft_living	251.0967	1.954	128.505	0.000	247.267	254.927

R-squared



$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

$$\mathrm{TSS} = \sum_{i=1}^{n} (y_i - \bar{y})^2$$

$$R^2 = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$$

p-value

t-statistics

Null hypothesis

$$H_0 : \beta_1 = 0$$

$$t = \frac{\hat{\beta}_1 - 0}{\mathrm{SE}(\hat{\beta}_1)}$$

Alternative hypothesis

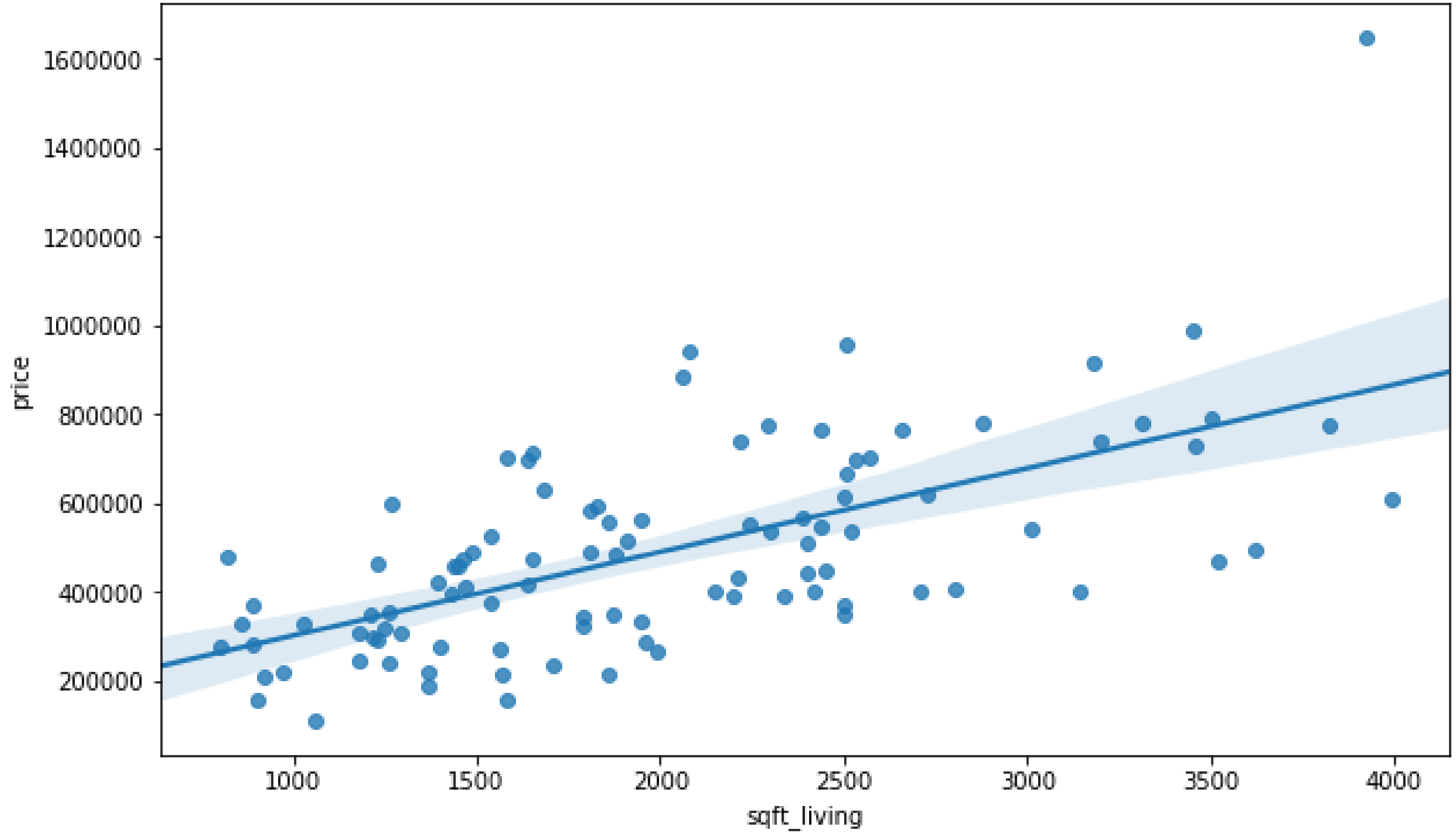
$$H_A : \beta_1 \neq 0$$

p-value

P > |*t*|

Accuracy of the coefficient estimates

```
import seaborn as sns
plt.figure(figsize=(10,6))
sns.regplot(x="sqft_living", y = "price", data=x_sample, ci=95)
```



95% Confidence Interval

$$\left[\hat{\beta}_1 - 2 \cdot \mathrm{SE}(\hat{\beta}_1),\ \hat{\beta}_1 + 2 \cdot \mathrm{SE}(\hat{\beta}_1)\right]$$

Under the hood...

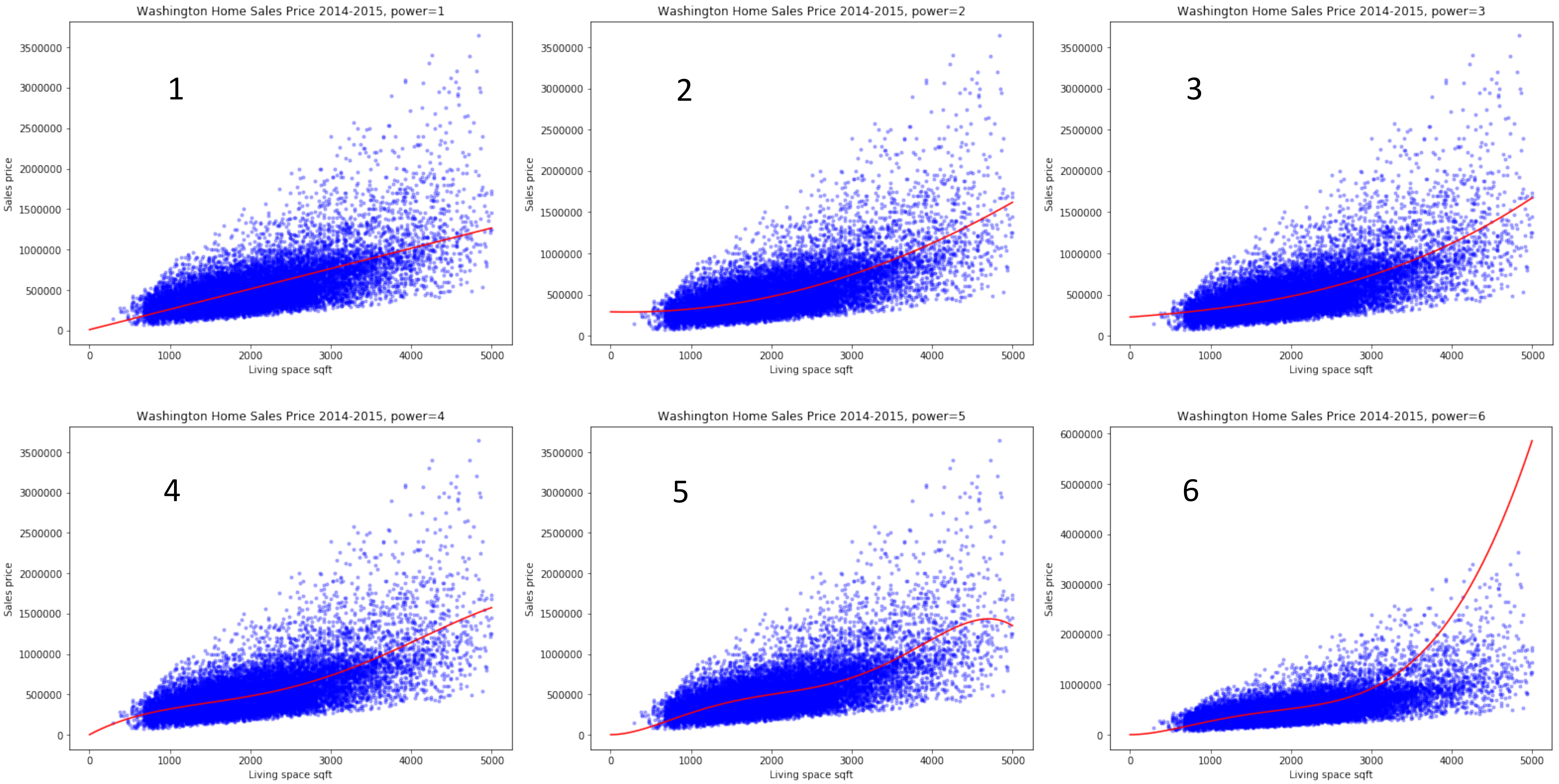
OLS Regression Results

Dep. Variable:	price	**R-squared:**	0.436
Model:	OLS	**Adj. R-squared:**	0.436
Method:	Least Squares	**F-statistic:**	1.651e+04
Date:	Tue, 14 Jan 2020	**Prob (F-statistic):**	0.00
Time:	19:04:13	**Log-Likelihood:**	-2.9523e+05
No. Observations:	21402	**AIC:**	5.905e+05
Df Residuals:	21400	**BIC:**	5.905e+05
Df Model:	1		
Covariance Type:	nonrobust		

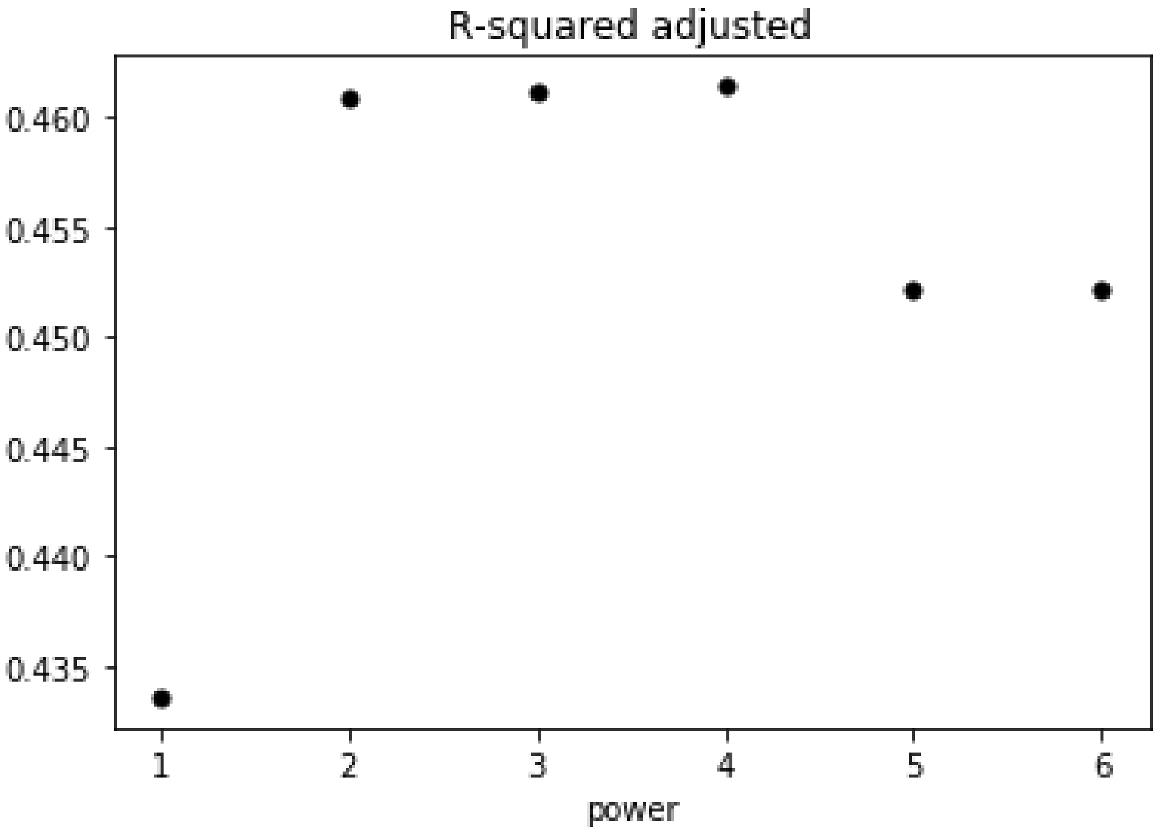
$$\prod_{i=1}^{n} p(y_i|x_i; b_0, b_1, s^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi s^2}} e^{-\frac{(y_i-(b_0+b_1x_i))^2}{2s^2}}$$

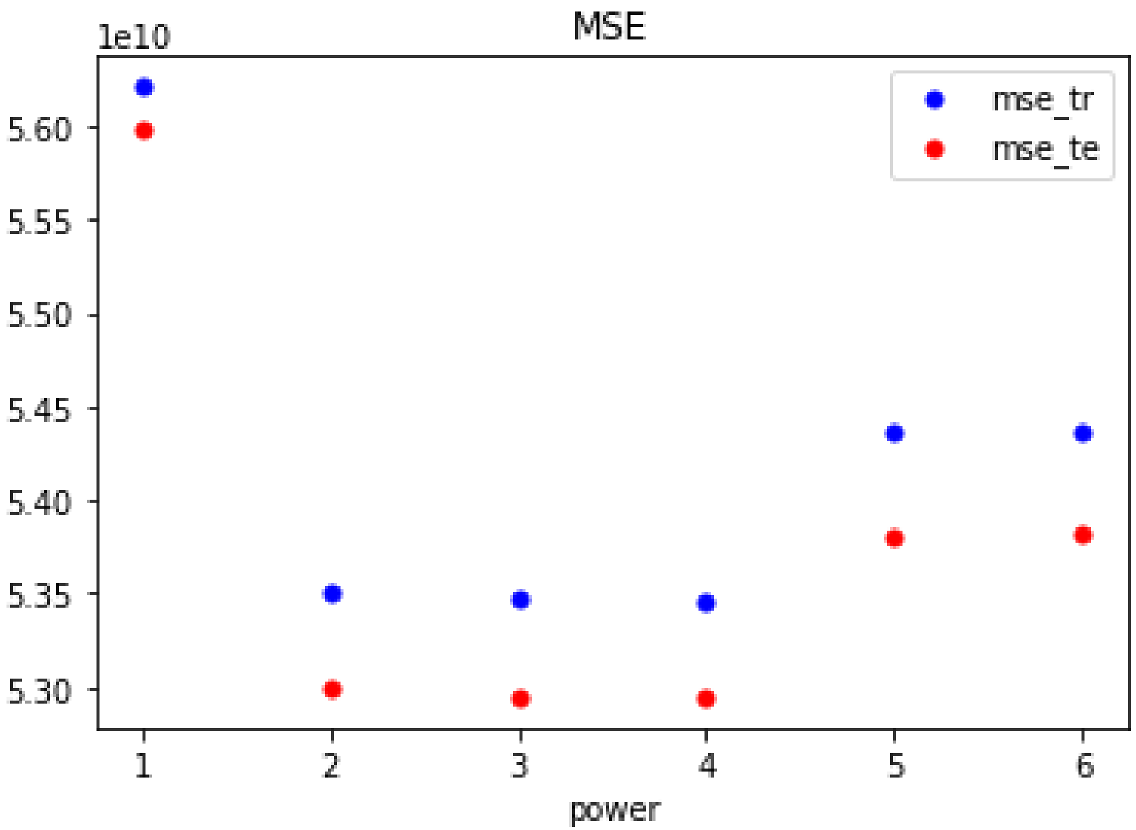
$$\begin{aligned} L(b_0, b_1, s^2) &= \log \prod_{i=1}^{n} p(y_i|x_i; b_0, b_1, s^2) \\ &= \sum_{i=1}^{n} \log p(y_i|x_i; b_0, b_1, s^2) \\ &= -\frac{n}{2}\log 2\pi - n\log s - \frac{1}{2s^2}\sum_{i=1}^{n}(y_i - (b_0 + b_1x_i))^2 \end{aligned}$$

Polynomial Regression

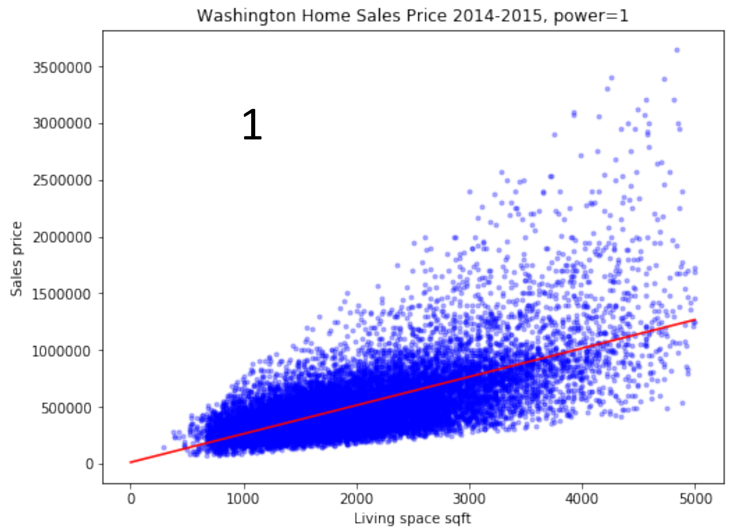


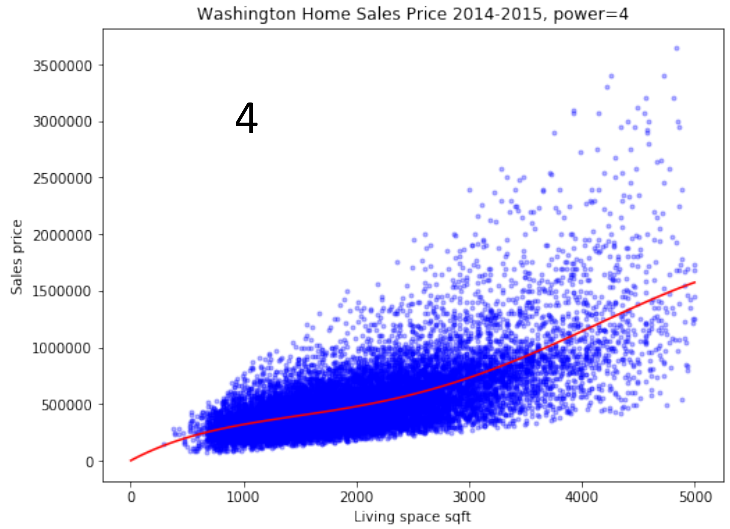
Where to stop?

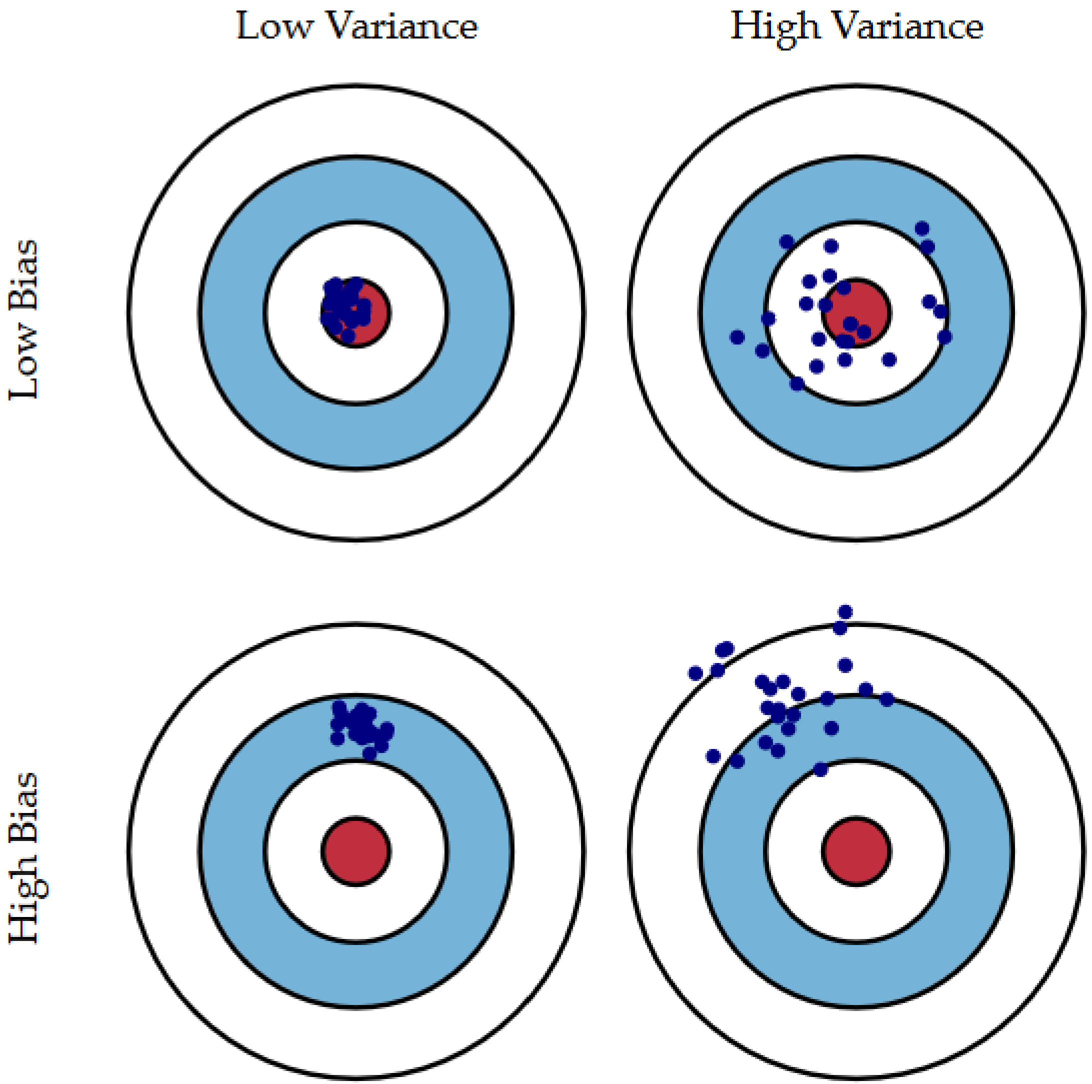




Bias-Variance Trade-off





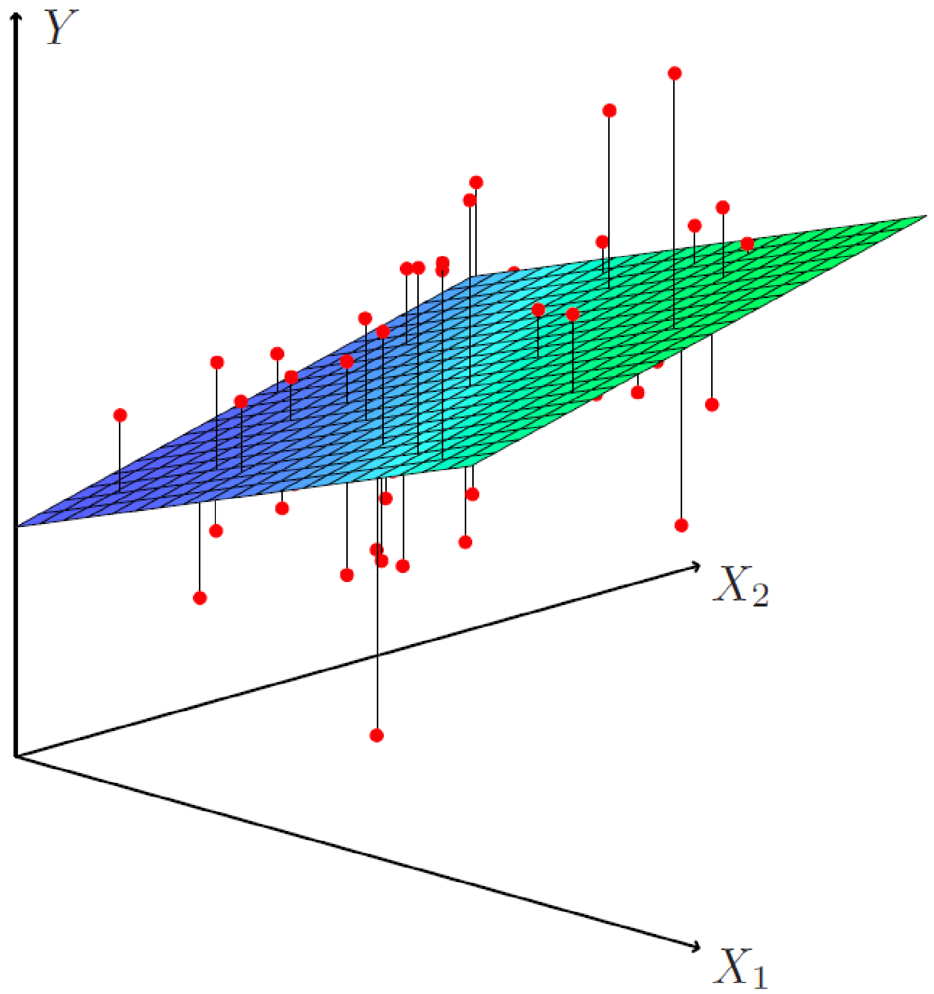


Multi-linear Regression

Multilinear regression model

All predictors(variables) X_1~X_p are linear to Y

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$



Multilinear regression model

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

β_j Average effect of X_j to Y when all other predictors fixed

Caution: In general,
predictors might be correlated

Collinearity

Caution: In general, predictors might be correlated

- When predictors are correlated, the coefficients estimates become inaccurate.

- Then it's called (multi-)collinear.

- The interpretation of the coefficient as the variable's contribution to the response becomes inaccurate.

Model selection

- Do I include all predictors or just a subset?

Forward Selection

step 1

```
y=b0
```

step 2

```
p=[]
for i=1,…,N
      y=b0+bi*Xi
      p.append(p_value(y))
k = argmin(p)
y = b0+bk*Xk
```

step 3

```
keep adding a predictor
that gives the lowest p-value
until it's satisfactory
```

Backward Selection

step 1

```
y0=b0+b1*X1+b2*X2+…+bN*XN
```

step 2

```
p=[]
for i=1,…,N
      y=y0-bi*Xi
      p.append(p_value(y))
k = argmax(p)
y = y0-bk*Xk
```

step 3

```
keep removing a predictor
that gives the largest p-value
until it's satisfactory
```