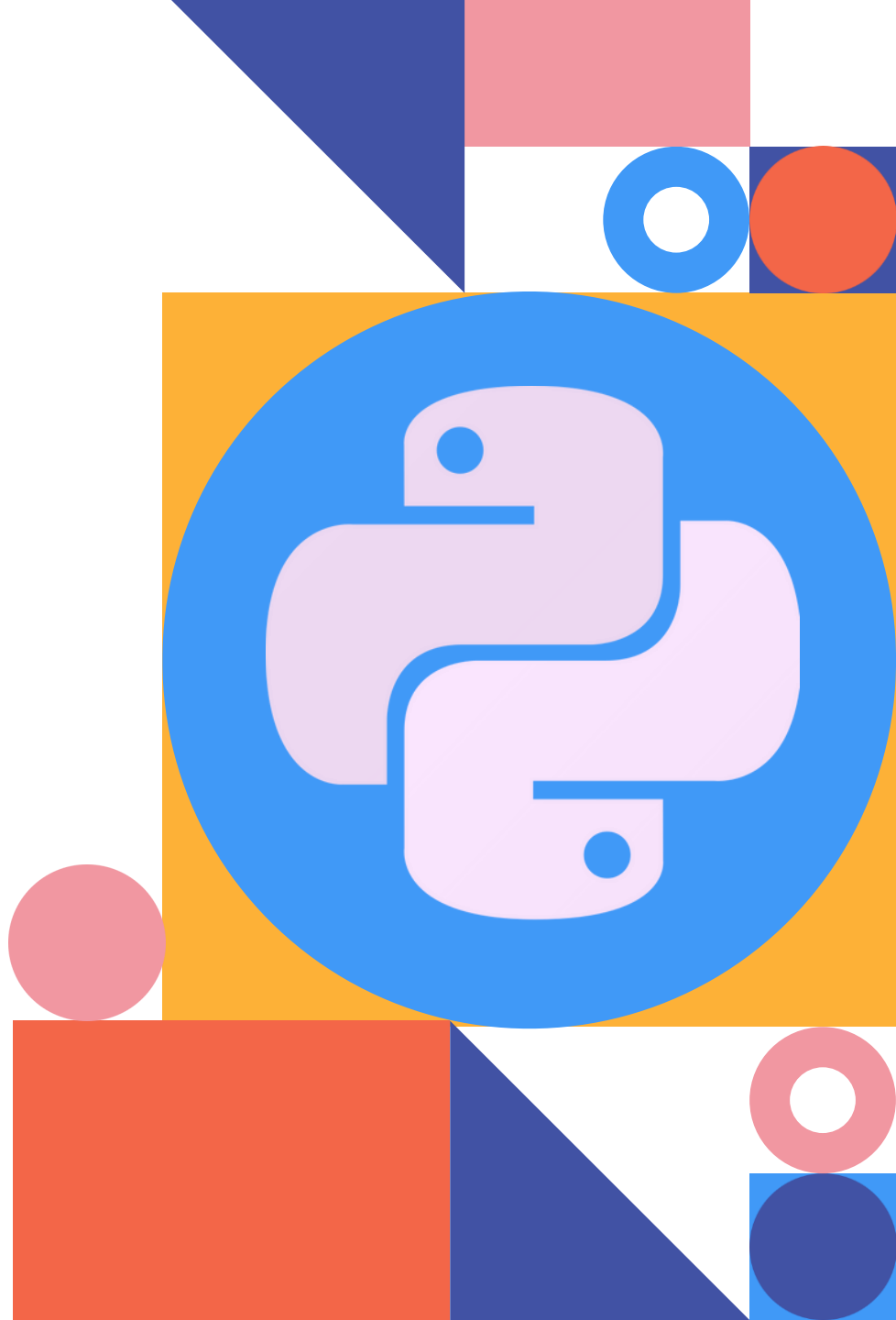
Machine Learning

Yan-Fu Kuo

Dept. of Biomechatronics Engineering
National Taiwan University



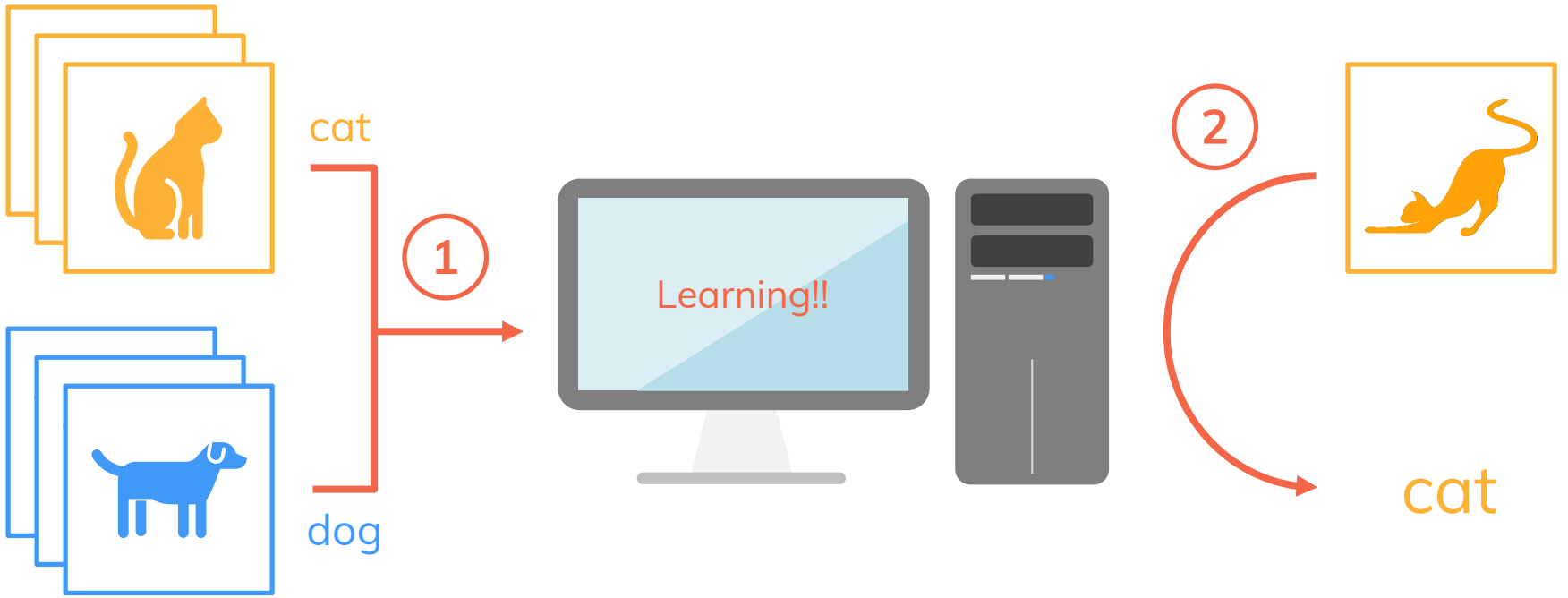
→

Contents

01 Introduction to ML

What is Machine Learning?

Getting computers to program themselves – let the computer learns from the data instead!

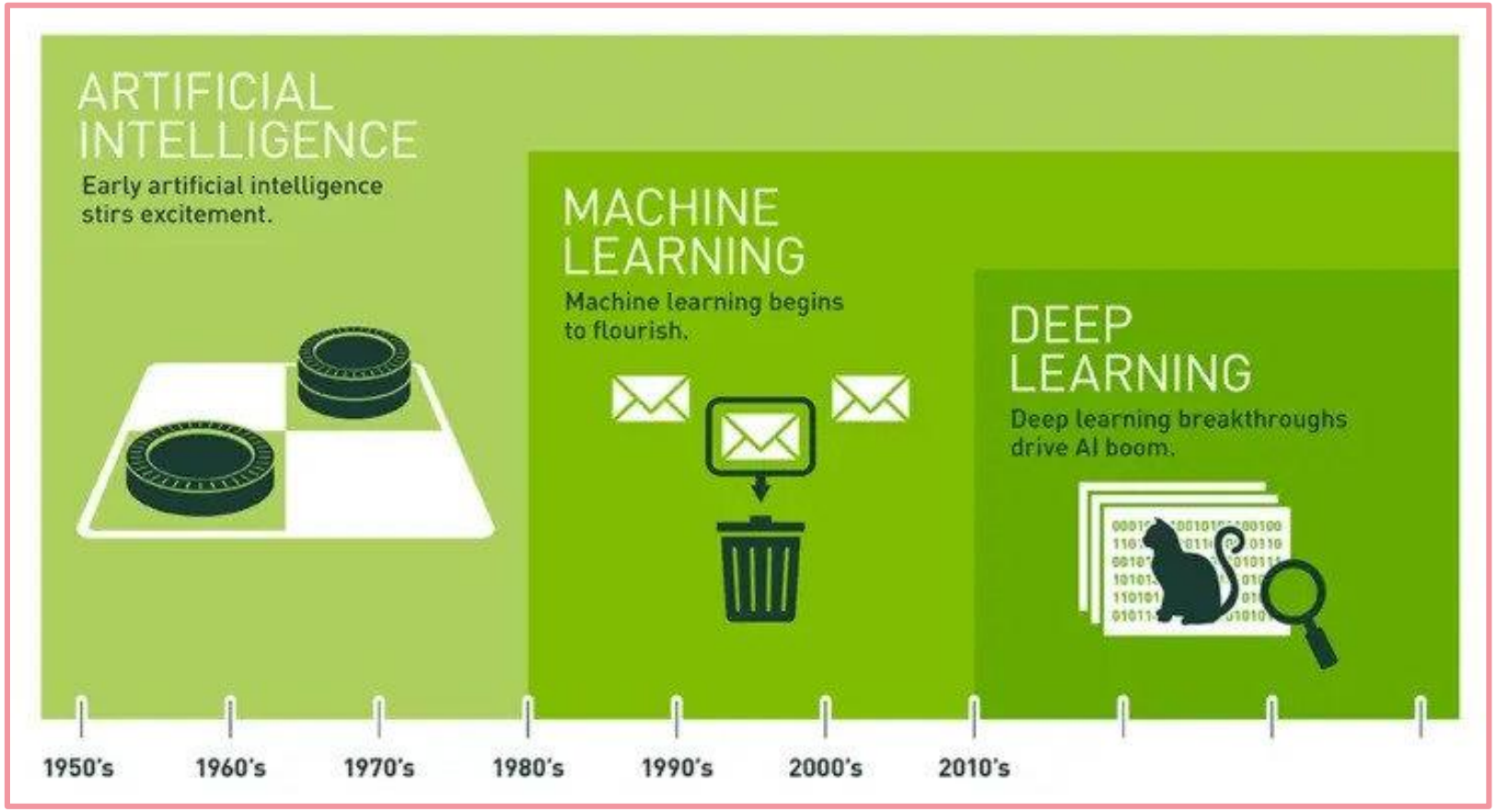


What is Machine Learning?

- **Traditional Programming**

- **Machine Learning**

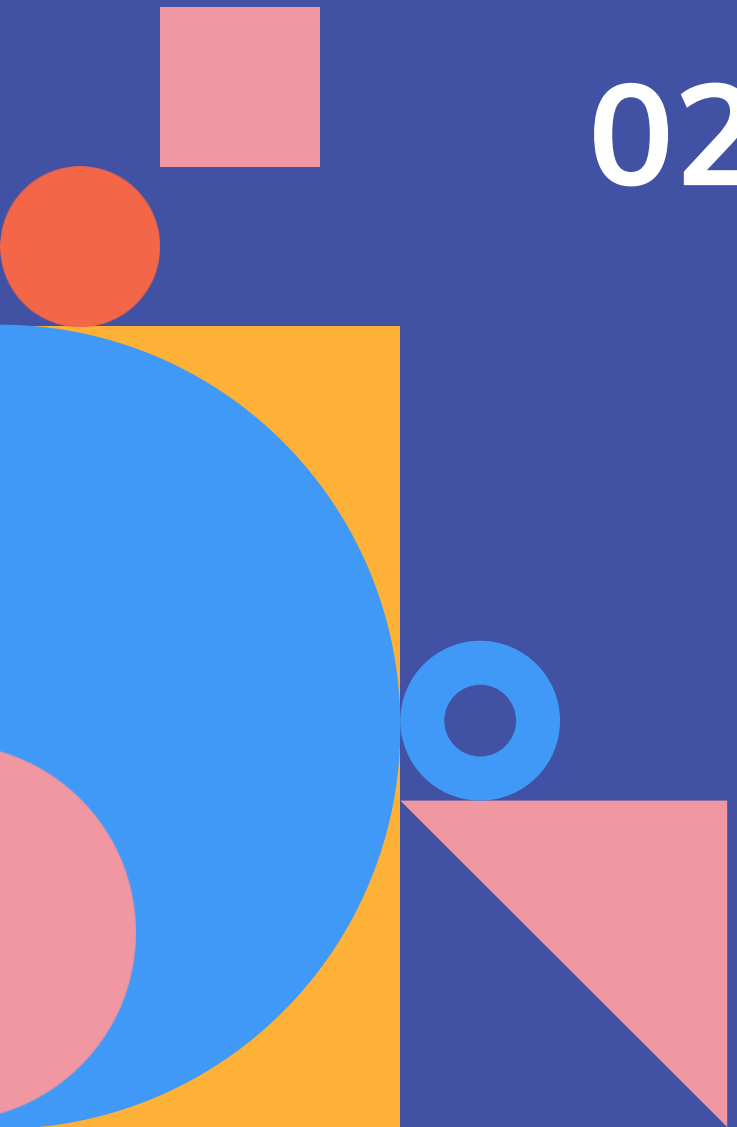
Difference between AI, ML,& DL



https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

Types of Learning Task

- **Supervised Learning**
  - Classification
    - SVM
    - KNN
    - Decision Tree
  - Regression
    - Linear Regression
    - SVR
- **Unsupervised Learning**
  - Clustering
    - SVD
    - K-means
  - Dimensionality Reduction
    - PCA



02 Mathematics behind ML

Example Dataset – House Price

	RM	LSTAT	PTRATIO	MEDV
0	6.575	4.98	15.3	504000
1	6.421	9.14	17.8	453600
2	7.185	4.03	17.8	728700
3	6.998	2.94	18.7	701400
4	7.147	5.33	18.7	760200

Features

RM: Average number of rooms per dwelling

LSTAT: % lower status of the population

PTRATIO: Pupil-teacher ratio by town

Target

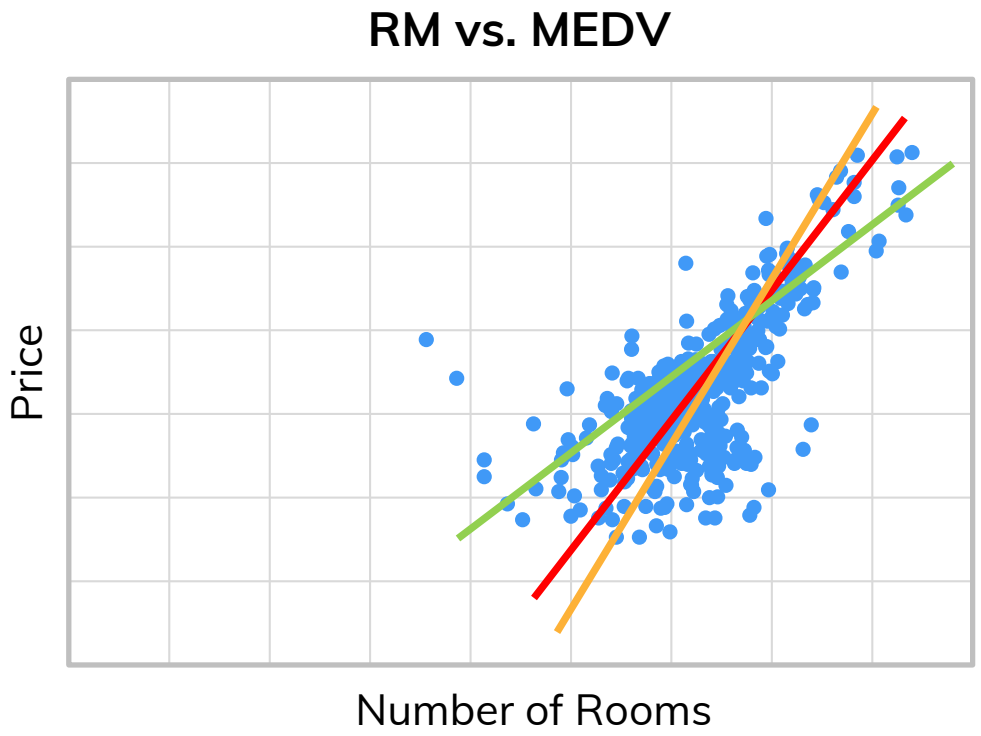
MEDV: Median value of owner-occupied homes

Simple Linear Regression

Can we predict the price through the numbers of rooms?

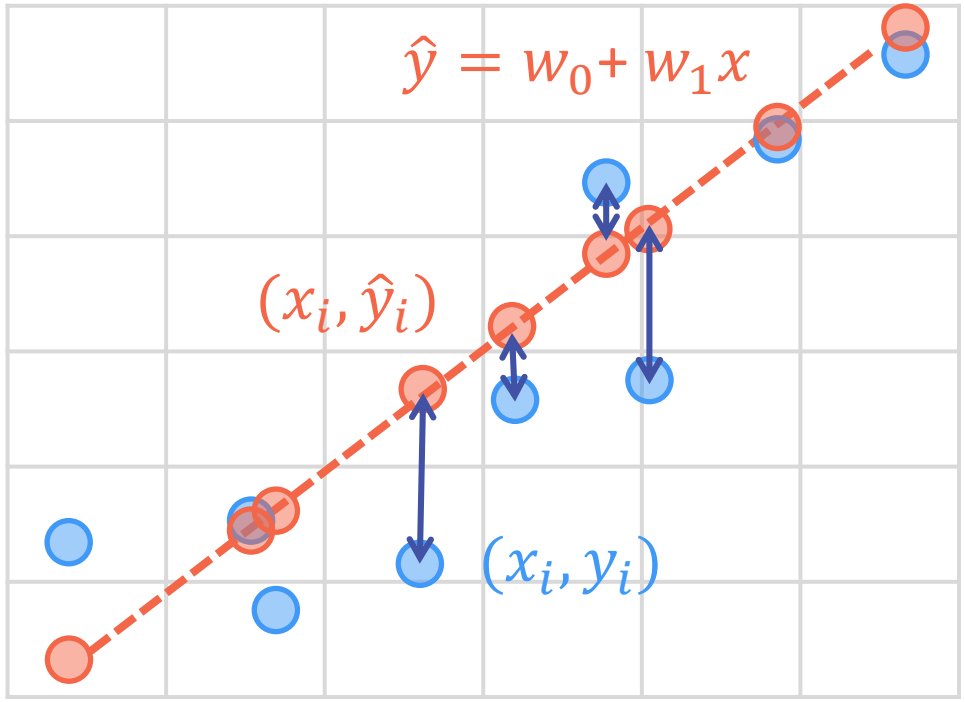
How does the number of rooms affects the price?

Answer: through optimization



Loss Function

A loss function is a function that computes the distance between the current output of the algorithm and the expected output.



$$loss(y, \hat{y}) = \frac{1}{N}\sum_i (y_i - \hat{y_i})^2$$

$$= \frac{1}{N}\sum_i {\epsilon_i}^2$$

$$= \frac{1}{N}\sum_i (y_i - w_o - w_1 x_i)^2$$

Mean Squared Error

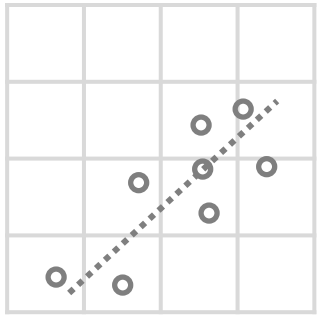
$$\min_{w_0,w_1} \; loss(y,\hat{y}) = MSE(w_o, w_1) = \frac{1}{N}\sum_{i=0}^{N}(y_i - w_o - w_1 x)^2$$

$$\begin{cases} \dfrac{\partial MSE}{\partial w_0} = \dfrac{-2}{N}\displaystyle\sum_{i=0}^{N}(y_i - w_o - w_1 x_i) = 0 \\ \dfrac{\partial MSE}{\partial w_1} = \dfrac{-2}{N}\displaystyle\sum_{i=0}^{N}(y_i - w_o - w_1 x_i)x_i = 0 \end{cases}$$

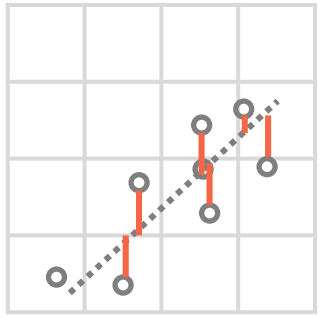
➡ Optimization

Review: Procedure of ML

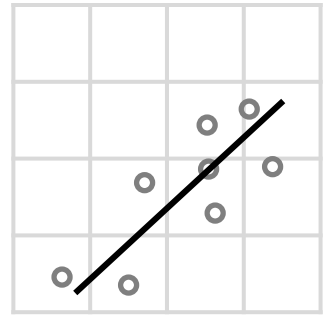
Define a Model



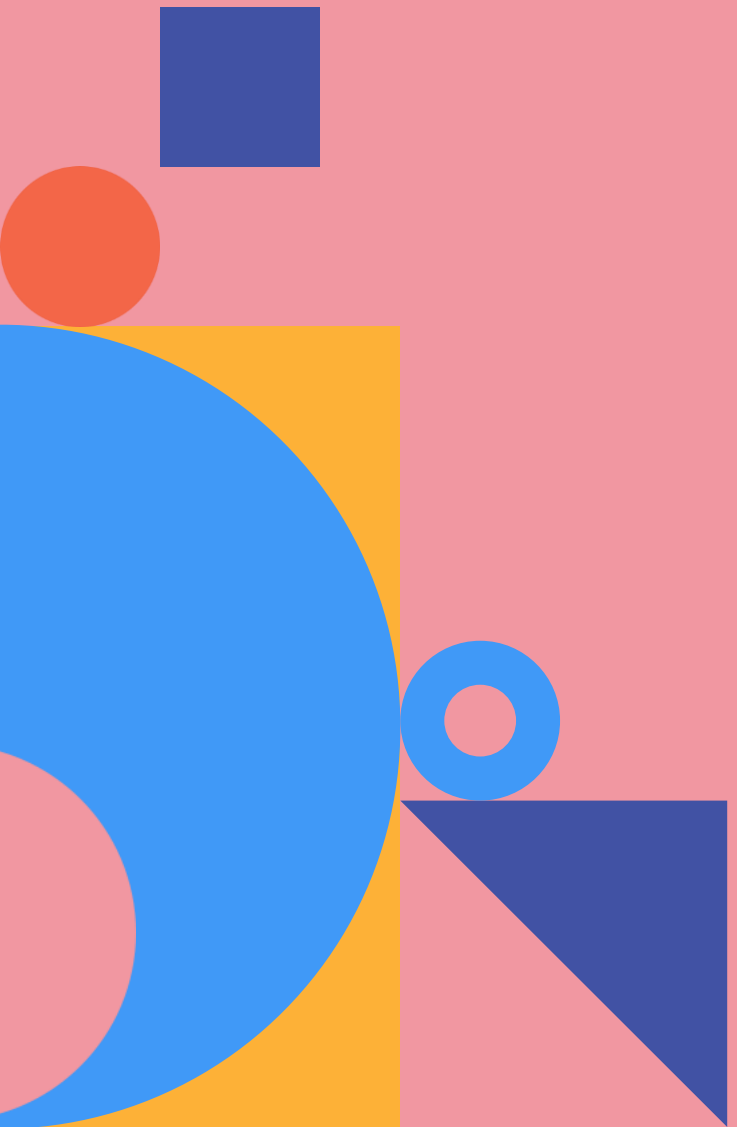
Define the Loss Function



Find the optimal solution



流水號	授課對象	課號	班次	課程名稱 查看課程大綱，請點選課程名稱	領域專長	學分	課程識別碼	全/半年	必/選修	授課教師	加選方式	時間 教室	總人數	選課限制條件	備註
79369	數學系	MATH5246		訊號處理和機器學習之數學基礎		3.0	221 U8820	半年	選修	黃文良	3	四7,8,9(天數302)	40	本校修課人數上限：40人	
79369	數學所	MATH5246		訊號處理和機器學習之數學基礎		3.0	221 U8820	半年	選修	黃文良	3	四7,8,9(天數302)	40	本校修課人數上限：40人	
79369	應數所	MATH5246		訊號處理和機器學習之數學基礎		3.0	221 U8820	半年	選修	黃文良	3	四7,8,9(天數302)	40	本校修課人數上限：40人	
32409	地理所	Geog7126		資料科學與機器學習入門		3.0	228EM9690	半年	選修	亞歷山卓	2	二7,8,9(地理二教室)	25(含開放臺大系統人數:4)	本校修課人數上限：21人,外系人數限制：3人	本課程以英語授課。本課程英文化程度說明
19185	醫材影像所	MDI7071		生醫機器學習專題研究I		2.0	458 M0710	半年	選修	張瀚	2	三9,10(基醫1222)	15	本校修課人數上限：15人,外系人數限制：5人	初選限定:本所碩士班一年級學生
81530	醫材影像所	MDI7078		機器學習的生醫應用		3.0	458EM0780	半年	必修	張瀚	2	三6,7,8(基醫1222) 異動	15	限本系所學生(含輔系、雙修生),本校修課人數上限：15人	本課程以英語授課。甲群組(資料處理領域):機器學習的生醫應用/醫學影像處理/數位生醫訊號處理(3科目中必選1科目)本課程英文化程度說明
81530	醫材影像所	MDI7078		機器學習的生醫應用		3.0	458EM0780	半年	選修	張瀚	2	三6,7,8(基醫1222) 異動	15	限本系所學生(含輔系、雙修生),本校修課人數上限：15人	本課程以英語授課。本課程英文化程度說明
81530	精準健康博士	MDI7078		機器學習的生醫應用		3.0	458EM0780	半年	必修	張瀚	2	三6,7,8(基醫1222) 異動	15	限本系所學生(含輔系、雙修生),本校修課人數上限：15人	本課程以英語授課。甲群組(資料處理領域):機器學習的生醫應用/醫學影像處理/數位生醫訊號處理(3科目中必選1科目)本課程英文化程度說明
89083	土木所	CIE5133		機器學習與深度學習導論	智慧人居環境	3.0	521 U9230	半年	選修	陳俊杉	2	三2,3,4(新505)	80(含開放臺大系統人數:4)	本校修課人數上限：76人	第一堂課以抽籤方式決定修課名單。
89083	土木所CAE組	CIE5133		機器學習與深度學習導論	智慧人居環境	3.0	521 U9230	半年	選修	陳俊杉	2	三2,3,4(新505)	80(含開放臺大系統人數:4)	本校修課人數上限：76人	第一堂課以抽籤方式決定修課名單。
89083	土木系	CIE5133		機器學習與深度學習導論	智慧人居環境	3.0	521 U9230	半年	選修	陳俊杉	2	三2,3,4(新505)	80(含開放臺大系統人數:4)	本校修課人數上限：76人	土木學群選修。第一堂課以抽籤方式決定修課名單。
77976	老人長照學程	DBME5027		機器學習		3.0	528 U0150	半年	選修	陳中明	3	四2,3,4(共406)	40	限學士班三年級以上,本校修課人數上限：40人,外系人數限制：1人	本課程中文授課,使用英文教科書。
77976	醫工所	DBME5027		機器學習		3.0	528 U0150	半年	選修	陳中明	3	四2,3,4(共406)	40	限學士班三年級以上,本校修課人數上限：40人,外系人數限制：1人	本課程中文授課,使用英文教科書。



03 How to Train a Model

Machine Learning in Python

scikit-learn

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on the top of NumPy, SciPy, and matplotlib
- Open source, commercially usable – BSD license

```
$ pip install scikit-learn
```

Basics of the API

- Step 1 | Choose a class of model
- Step 2 | Choose model hyperparameters

Strong Mathematical Foundation

- Step 3 | Arrange data into a features matrix and target vector
- Step 4 | Fit the model to your data by calling the fit() method **(training)**
- Step 5 | Apply the model to new data by using predict() or transform() method **(predict)**

STEP0

Data Preparation

```
import pandas as pd

data = pd.read_csv('housing.csv')
```

	RM	LSTAT	PTRATIO	MEDV
0	6.575	4.98	15.3	504000
1	6.421	9.14	17.8	453600
2	7.185	4.03	17.8	728700
3	6.998	2.94	18.7	701400
4	7.147	5.33	18.7	760200

Features

RM: Average number of rooms per dwelling

LSTAT: % lower status of the population

PTRATIO: Pupil-teacher ratio by town

Target

MEDV: Median value of owner-occupied homes

STEP0

Data Preparation (build-in datasets)

```
from sklearn import datasets
```

datasets.load_iris()

Classes	3
Samples per class	50
Sample total	150
Dimensionality	4
Features	real, positive

datasets.load_boston()

Samples total	506
Dimensionality	13
Features	real, positive
Targets	Real 5. -50.

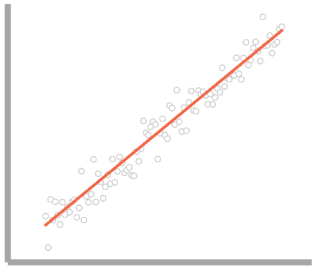
datasets.load_digits()

Classes	10
Samples per class	~180
Sample total	1797
Dimensionality	64
Features	Integers 0-16

STEP1

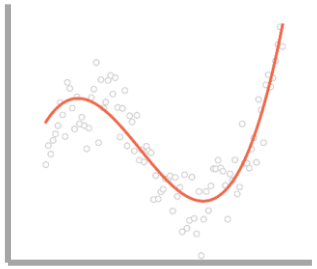
Choose a Class of Model

Linear



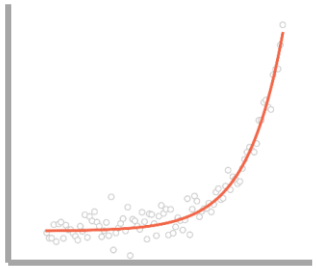
$y = ax + b$

Polynomial



$y = ax^2 + bx + \text{c}$

Exponential



$y = a(e^{bx})$

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

https://scikit-learn.org/stable/modules/linear_model.html

STEP2

Choosing Model Hyperparameters

Depending on the model class we are working with, we might need to answer one or more questions like the following:

`(fit_intercept:bool, default=True)`

- Would we like to fit for the offset (i.e., y-intercept)?

`(normalize:bool, default=False)`

- Would we like the model to be normalized?

`(sklearn.preprocessing)`

- Would we like to preprocess our features to add model flexibility?
- What degree of regularization would we like to use in our model?
- How many model components would we like to use?

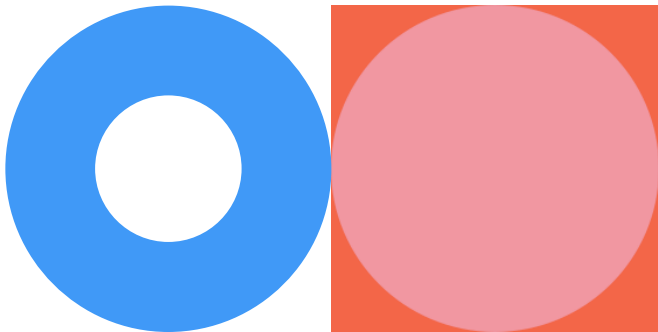
STEP2

Choosing Model Hyperparameters

In scikit-learn, hyperparameters are chosen by passing values at model instantiation.

```
model = LinearRegression(fit_intercept=True)
```

LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)



STEP3

Arrange Data

```
X = data['RM'].to_numpy()
y = data['MEDV'].to_numpy()
```

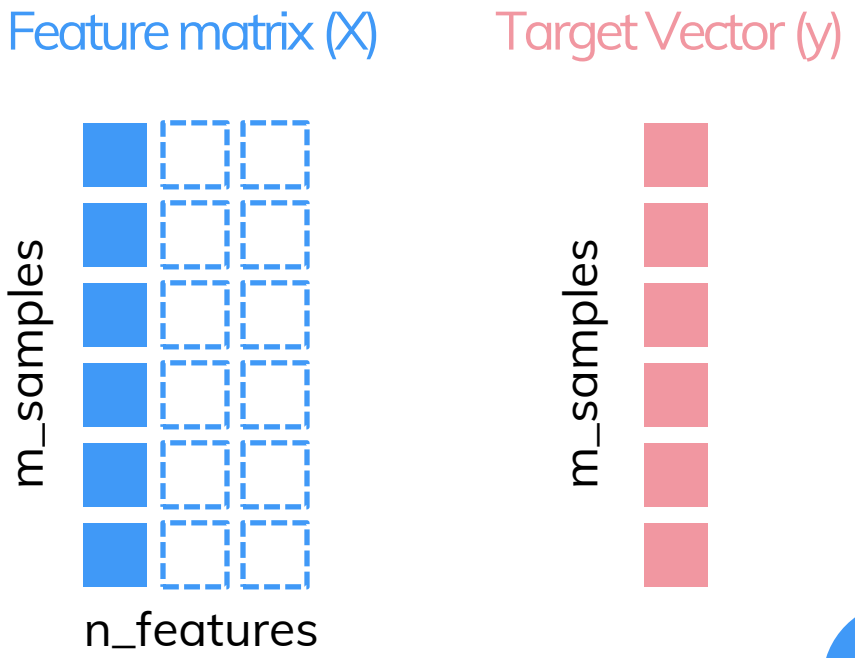
Making X a 2D array

```
X = np.expand_dims(X, axis=1)
```

or

```
X = X[:, np.newaxis]
X.shape
```

(498, 1)



STEP4

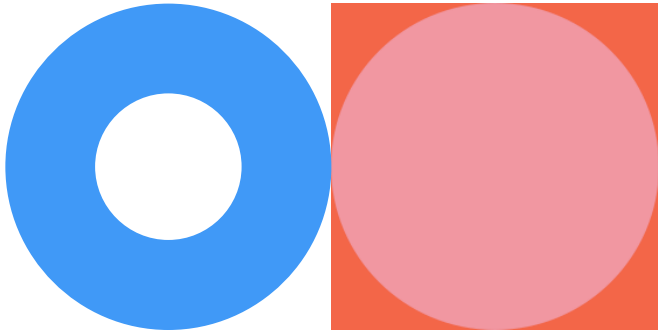
Fit the Model to Your Data

```
model.fit(X, y)
```

This `fit()` command causes a number of model-dependent internal computations to take place, and the results of these computations are stored in model-specific attributes that the user can explore.

```
model.coef_, model.intercept_
```

By convention, all model parameters that were learned during the fit() process have trailing underscores

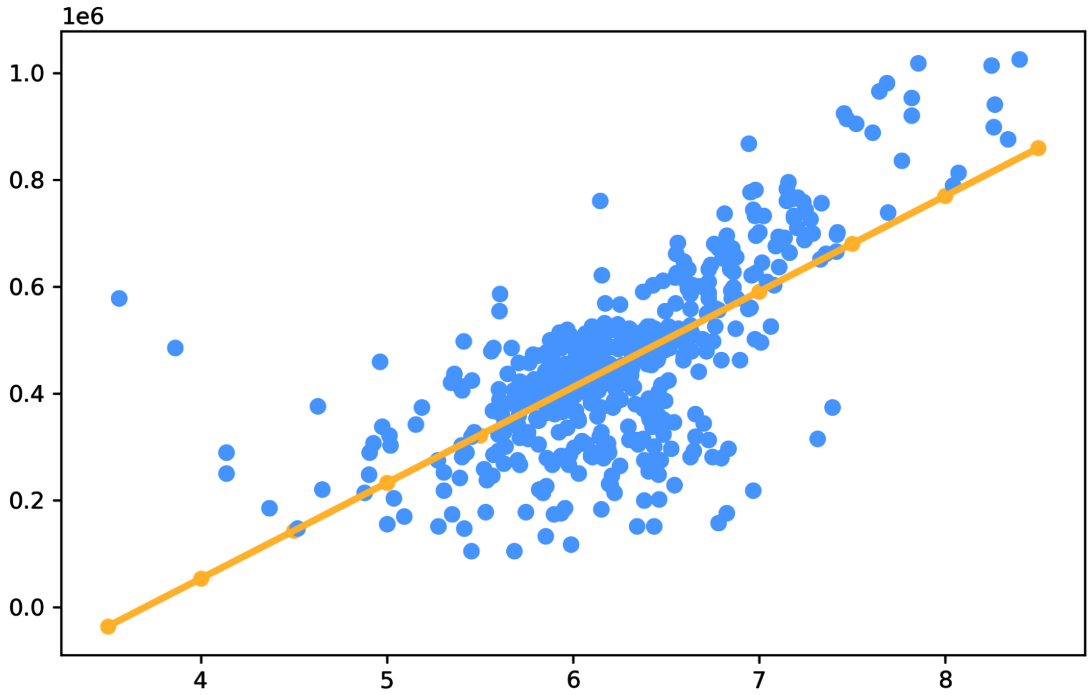


STEP5

Predict Labels for Unknown Data

```
yfit = model.predict(Xfit)
```

```
model.score(X, y)
```



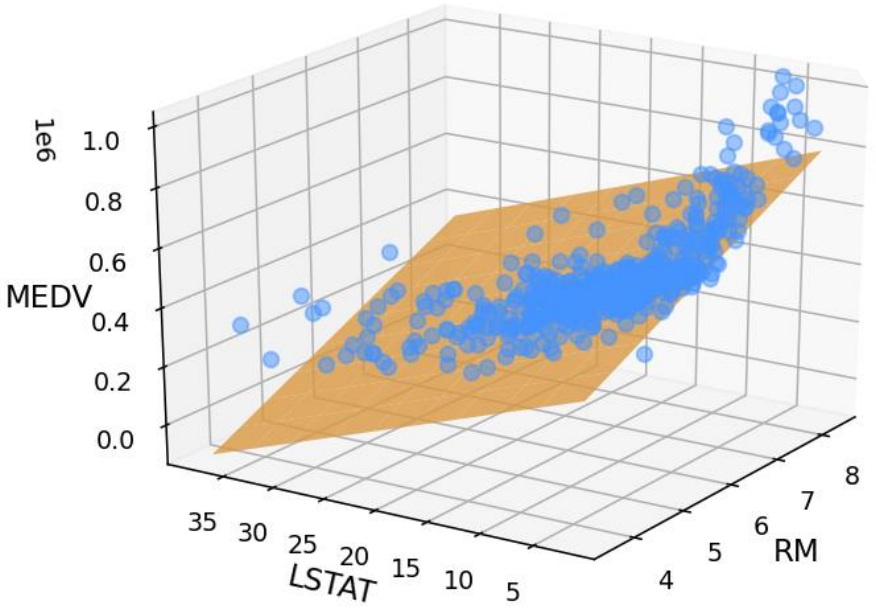
Multiple Regression

```
X = data.loc[:, 'RM':'LSTAT'].to_numpy()
y = data['MEDV'].to_numpy()
```

```
model.fit(X, y)
model.coef_, model.intercept_
```

(array([95148.09, -12466.44]), 21902.60)

w_1 w_2



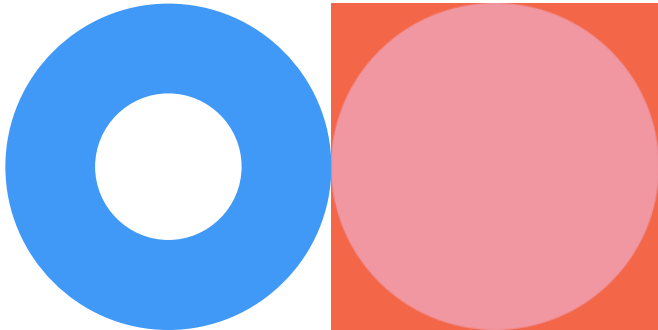
Exercise (10 mins)

Does the number of rooms predicts the price?

Increase the number of rooms increase the price?

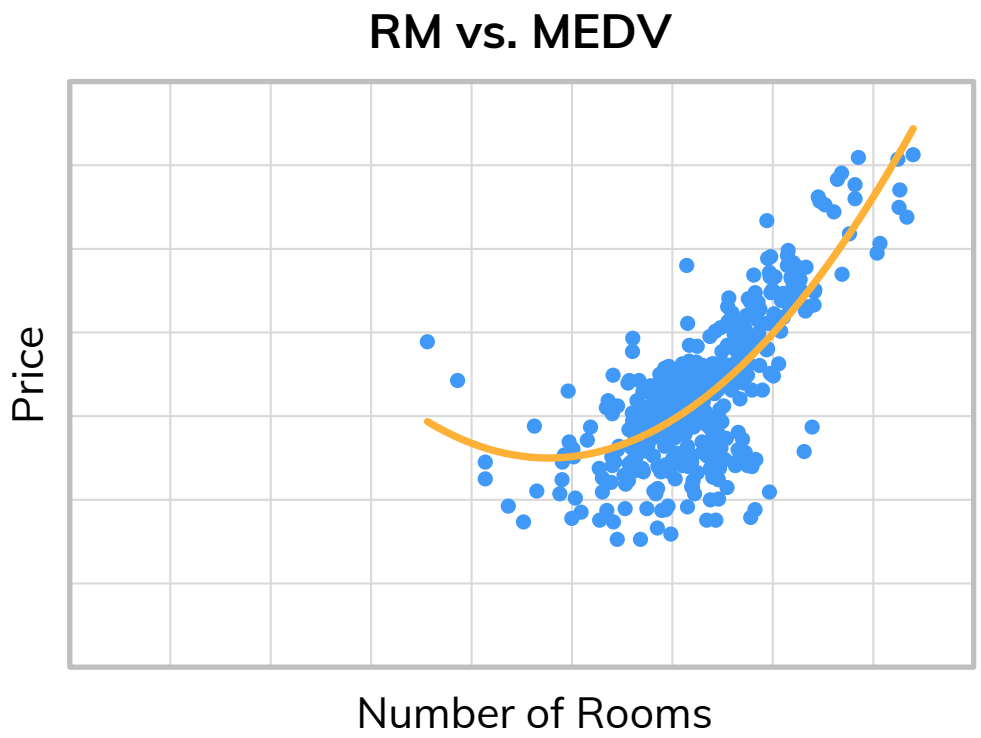
How does the number of rooms affects the price?

Predict the median price of the town with RM=6.976, LSTAT=5.64, and PTRATIO=21.



Polynomial Regression

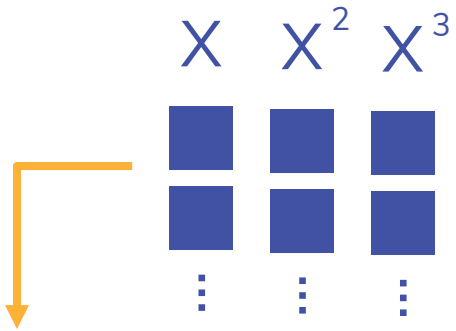
$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3 \ldots$$



Polynomial Regression


```
model = PolynomialRegression(degree=3)
```

$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$




```
model = LinearRegression()
```

$$y = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$

$$y = w_0 + w_1 x + w_2 \exp(x)$$

Polynomial Regression

```
from sklearn.preprocessing import PolynomialFeatures
quadratic = PolynomialFeatures(degree=2)
```

```
X_quad = quadratic.fit_transform(X)
```

or

```
quadratic.fit(X)
X_quad = quadratic.transform(X)
```

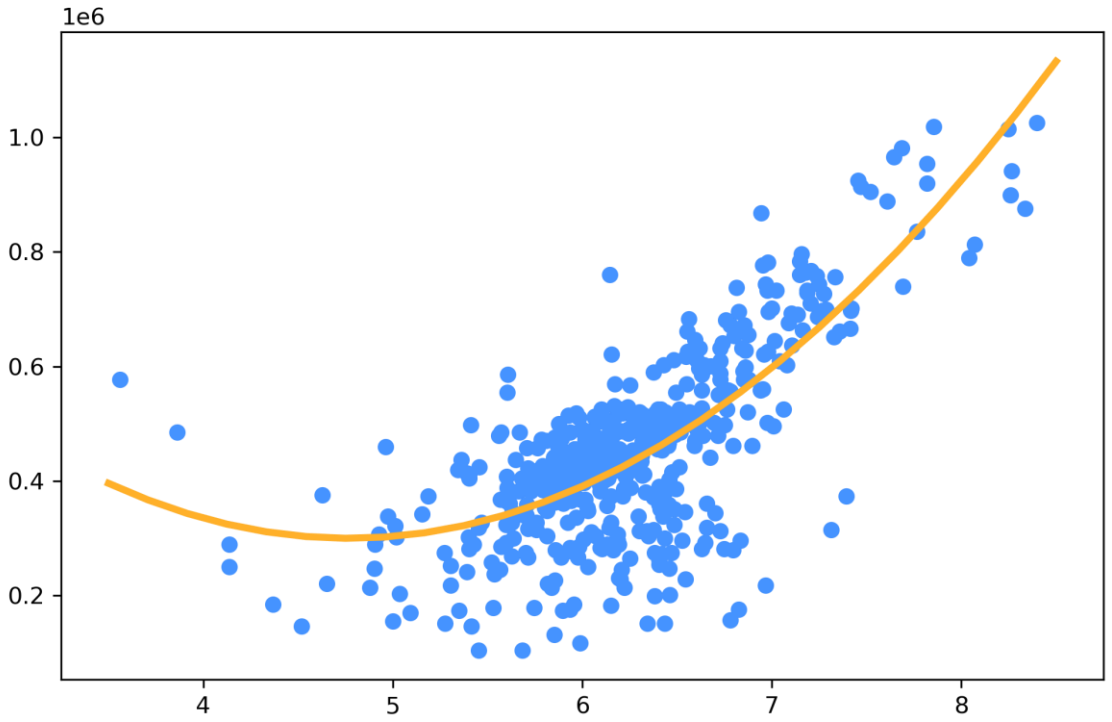
STEP2 Choosing Model Hyperparameters

- Would we like to preprocess our features to add model flexibility?

Polynomial Regression

```
model.fit(X, y)
```

```
model.score(X, y)
```



Model Validation

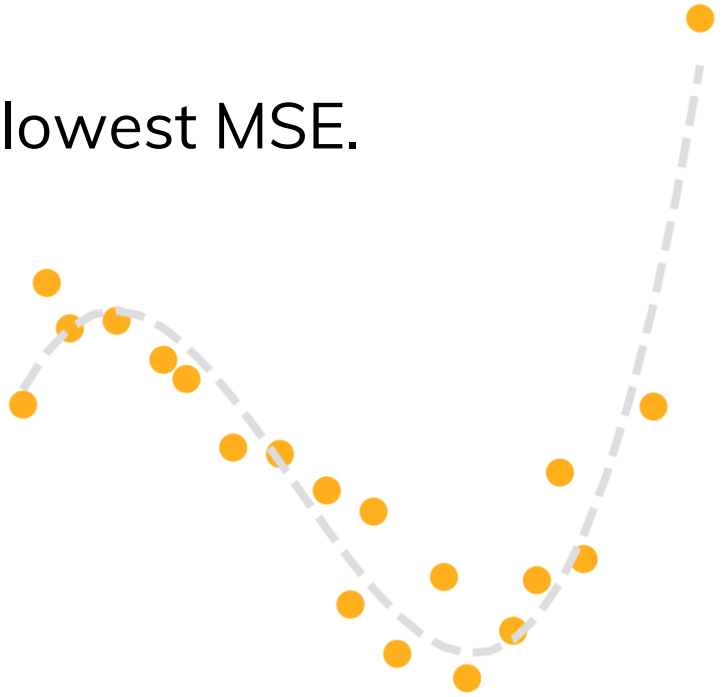
Model validation is to estimate how effective our trained model is by applying it to some of the training data and comparing the prediction to the known value

MSE for regression problems

```
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_train, y_pred)
```

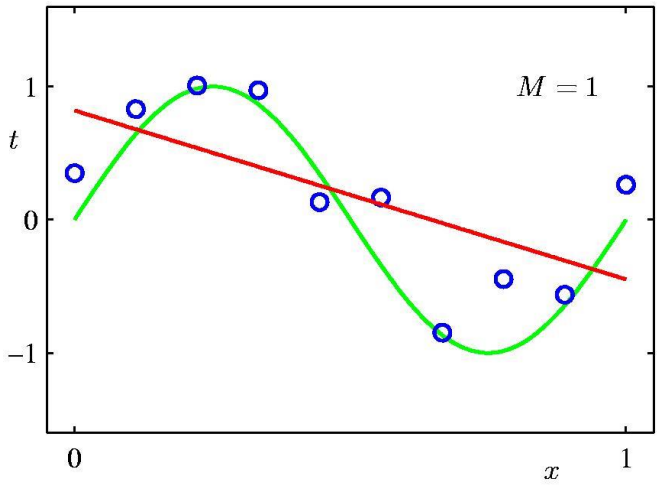
Exercise (15 mins)

1. Estimate the target function that generated the data in train.csv by using polynomial regression.
2. Find the order of the polynomial with the lowest MSE.

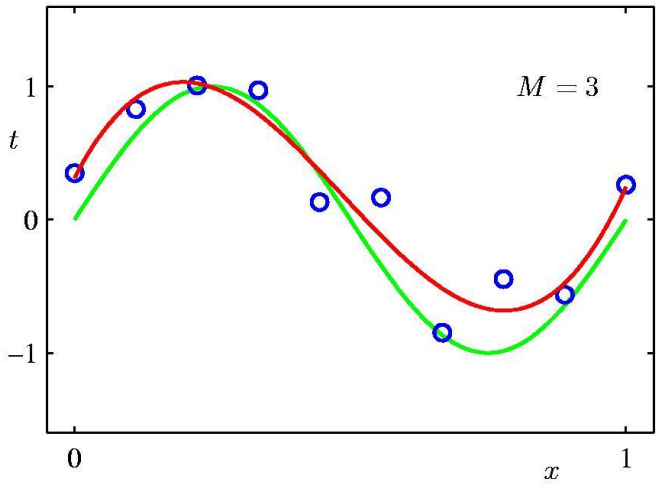


Order of the Polynomial Model?

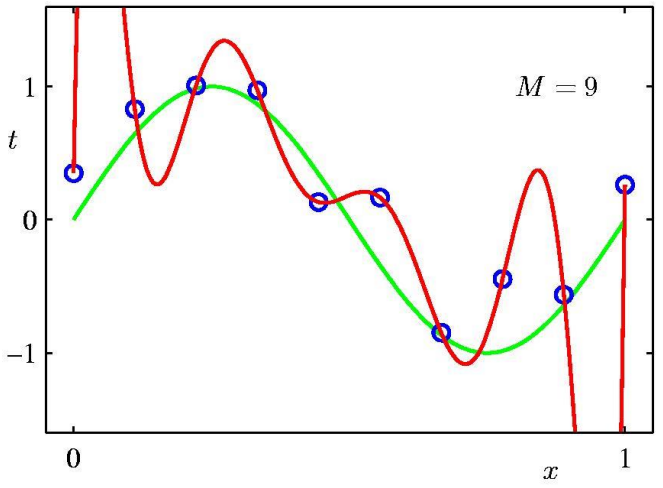
Data drawn from a sinusoidal model $\sin(\frac{x}{2\pi})$ with noise



Underfitting



Best fit



Overfitting

The Wrong Approach

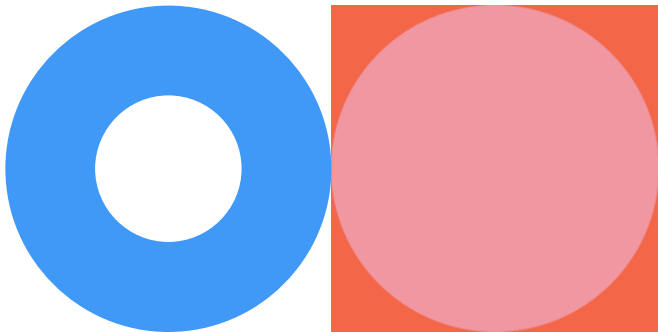
```
df = pd.read_csv('train.csv')
X_train = np.expand_dims(df['data'].to_numpy(), axis=1)
y_train = df['target'].to_numpy()
```

```
poly = PolynomialFeatures(degree=10)
X_train_poly = poly.fit_transform(X_train)
```

```
nd_model = LinearRegression(fit_intercept=True)
nd_model.fit(X_train_poly, y_train)
```

```
y_pred = nd_model.predict(X_train_poly)
mse = mean_squared_error(y_train, y_pred)
```

This model was trained and evaluated on the same data.

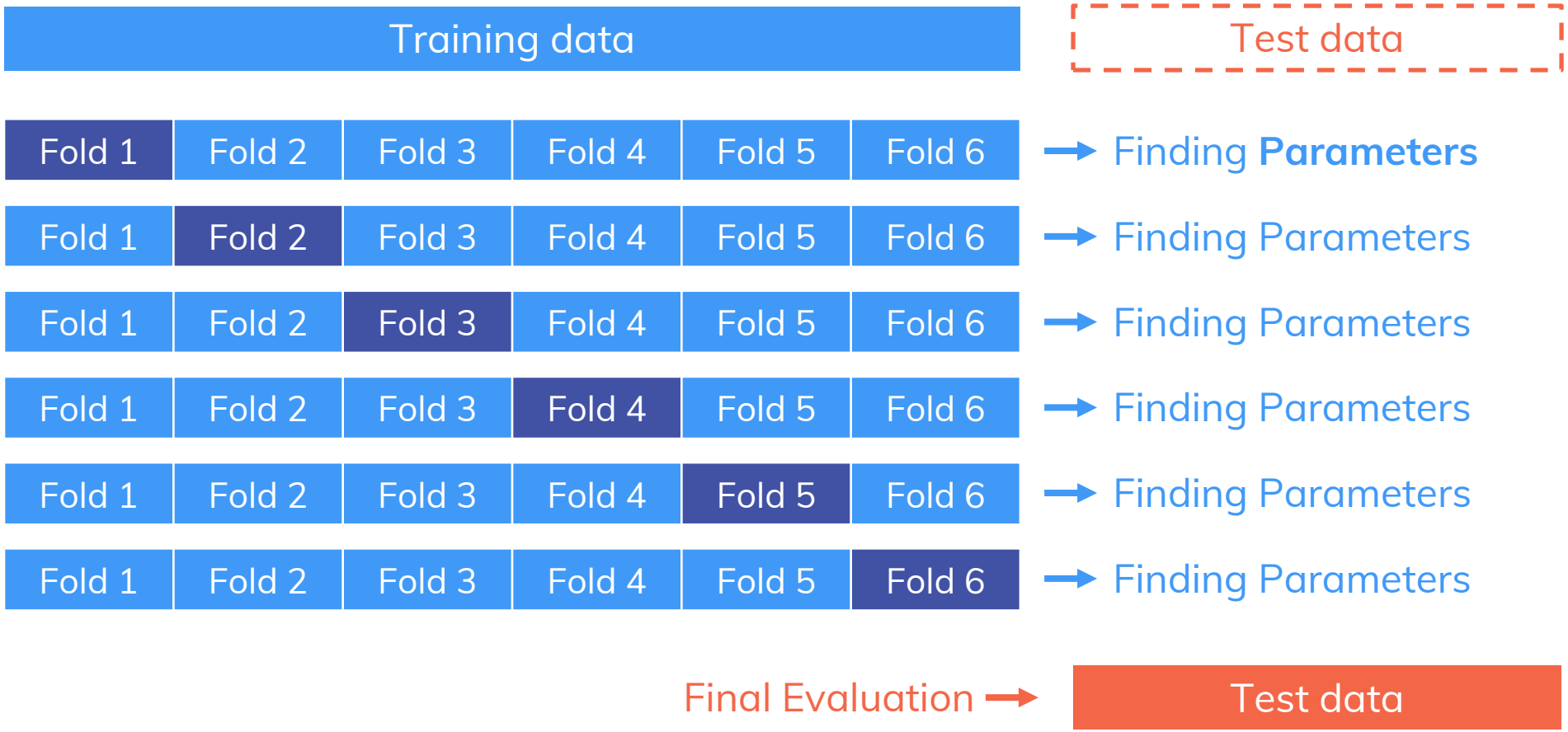


Splitting the Dataset

A better sense of a model's performance can be found using what's known as a validation set

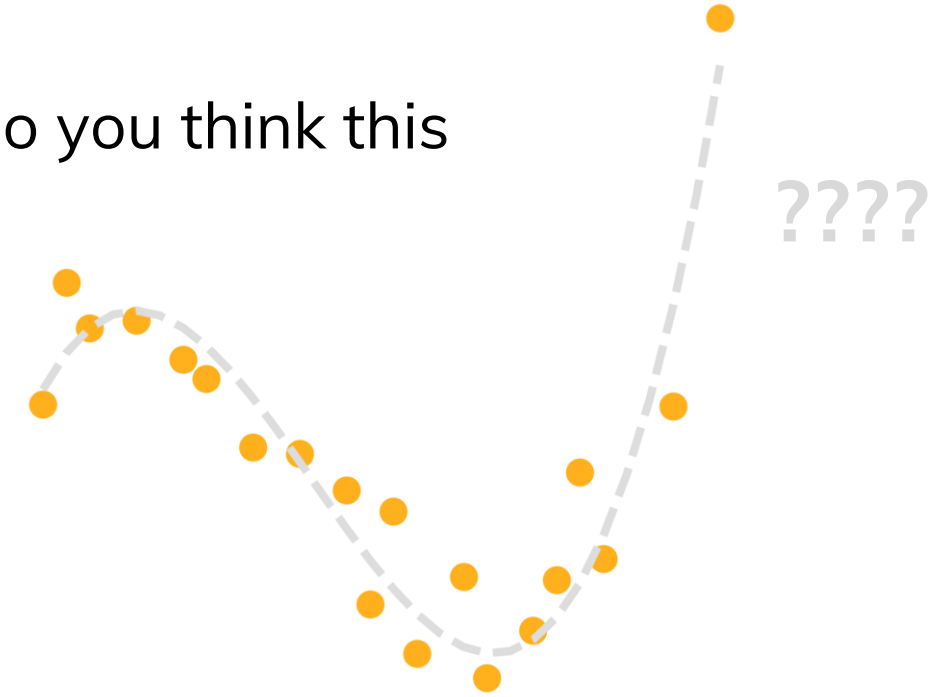

```
from sklearn.cross_validation import train_test_split
```

Cross-Validation (6-fold CV)



Exercise (15 mins)

1. Estimate the target function that generated the data in train.csv by using polynomial regression.
2. Find the function with the lowest MSE. Do you think this function is the target function?



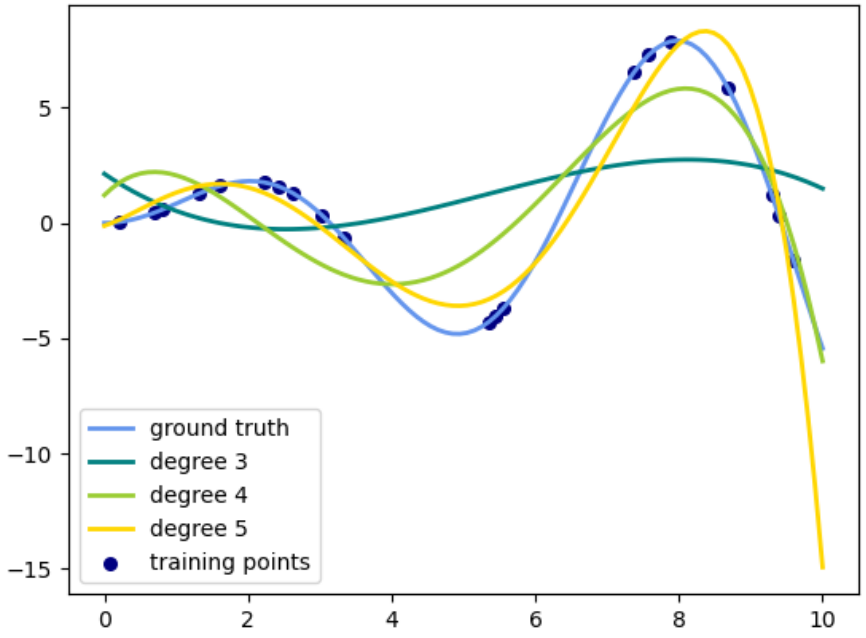
Model Regularization

- `Linear_model.LinearRegression`

  Loss: $\|Xw - y\|_2^2$

- `Linear_model.Ridge`

  Loss: $\|Xw - y\|_2^2 + \alpha\|w\|_2^2$



Encoding Categorical Features

```
from sklearn import preprocessing
```

		.OrdinalEncoder()	.OneHotEncoder()
pclass	‘upper’	1	[1, 0, 0]
	‘middle’	2	[0, 1, 0]
	‘lower’	3	[0, 0, 1]
sex	‘male’	1	[1, 0]
	‘female’	2	[0, 1]

Feature Scaling

```
from sklearn import preprocessing
```

Product A

Amount Per Serving	
Total Fat	10g
Protein	10g
Total Carbohydrate	6g
Vitamin C	600mg
Vitamin B1	19mg

Price **$40**

Product B

Amount Per Serving	
Total Fat	5g
Protein	10g
Total Carbohydrate	5g
Vitamin C	10mg
Vitamin B1	0mg

Price **$5**

`.MinMaxScaler()`

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

`.StandardScaler()`

$$x' = \frac{x - \bar{x}}{\sigma}$$

`.normalize()`

$$x' = \frac{x}{\|x\|}$$


thank you!