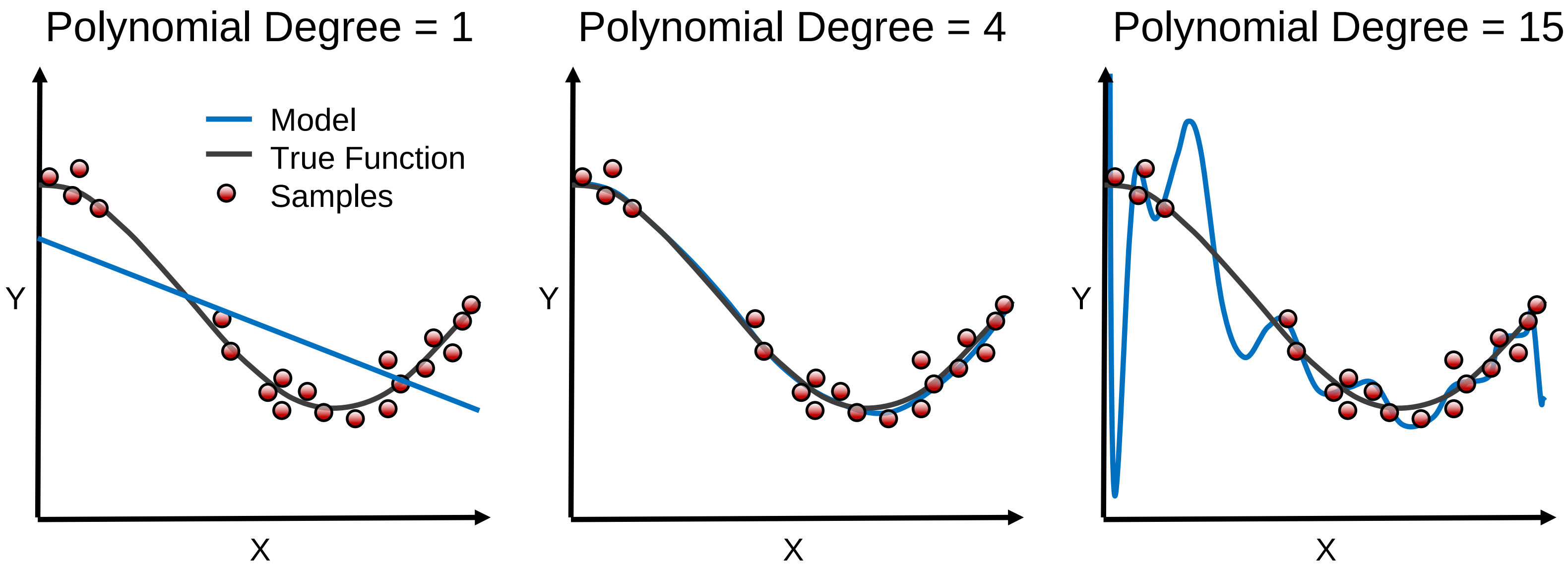
Chapter 3

Model Generalization
Advanced Linear Regression

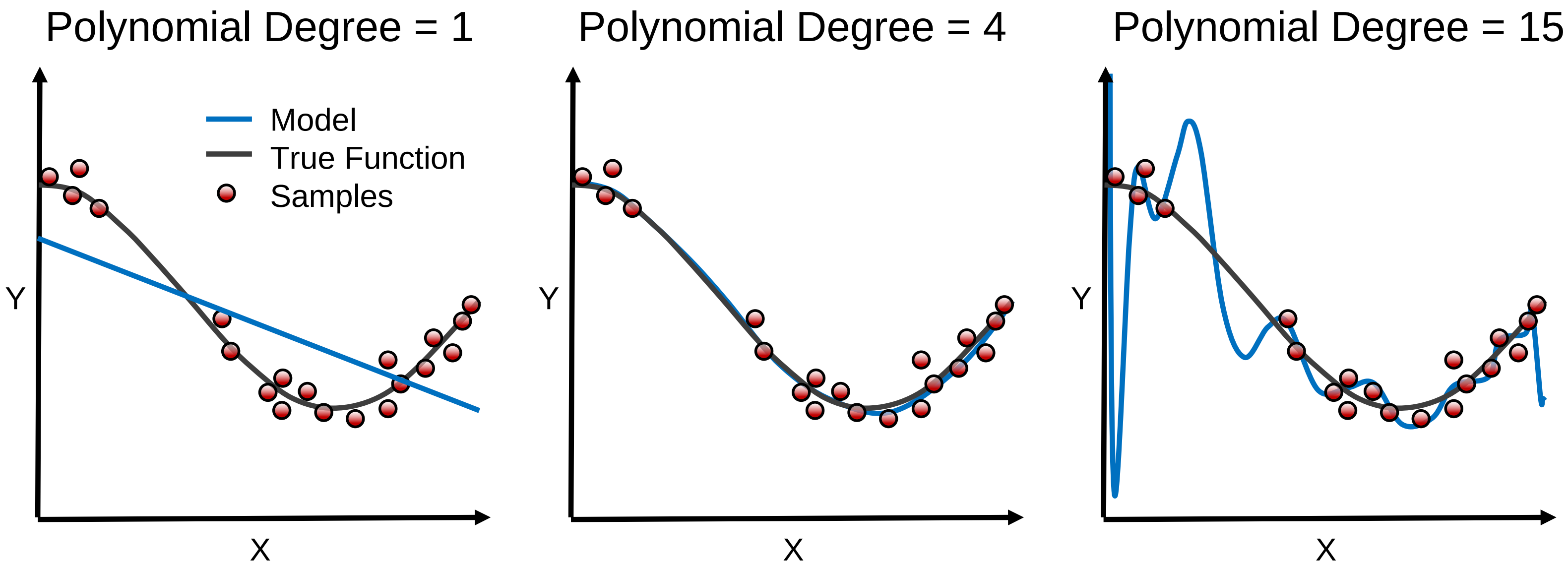
Model Generalization

Choosing Between Different Complexities



intel Software

How Well Does the Model Generalize?

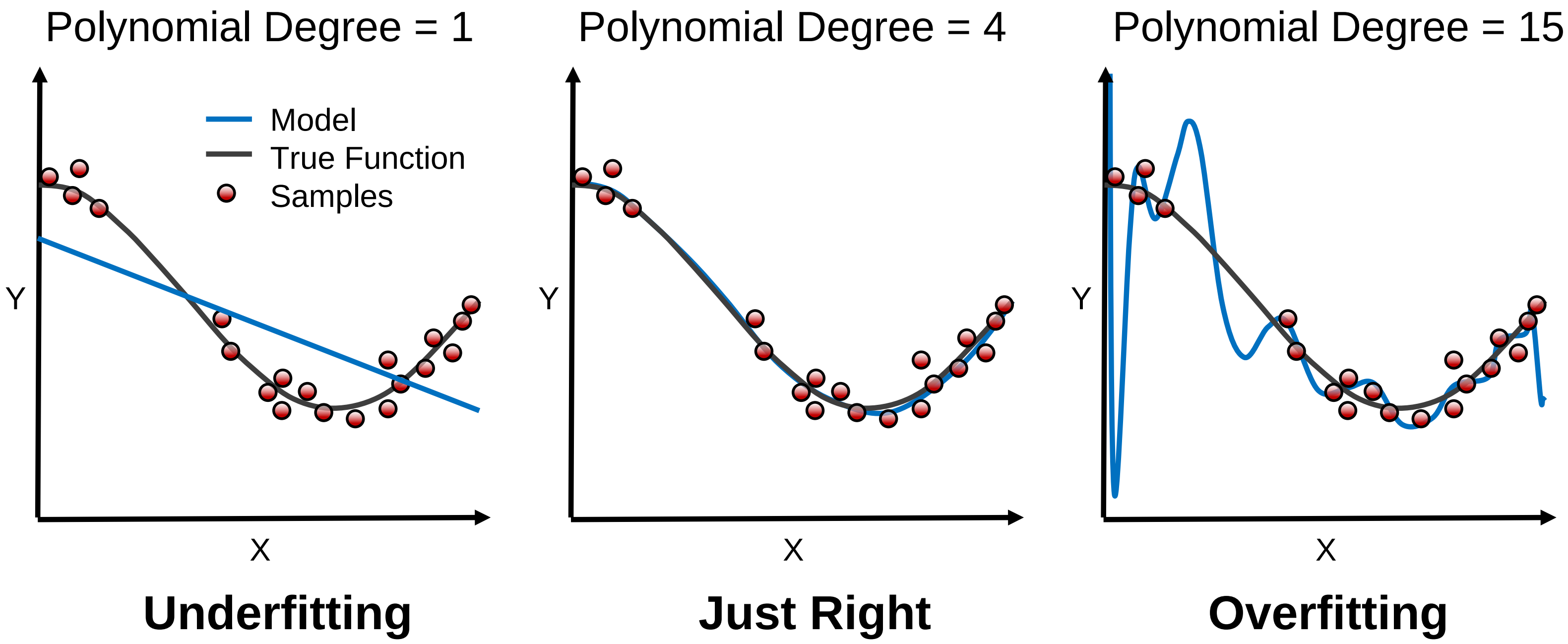


Poor at Training
Poor at Predicting

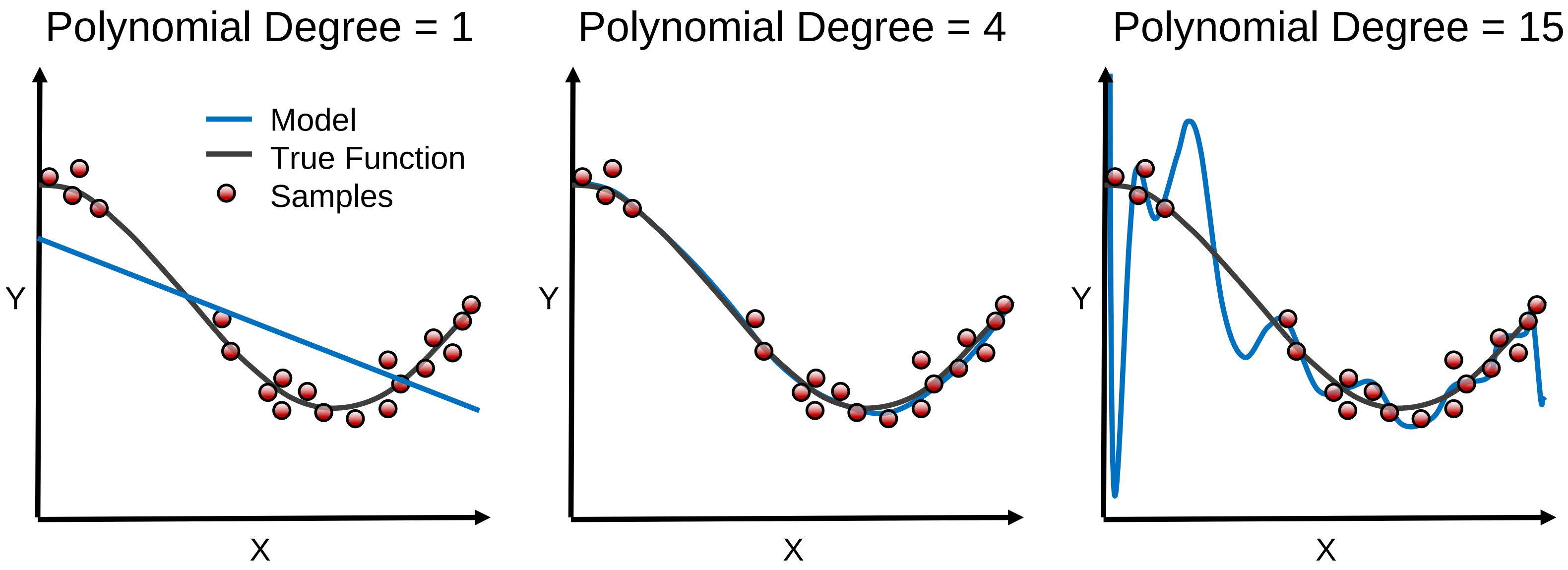
Just Right

Good at Training
Poor at Predicting

Underfitting vs Overfitting



Bias – Variance Tradeoff



High Bias
Low Variance

Just Right

Low Bias
High Variance

Training and Test Splits

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

Training and Test Splits

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

Training Data

Test Data

Using Training and Test Data

Training Data

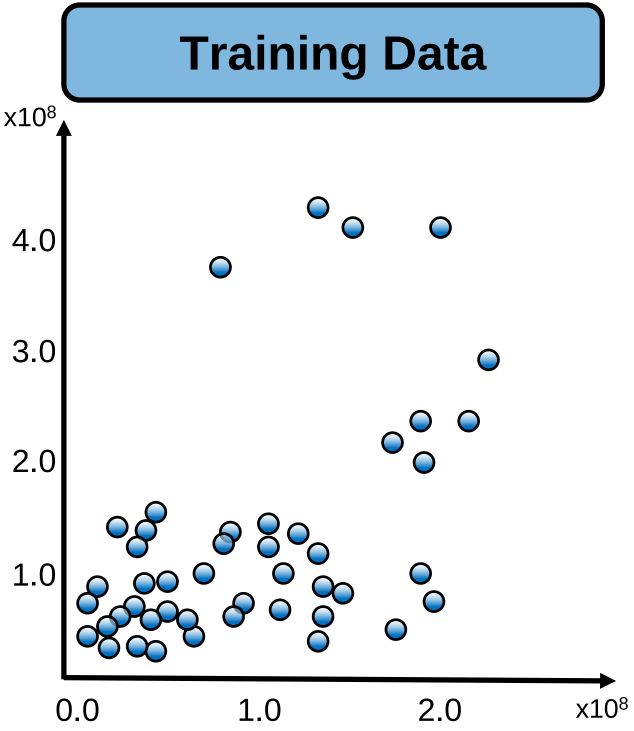
fit the model

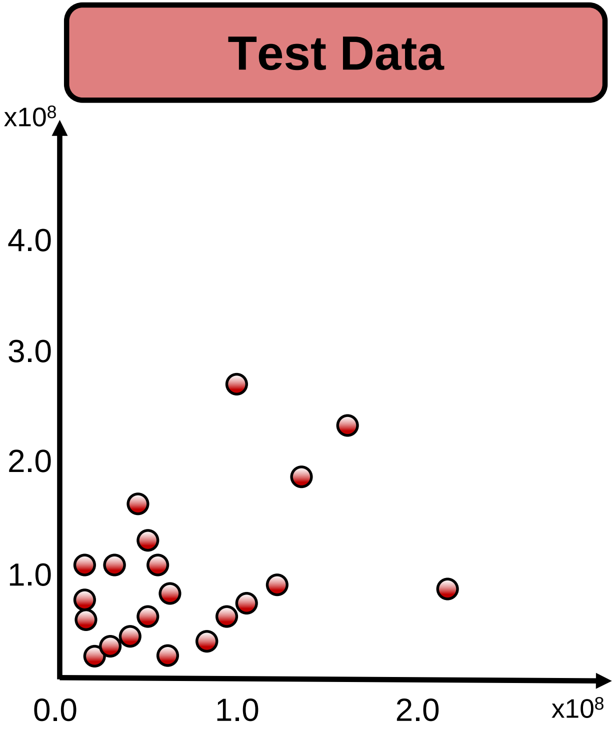
Test Data

measure performance

- predict label with model
- compare with actual value
- measure error

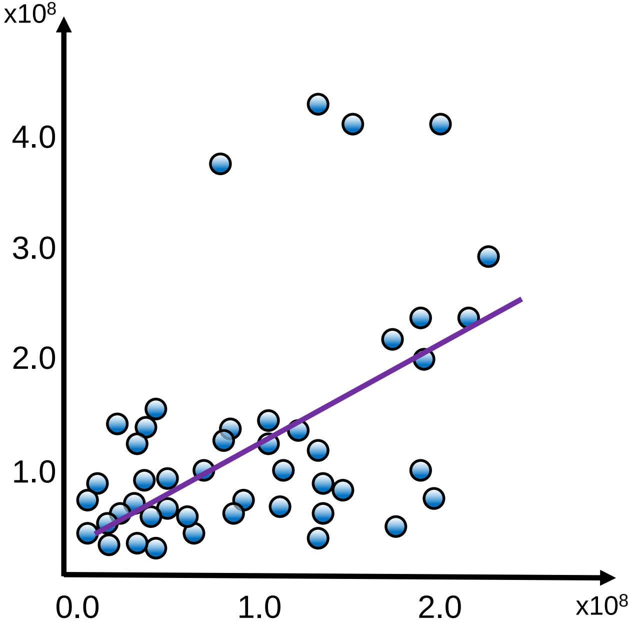
Using Training and Test Data



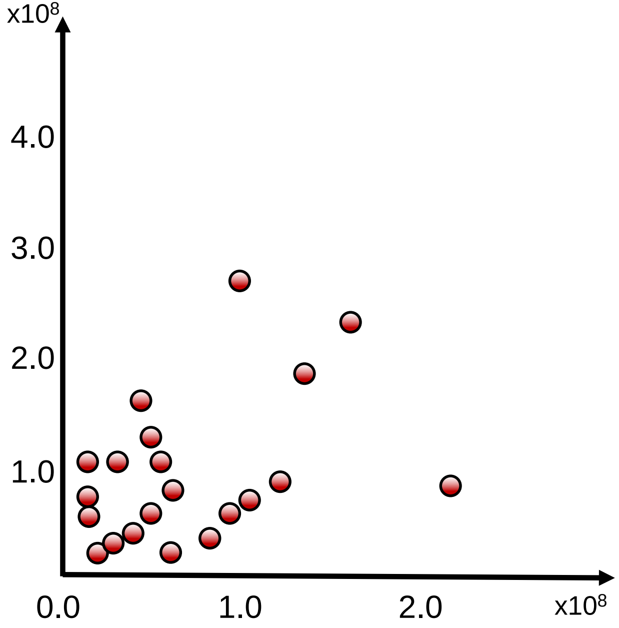


Using Training and Test Data

Training Data

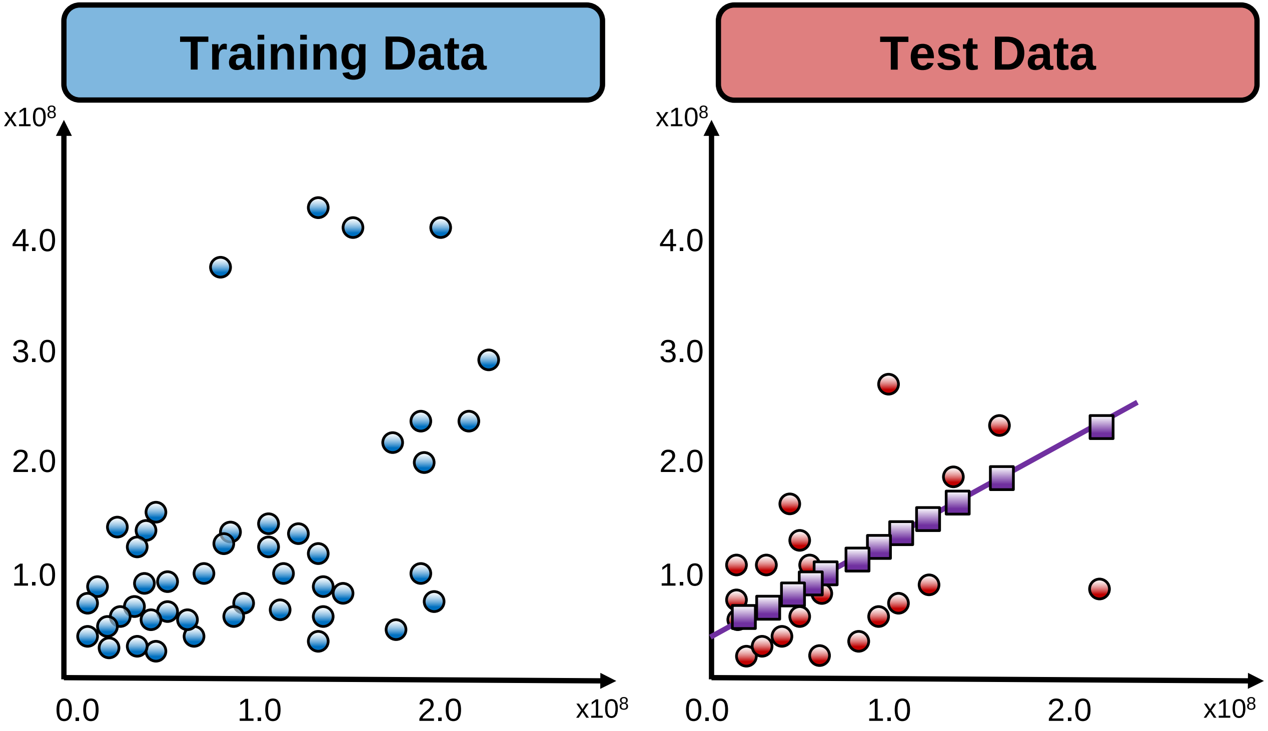


Test Data



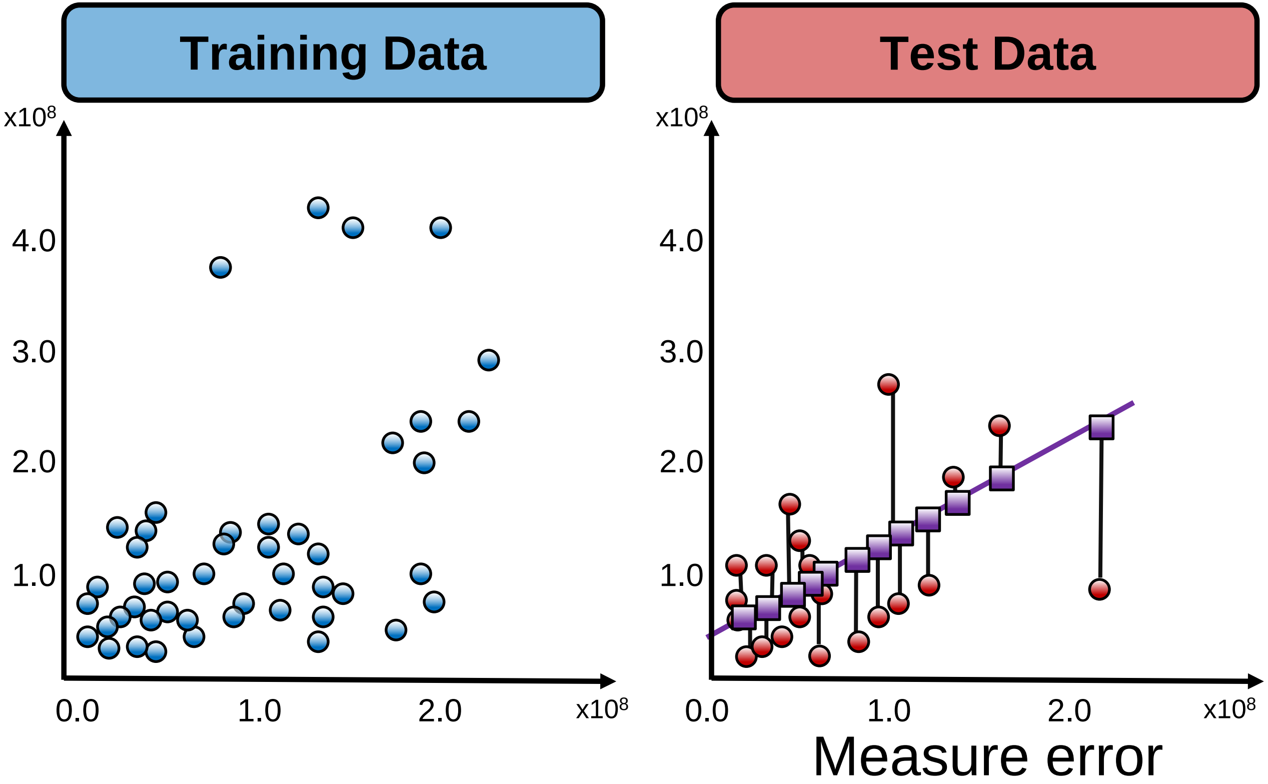
Fit the model

Using Training and Test Data

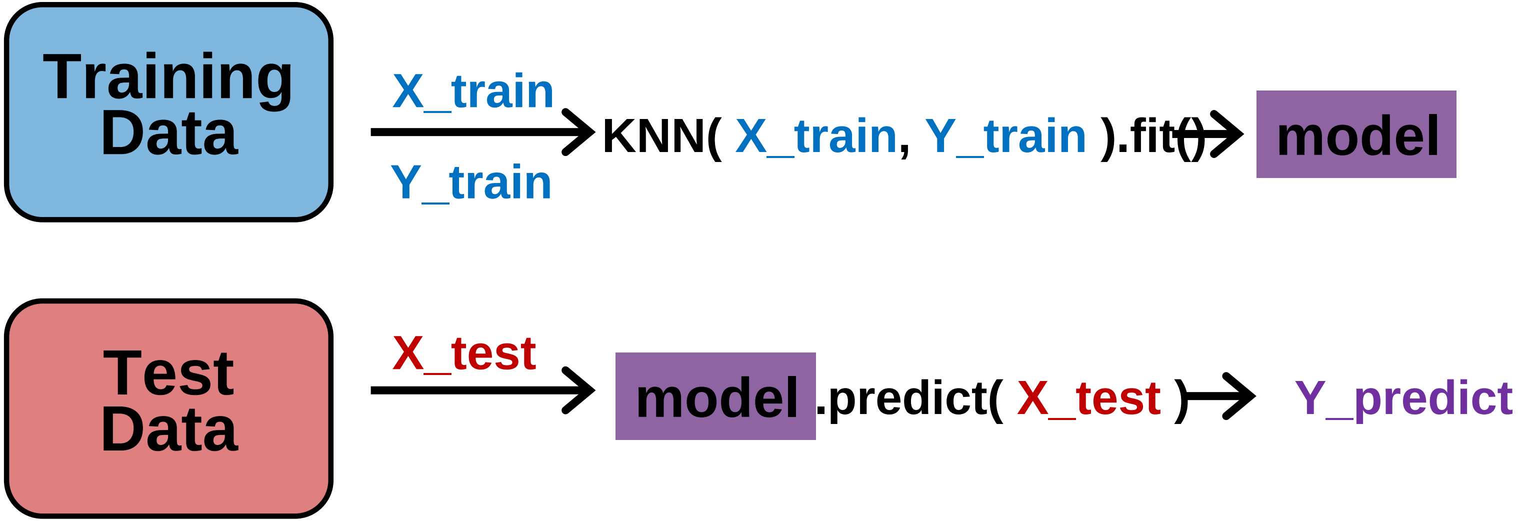


Make predictions

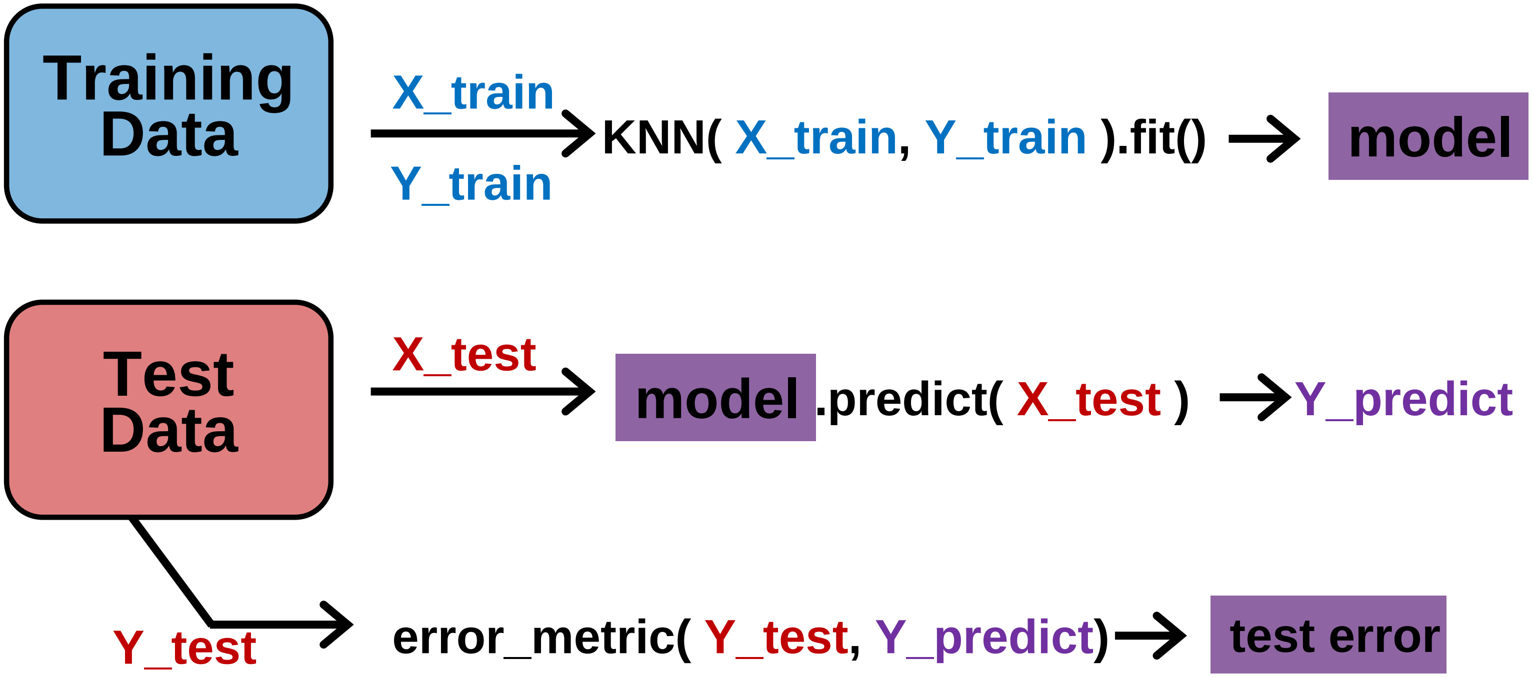
Using Training and Test Data



Fitting Training and Test Data



Fitting Training and Test Data



Train and Test Splitting: The Syntax

Import the train and test split function

from sklearn.model_selection import train_test_split

Software

Train and Test Splitting: The Syntax

Import the train and test split function

```
from sklearn.model_selection import train_test_split
```

Split the data and put 30% into the test set

```
train, test = train_test_split(data, test_size=0.3)
```

Train and Test Splitting: The Syntax

Import the train and test split function

```
from sklearn.model_selection import train_test_split
```

Split the data and put 30% into the test set

```
train, test = train_test_split(data, test_size=0.3)
```

Other method for splitting data:

```
from sklearn.model_selection import ShuffleSplit
```

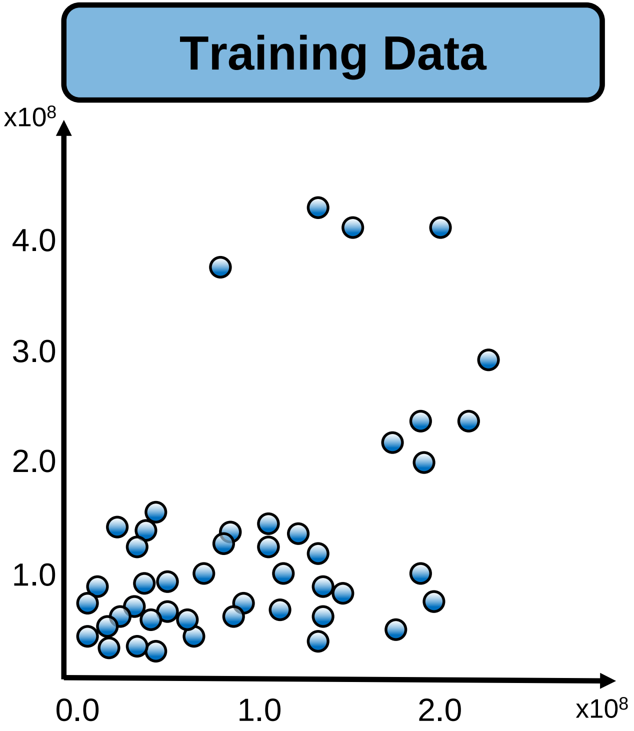
Beyond a Single Test Set: Cross Validation

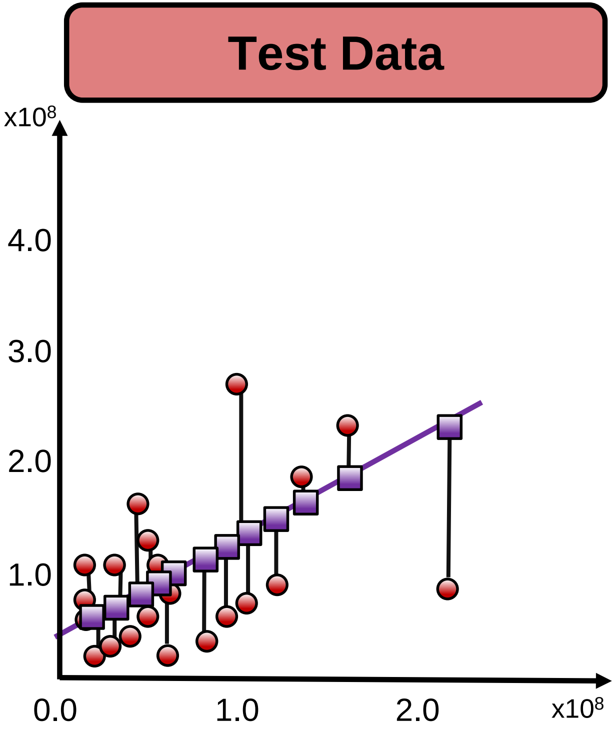
	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

Training Data

Validation Data

Beyond a Single Test Set: Cross Validation





Best model for **this** test set

Beyond a Single Test Set: Cross Validation

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

Training Data 1

Validation Data 1

Beyond a Single Test Set: Cross Validation

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

Training Data 2

Validation Data 2

Beyond a Single Test Set: Cross Validation

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

Validation Data 3

Training Data 3

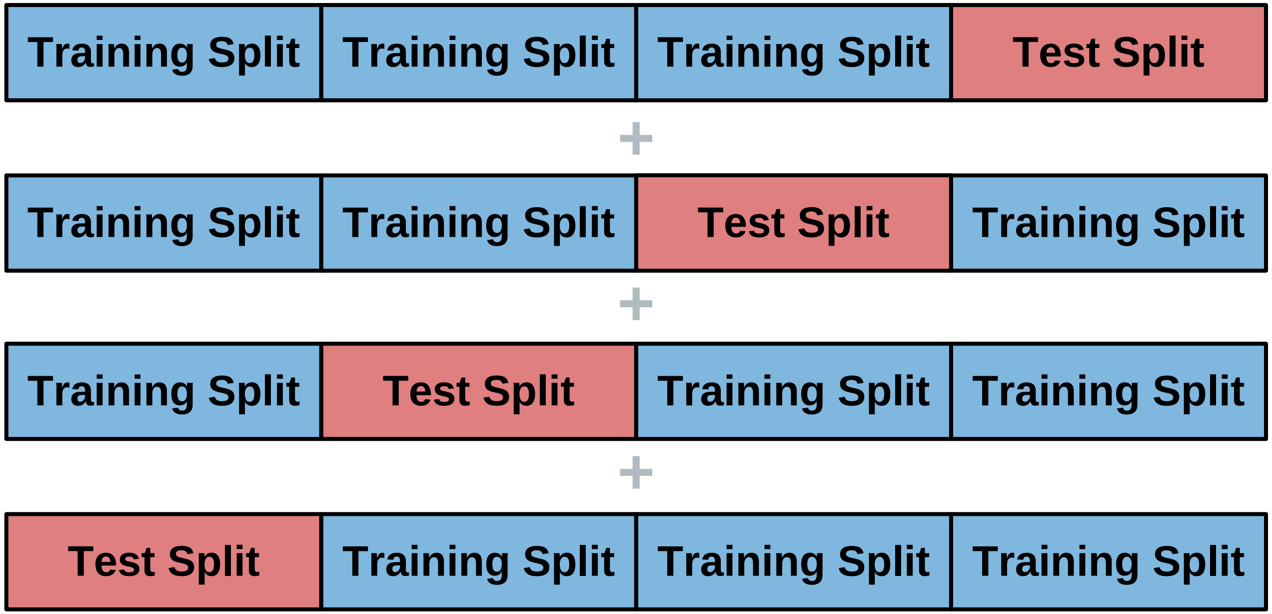
Beyond a Single Test Set: Cross Validation

	Date	Title	Budget	DomesticTotalGross	Director	Rating	Runtime
0	2013-11-22	The Hunger Games: Catching Fire	130000000	424668047	Francis Lawrence	PG-13	146
1	2013-05-03	Iron Man 3	200000000	409013994	Shane Black	PG-13	129
2	2013-11-22	Frozen	150000000	400738009	Chris BuckJennifer Lee	PG	108
3	2013-07-03	Despicable Me 2	76000000	368061265	Pierre CoffinChris Renaud	PG	98
4	2013-06-14	Man of Steel	225000000	291045518	Zack Snyder	PG-13	143
5	2013-10-04	Gravity	100000000	274092705	Alfonso Cuaron	PG-13	91
6	2013-06-21	Monsters University	NaN	268492764	Dan Scanlon	G	107
7	2013-12-13	The Hobbit: The Desolation of Smaug	NaN	258366855	Peter Jackson	PG-13	161
8	2013-05-24	Fast & Furious 6	160000000	238679850	Justin Lin	PG-13	130
9	2013-03-08	Oz The Great and Powerful	215000000	234911825	Sam Raimi	PG	127
10	2013-05-16	Star Trek Into Darkness	190000000	228778661	J.J. Abrams	PG-13	123
11	2013-11-08	Thor: The Dark World	170000000	206362140	Alan Taylor	PG-13	120
12	2013-06-21	World War Z	190000000	202359711	Marc Forster	PG-13	116
13	2013-03-22	The Croods	135000000	187168425	Kirk De MiccoChris Sanders	PG	98
14	2013-06-28	The Heat	43000000	159582188	Paul Feig	R	117
15	2013-08-07	We're the Millers	37000000	150394119	Rawson Marshall Thurber	R	110
16	2013-12-13	American Hustle	40000000	150117807	David O. Russell	R	138
17	2013-05-10	The Great Gatsby	105000000	144840419	Baz Luhrmann	PG-13	143

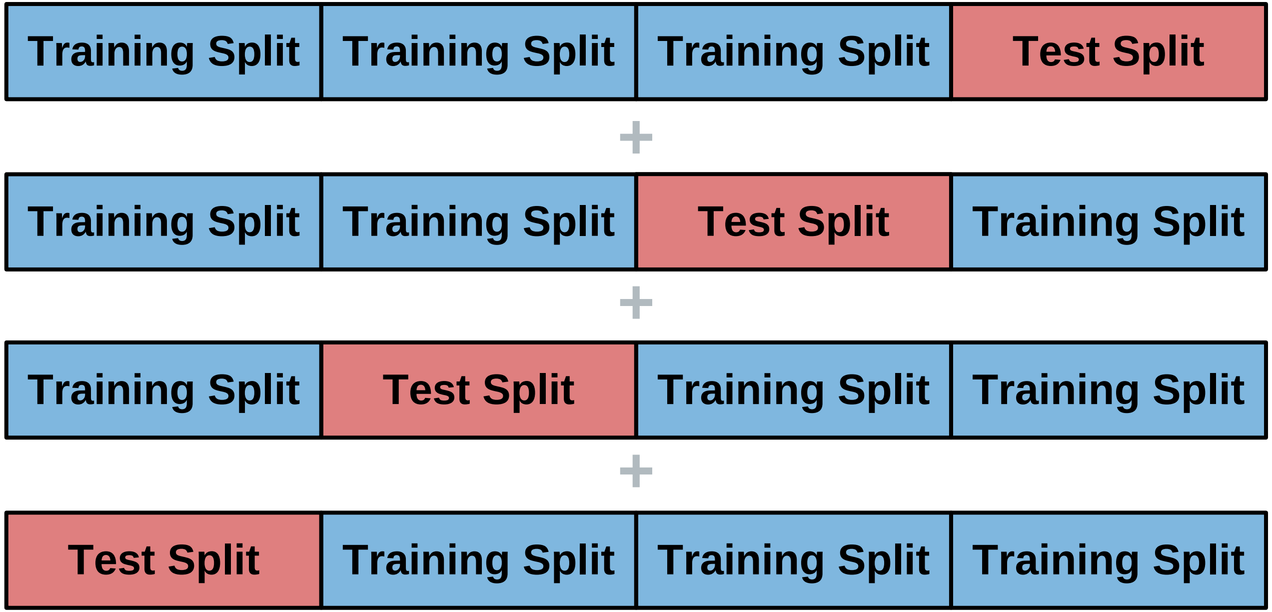
Validation Data 4

Training Data 4

Beyond a Single Test Set: Cross Validation

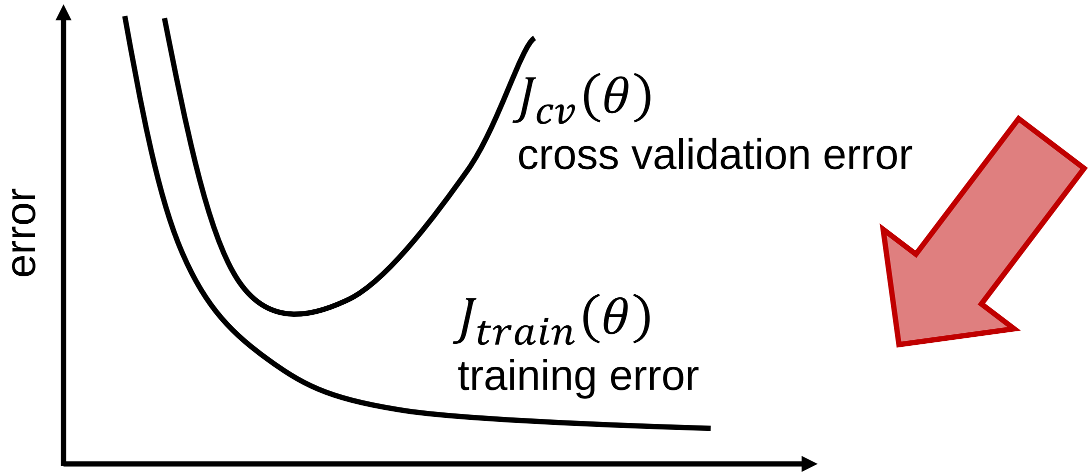


Beyond a Single Test Set: Cross Validation

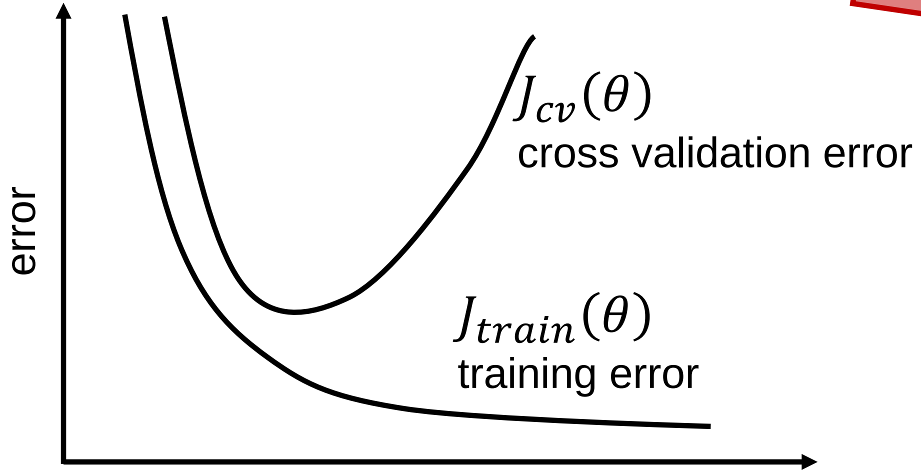


Average cross validation results.

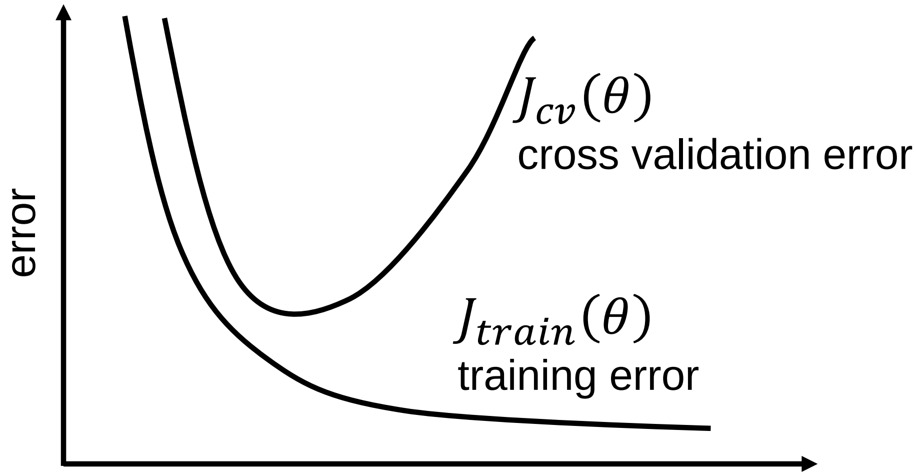
Model Complexity vs Error



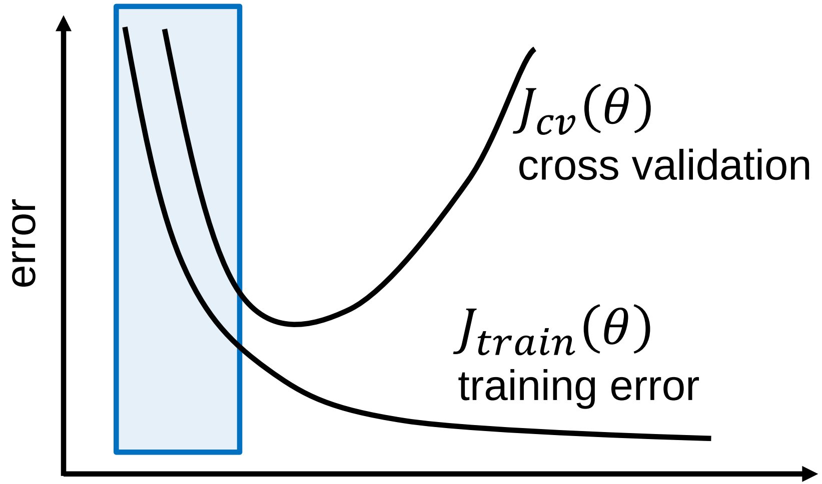
Model Complexity vs Error

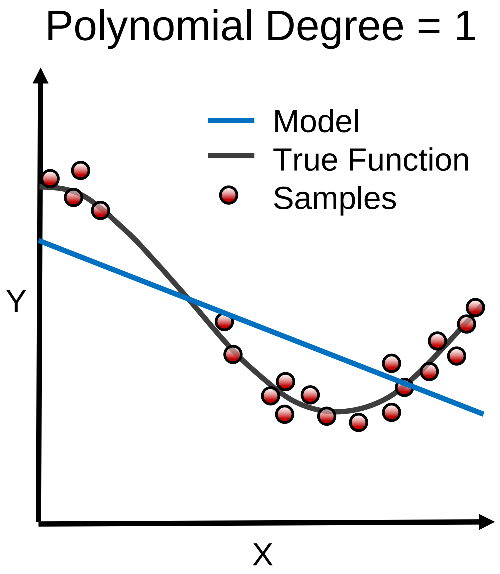


Model Complexity vs Error



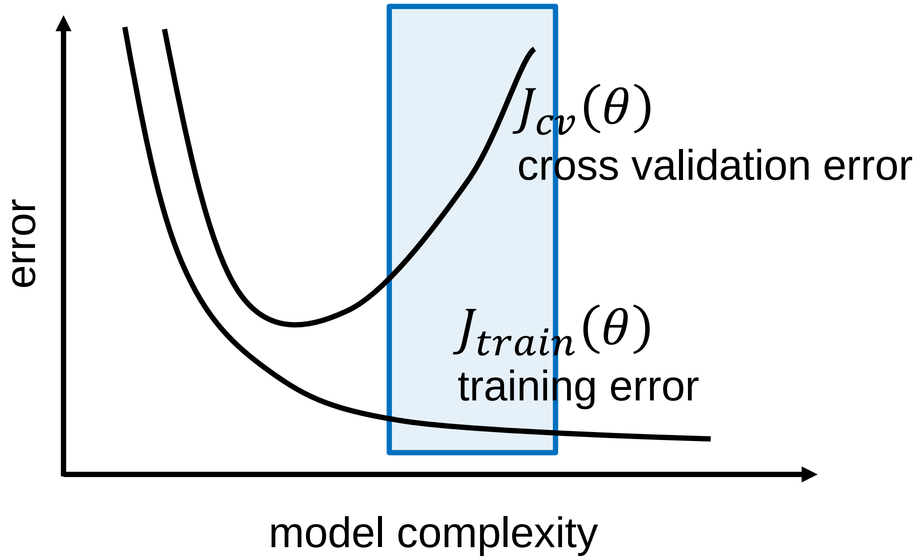
Model Complexity vs Error

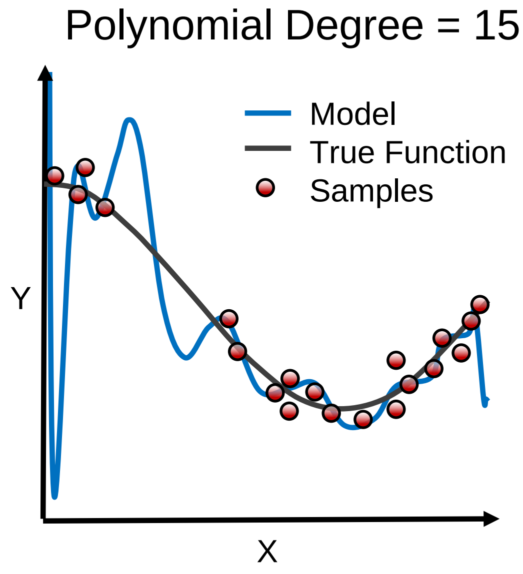




Underfitting: training and cross validation error are high

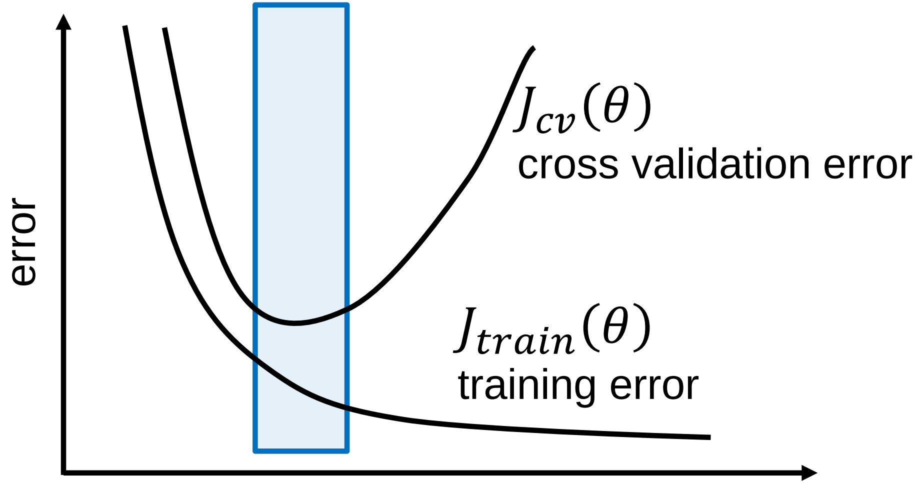
Model Complexity vs Error

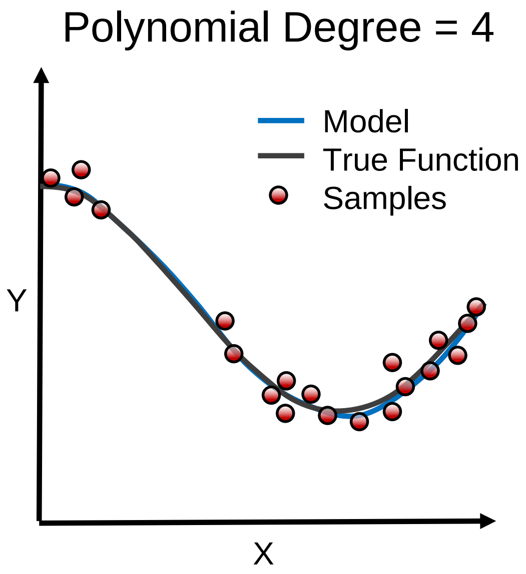




Overfitting: training error is low, cross validation is high

Model Complexity vs Error





Just right: training and cross validation errors are low

Cross Validation: The Syntax

Import the train and test split function

```
from sklearn.model_selection import cross_val_score
```

Cross Validation: The Syntax

Import the train and test split function

```
from sklearn.model_selection import cross_val_score
```

Perform cross-validation with a given model

```
cross_val = cross_val_score(KNN, X_data, y_data, cv=4,
                            scoring='neg_mean_squared_error')
```

Cross Validation: The Syntax

Import the train and test split function

```
from sklearn.model_selection import cross_val_score
```

Perform cross-validation with a given model

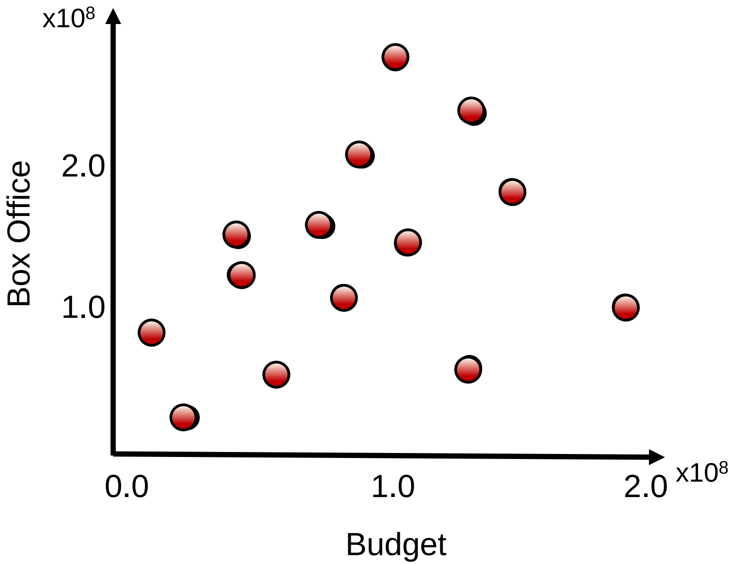
```
cross_val = cross_val_score(KNN, X_data, y_data, cv=4,
                            scoring='neg_mean_squared_error')
```

Other methods for cross validation:

```
from sklearn.model_selection import KFold, StratifiedKFold
```

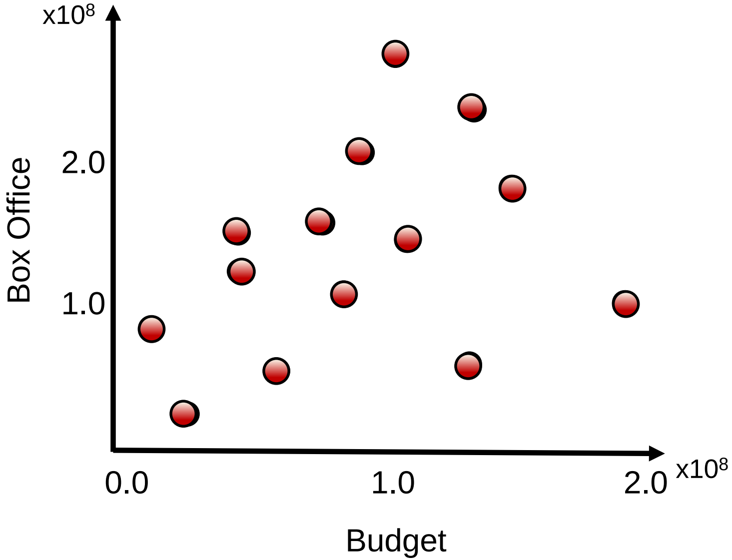
Linear Regression

Introduction to Linear Regression



$$y_\beta(x) = \beta_0 + \beta_1 x$$

Introduction to Linear Regression



$$y_\beta(x) = \beta_0 + \beta_1 x$$

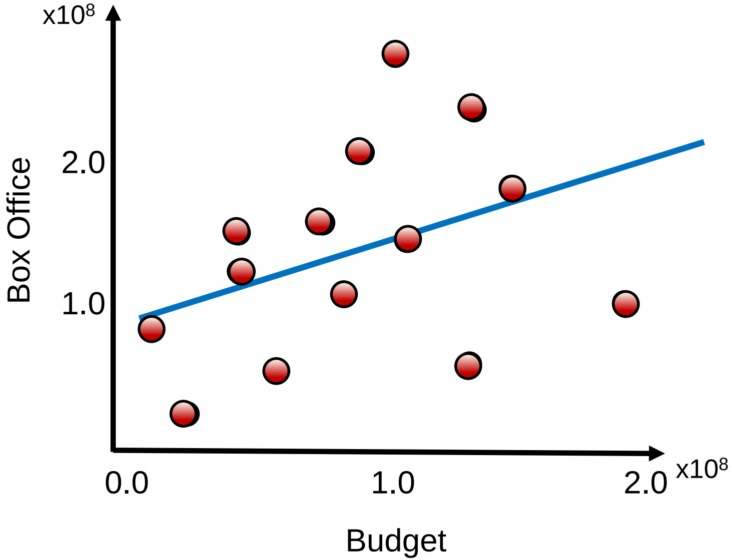
box office revenue

coefficient 0

coefficient 1

movie budget

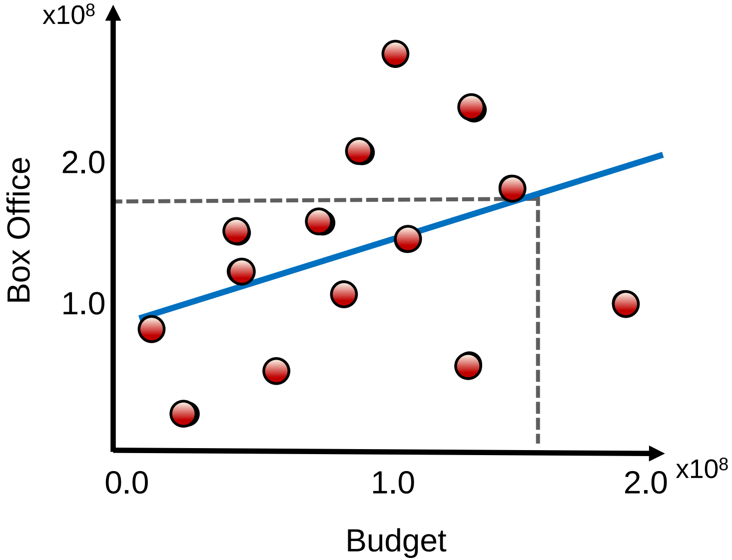
Introduction to Linear Regression



$$y_\beta(x) = \beta_0 + \beta_1 x$$

β_0= 80 million, β_1= 0.6

Predicting from Linear Regression

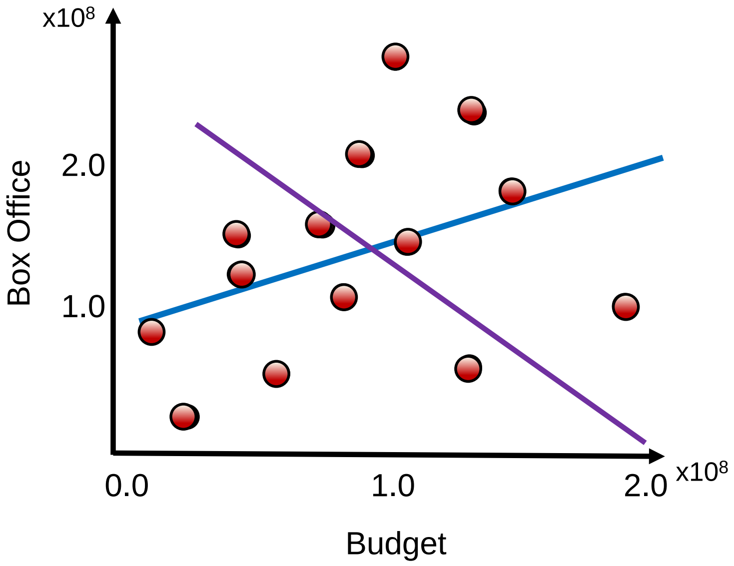


$$y_{\beta}(x) = \beta_0 + \beta_1 x$$

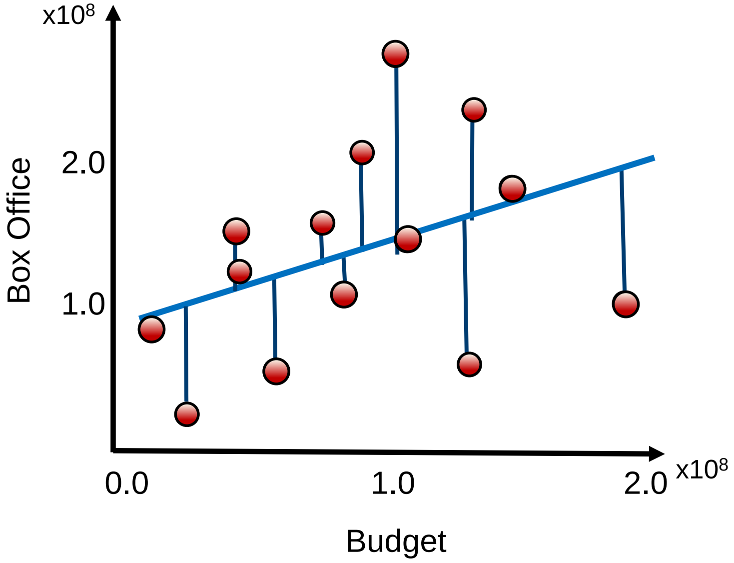
β_0= 80 million, β_1= 0.6

Predict 175 Million Gross for 160 Million Budget

Which Model Fits the Best?

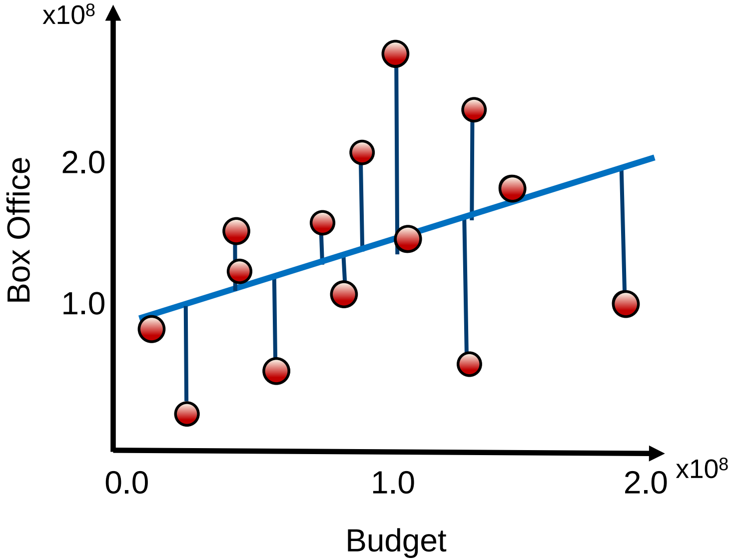


Calculating the Residuals



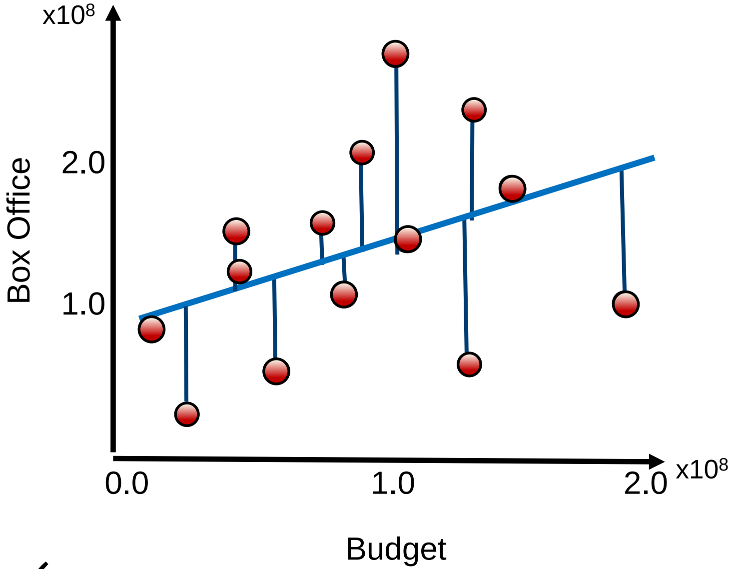
predicted value $\rightarrow$ $$y_\beta\left(x_{obs}^{(i)}\right) - y_{obs}^{(i)}$$ $\leftarrow$ observed value

Calculating the Residuals



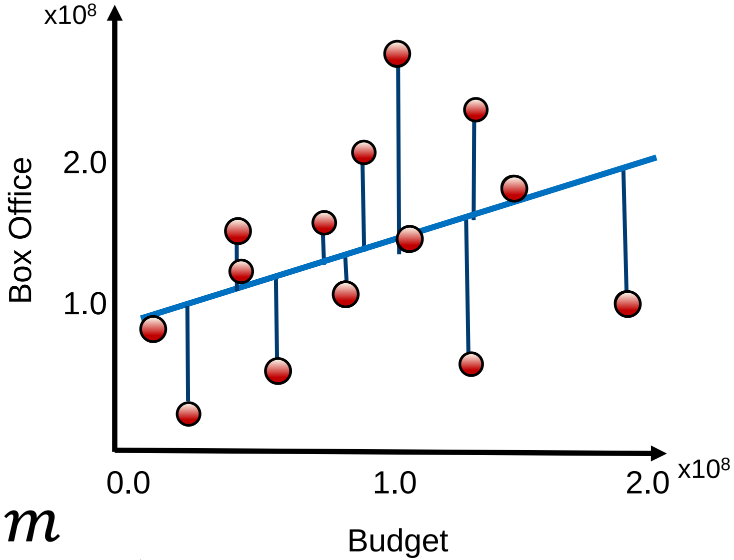
$$\left(\beta_0 + \beta_1 x_{obs}^{(i)}\right) - y_{obs}^{(i)}$$

Mean Squared Error



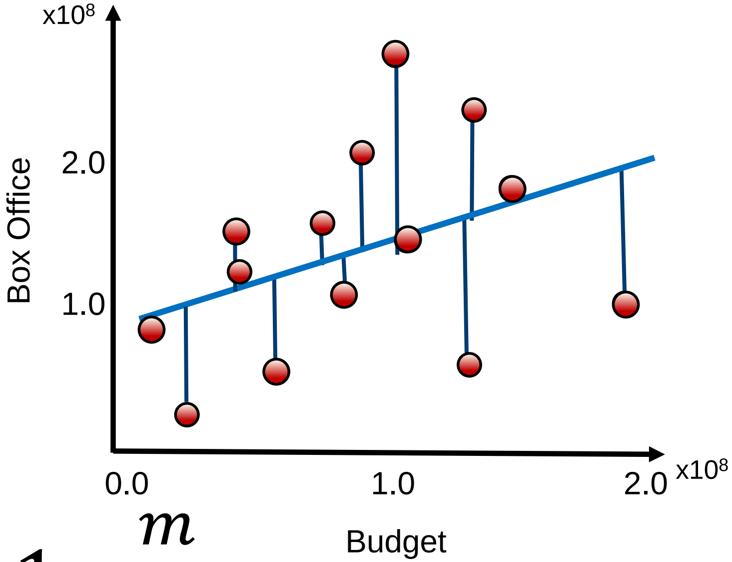
$$\frac{1}{m}\sum_{i=1}^{m}\left(\left(\beta_0+\beta_1 x_{obs}^{(i)}\right)-y_{obs}^{(i)}\right)^2$$

Minimum Mean Squared Error



$$\min_{\beta_0,\beta_1} \frac{1}{m} \sum_{i=1}^{m} \left(\left(\beta_0 + \beta_1 x_{obs}^{(i)} \right) - y_{obs}^{(i)} \right)^2$$

Cost Function



$$J(\beta_0, \beta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(\left(\beta_0 + \beta_1 x_{obs}^{(i)} \right) - y_{obs}^{(i)} \right)^2$$

Modelling Best Practice

- Use cost function to fit model
- Develop multiple models
- Compare results and choose best one

Other Model Metrics

Sum of Squared Error (SSE):

$$\sum_{i=1}^{m}\left(y_{\beta}(x^{(i)}) - y_{obs}^{(i)}\right)^2$$

Other Measures of Error

Sum of Squared Error (SSE): $$\sum_{i=1}^{m}\left(y_{\beta}(x^{(i)}) - y_{obs}^{(i)}\right)^2$$

Total Sum of Squares (TSS): $$\sum_{i=1}^{m}\left(\overline{y_{obs}} - y_{obs}^{(i)}\right)^2$$

Other Measures of Error

Sum of Squared Error (SSE):

$$\sum_{i=1}^{m} \left(y_{\beta}(x^{(i)}) - y_{obs}^{(i)} \right)^2$$

Total Sum of Squares (TSS):

$$\sum_{i=1}^{m} \left(\overline{y_{obs}} - y_{obs}^{(i)} \right)^2$$

Correlation Coefficient (R^2):

$$1 - \frac{SSE}{TSS}$$

Comparing Linear Regression and KNN

Linear Regression	K Nearest Neighbors
• Fitting involves minimizing cost function (*slow*)	• Fitting involves storing training data (*fast*)

Comparing Linear Regression and KNN

Linear Regression	K Nearest Neighbors
• Fitting involves minimizing cost function (*slow*)	• Fitting involves storing training data (*fast*)
• Model has few parameters (*memory efficient*)	• Model has many parameters (*memory intensive*)

Comparing Linear Regression and KNN

Linear Regression

- Fitting involves minimizing cost function (*slow*)
- Model has few parameters (*memory efficient*)
- Prediction involves calculation (*fast*)

K Nearest Neighbors

- Fitting involves storing training data (*fast*)
- Model has many parameters (*memory intensive*)
- Prediction involves finding closest neighbors (*slow*)

Linear Regression: The Syntax

Import the class containing the regression method

```
from sklearn.linear_model import LinearRegression
```

Linear Regression: The Syntax

Import the class containing the regression method

```
from sklearn.linear_model import LinearRegression
```

Create an instance of the class

```
LR = LinearRegression()
```

Linear Regression: The Syntax

Import the class containing the regression method

```
from sklearn.linear_model import LinearRegression
```

Create an instance of the class

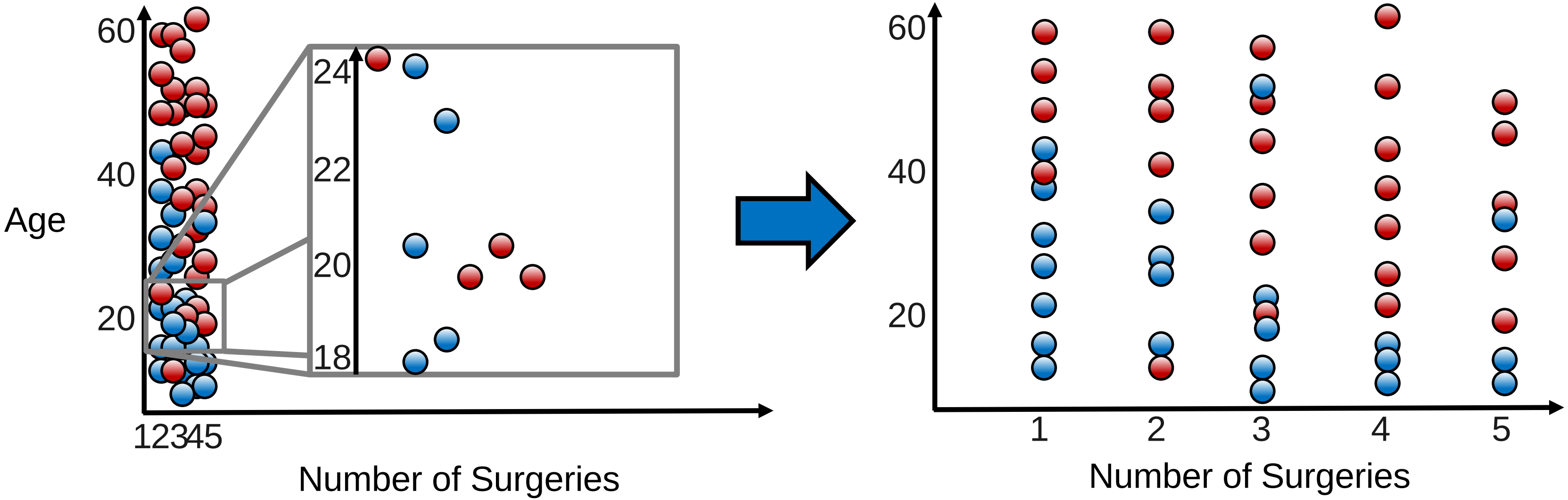
```
LR = LinearRegression()
```

Fit the instance on the data and then predict the expected value

```
LR = LR.fit(X_train, y_train)
y_predict = LR.predict(X_test)
```

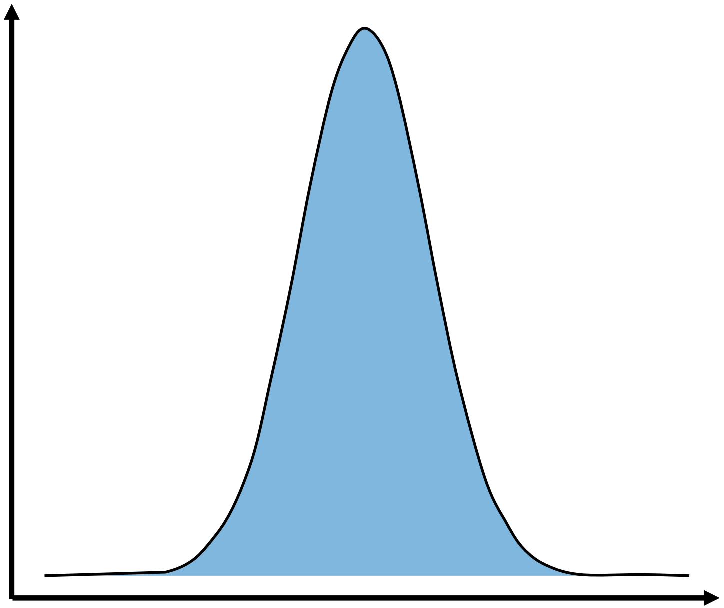
Advanced Linear Regression

Scaling is a Type of Feature Transformation



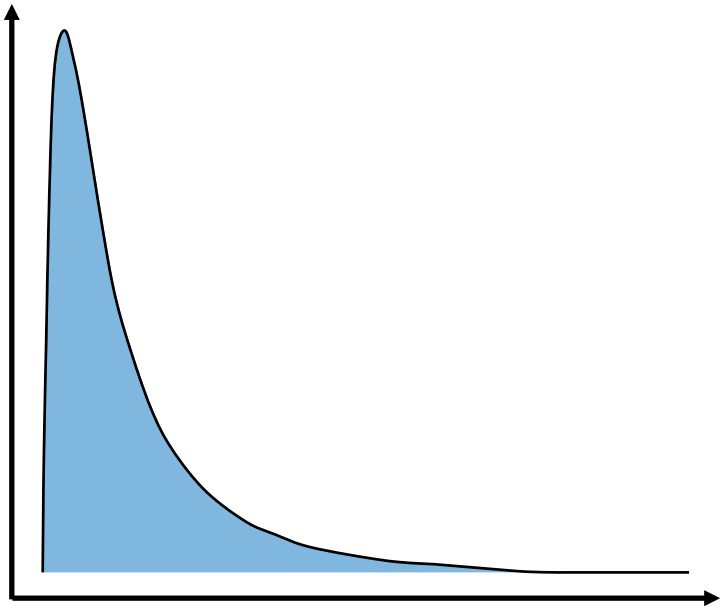
Transformation of Data Distributions

- Predictions from linear regression models assume residuals are normally distributed

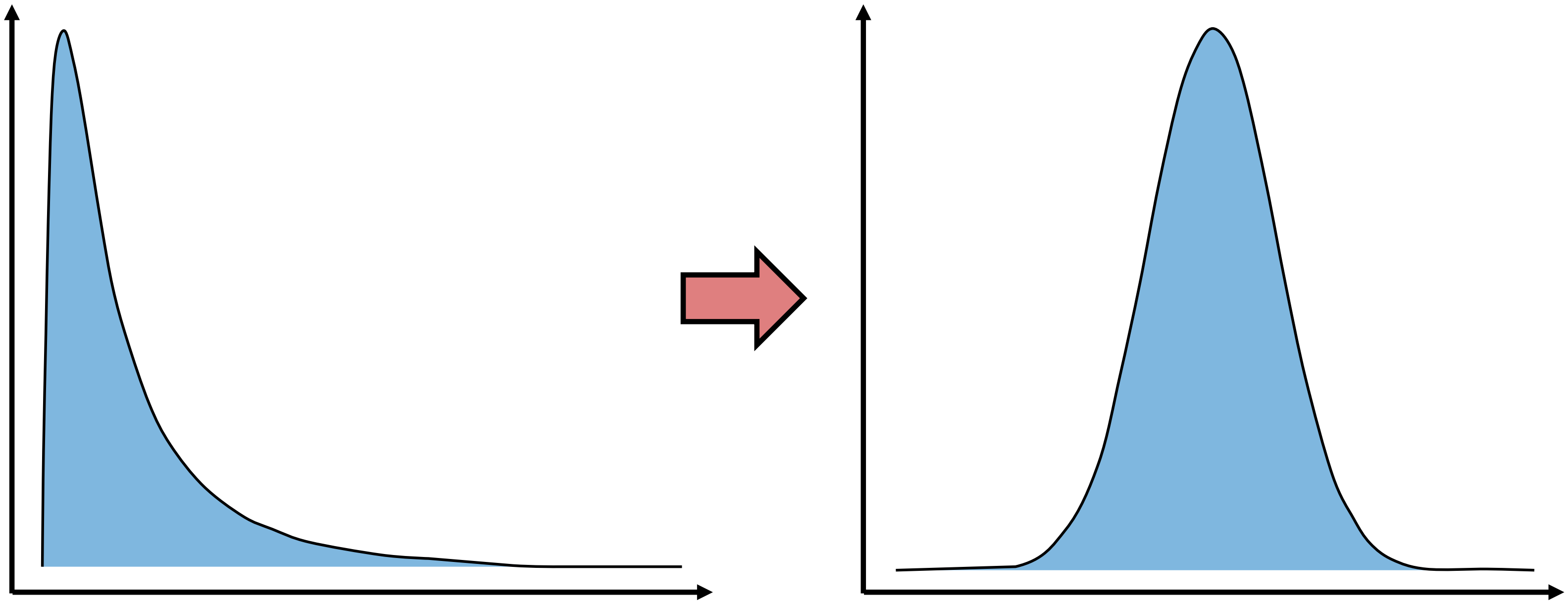


Transformation of Data Distributions

- Predictions from linear regression models assume residuals are normally distributed
- Features and predicted data are often skewed



Transformation of Data Distributions

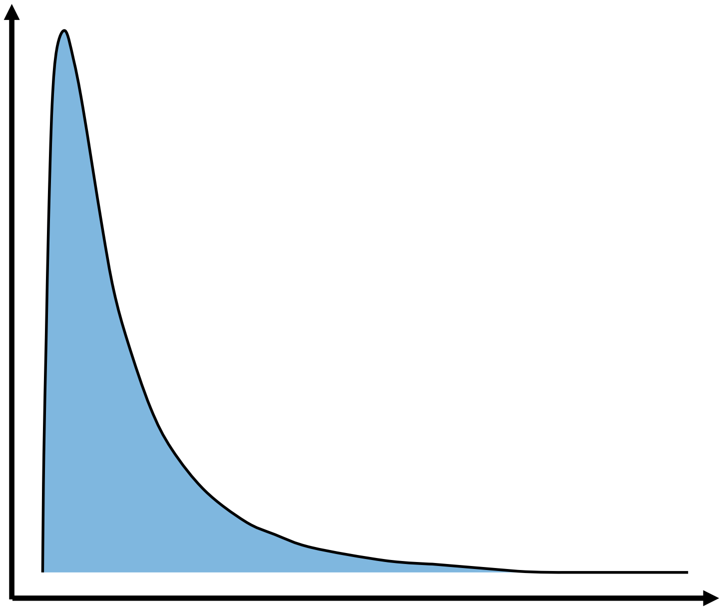


from numpy import log, log1p

from scipy.stats import boxcox

Transformation of Data Distributions

- Predictions from linear regression models assume residuals are normally distributed
- Features and predicted data are often skewed
- Data transformations can solve this issue



Feature Types

Feature Type	Transformation
• **Continuous:** numerical values	

Feature Types

Feature Type	Transformation
• **Continuous:** numerical values	• Standard Scaling, Min-Max Scaling

Feature Types

Feature Type	Transformation
• **Continuous:** numerical values	• Standard Scaling, Min-Max Scaling
• **Nominal:** categorical, unordered features (*True* or *False*)	• One-hot encoding (0, 1)

from sklearn.preprocessing import LabelEncoder, LabelBinarizer, OneHotEncoder

Feature Types

Feature Type	Transformation
• **Continuous:** numerical values	• Standard Scaling, Min-Max Scaling
• **Nominal:** categorical, unordered features (*True* or *False*)	• One-hot encoding (0, 1)
• **Ordinal:** categorical, ordered features (movie ratings)	• Ordinal encoding (0, 1, 2, 3)

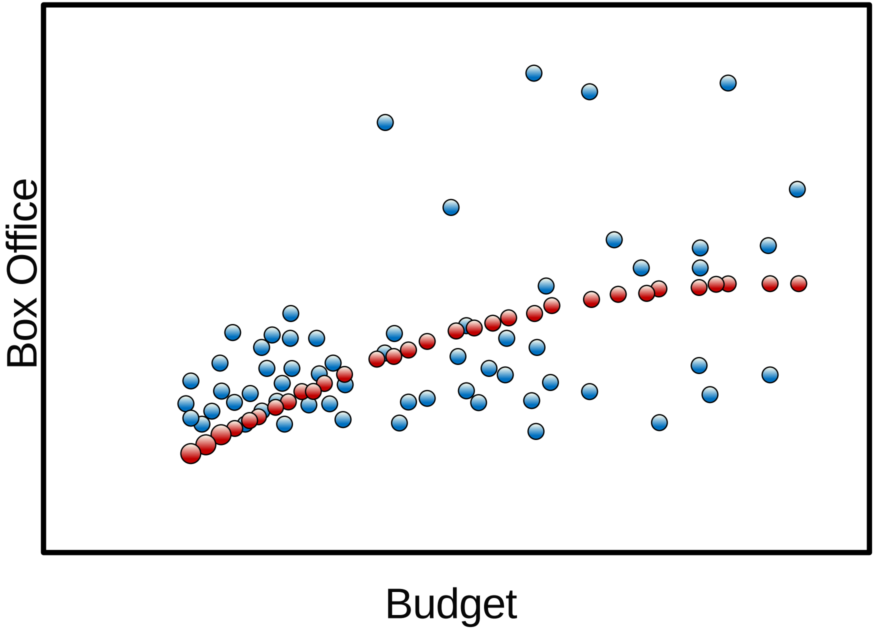
from sklearn.feature_extraction import DictVectorizer

from pandas import get_dummies

Addition of Polynomial Features

$$y_\beta(x) = \beta_0 + \beta_1 x + \beta_2 x^2$$

- Capture higher order features of data by adding polynomial features

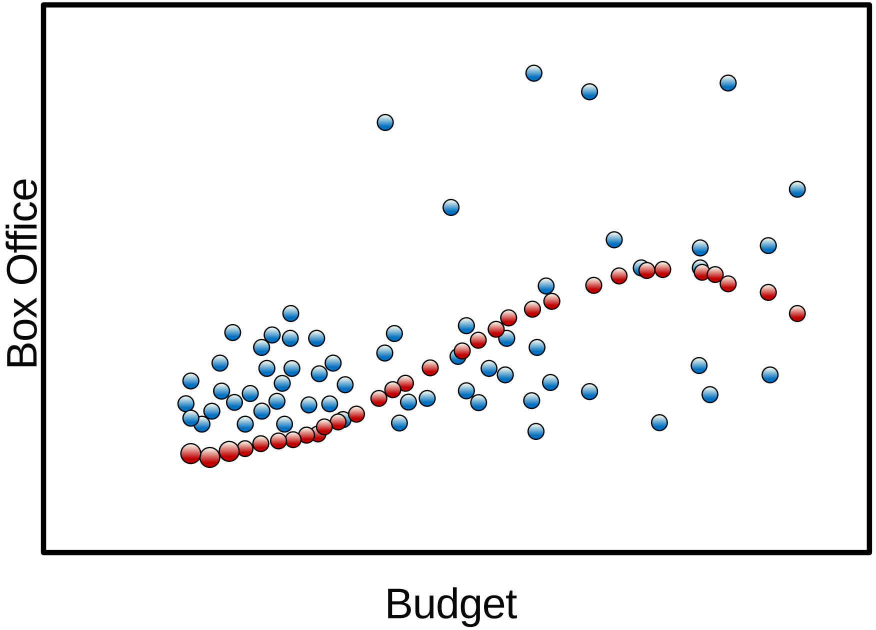


Addition of Polynomial Features

$$y_\beta(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

- Capture higher order features of data by adding polynomial features

- "Linear regression" means linear combinations of features

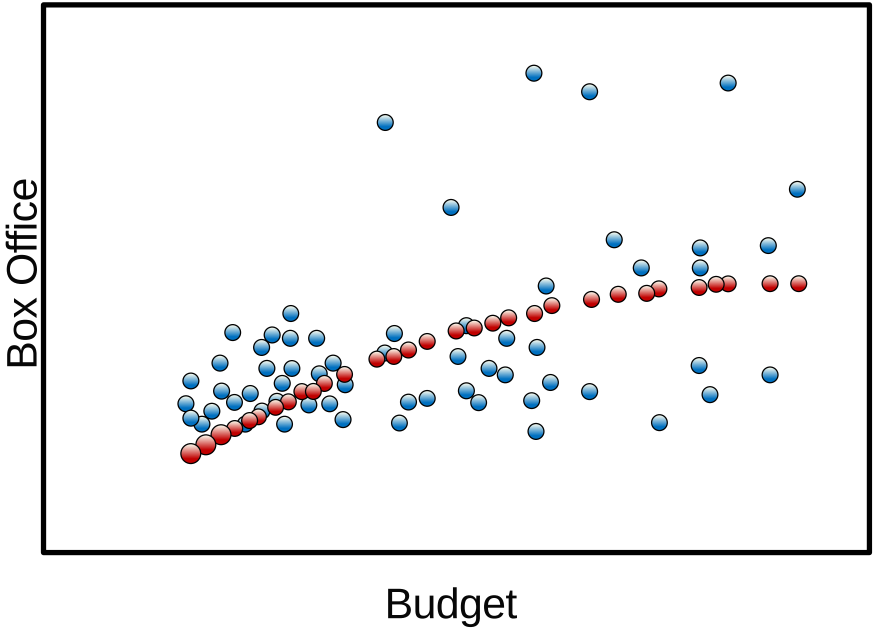


Addition of Polynomial Features

$$y_\beta(x) = \beta_0 + \beta_1 x + \beta_2 x^2$$

- Capture higher order features of data by adding polynomial features

- "Linear regression" means linear combinations of features

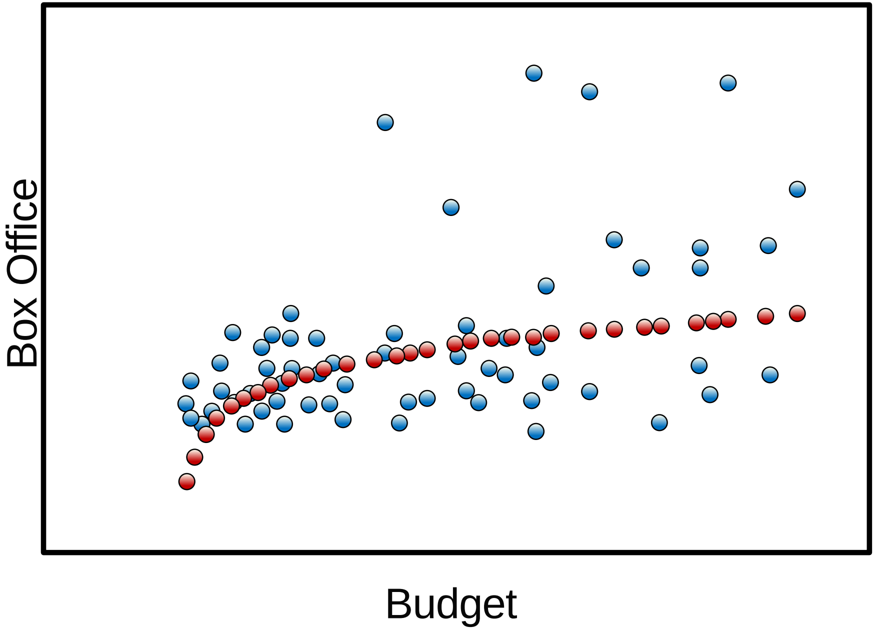


Addition of Polynomial Features

$$y_\beta(x) = \beta_0 + \beta_1 \log(x)$$

- Capture higher order features of data by adding polynomial features

- "Linear regression" means linear combinations of features



Addition of Polynomial Features

- Can also include variable interactions

$$y_\beta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$$

Addition of Polynomial Features

- Can also include variable interactions

$$y_\beta(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$$

- How is the correct functional form chosen? $\Rightarrow$ Check relationship of each variable or with outcome

Polynomial Features: The Syntax

Import the class containing the transformation method

```
from sklearn.preprocessing import PolynomialFeatures
```

Polynomial Features: The Syntax

Import the class containing the transformation method

```
from sklearn.preprocessing import PolynomialFeatures
```

Create an instance of the class

```
polyFeat = PolynomialFeatures(degree=2)
```

Polynomial Features: The Syntax

Import the class containing the transformation method

```
from sklearn.preprocessing import PolynomialFeatures
```

Create an instance of the class

```
polyFeat = PolynomialFeatures(degree=2)
```

Create the polynomial features and then transform the data

```
polyFeat = polyFeat.fit(X_data)
X_poly = polyFeat.transform(X_data)
```

Legal Notices and Disclaimers

This presentation is for informational purposes only. INTEL MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. Check with your system manufacturer or retailer or learn more at intel.com.

This sample source code is released under the Intel Sample Source Code License Agreement.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2017, Intel Corporation. All rights reserved.