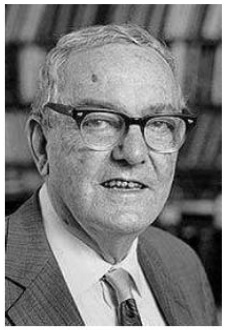
Engineering Analytics (ESE1007) Seminar 6

Machine Learning Linear Regression

Machine Learning

Herbert Alexander Simon:

- "Learning is any process by which a system improves performance from experience."
- "Machine Learning is concerned with computer programs that automatically improve their performance through experience."



Herbert Simon
Turing Award 1975
Nobel Prize in Economics 1978

Machine Learning Main Tasks

- Develop systems that can automatically adapt and customize themselves to individual users.
  - Personalized news or mail filter
- Discover new knowledge from large databases

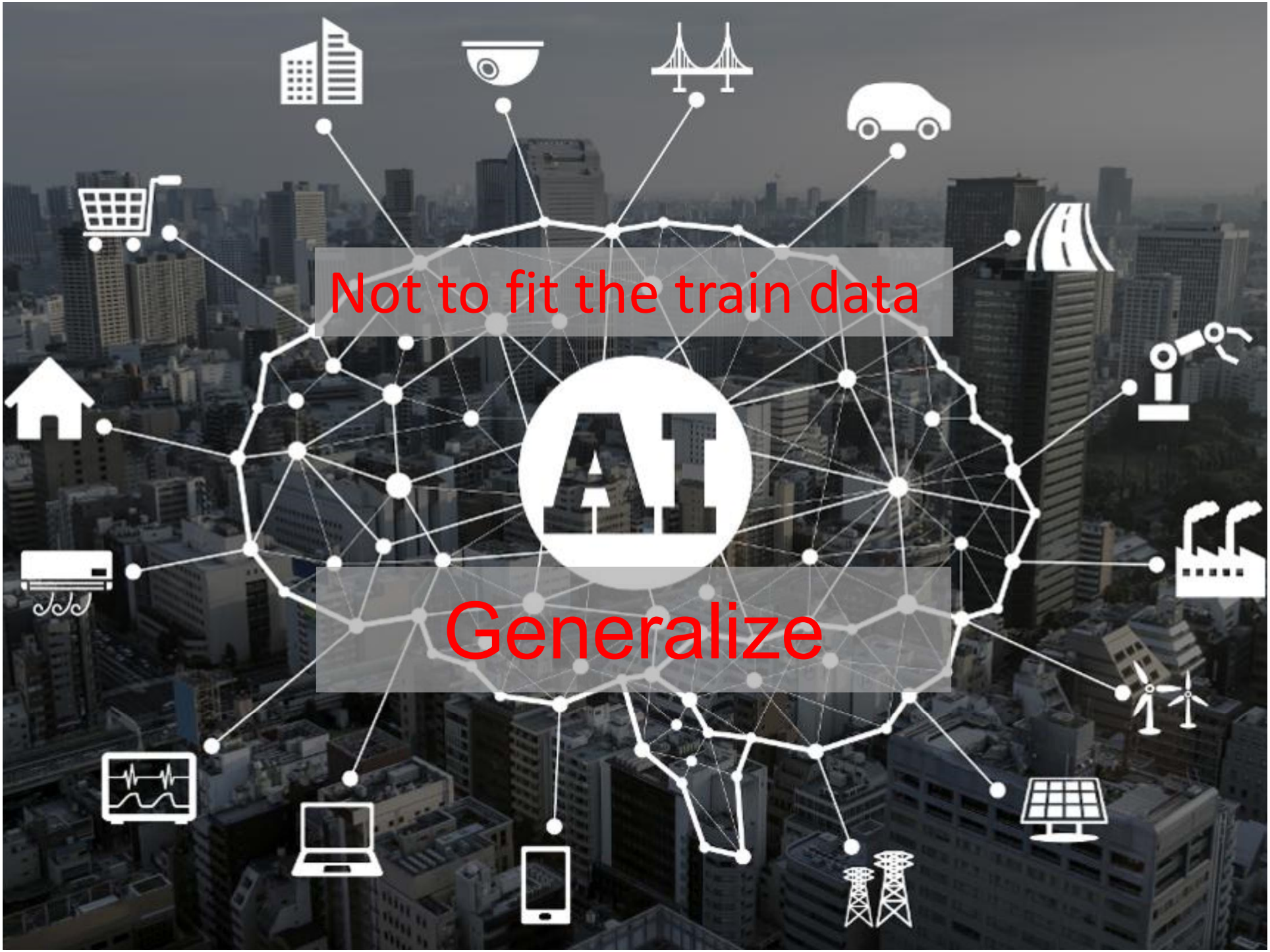
– Market basket analysis (e.g. diapers and beer)

Machine Learning Main Tasks

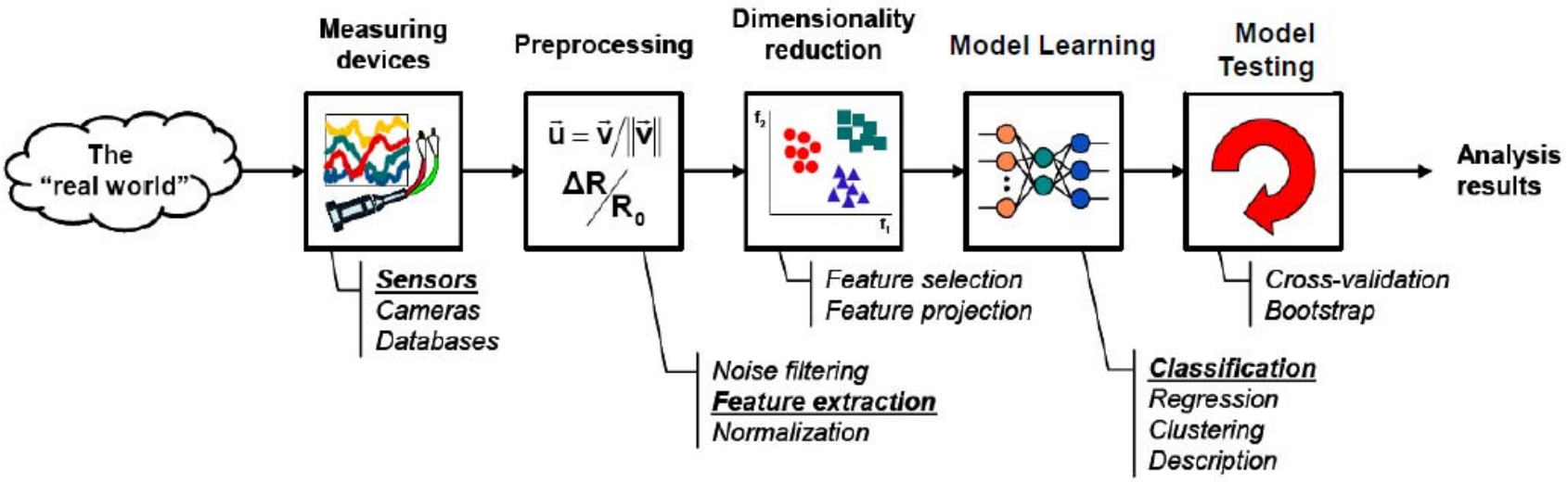
- Ability to mimic human and replace certain monotonous tasks which require some intelligence.
- like recognizing handwritten characters
- Develop systems that are too difficult/expensive to construct manually because they require specific detailed skills or knowledge tuned to a specific task (knowledge engineering bottleneck).

Human Learning

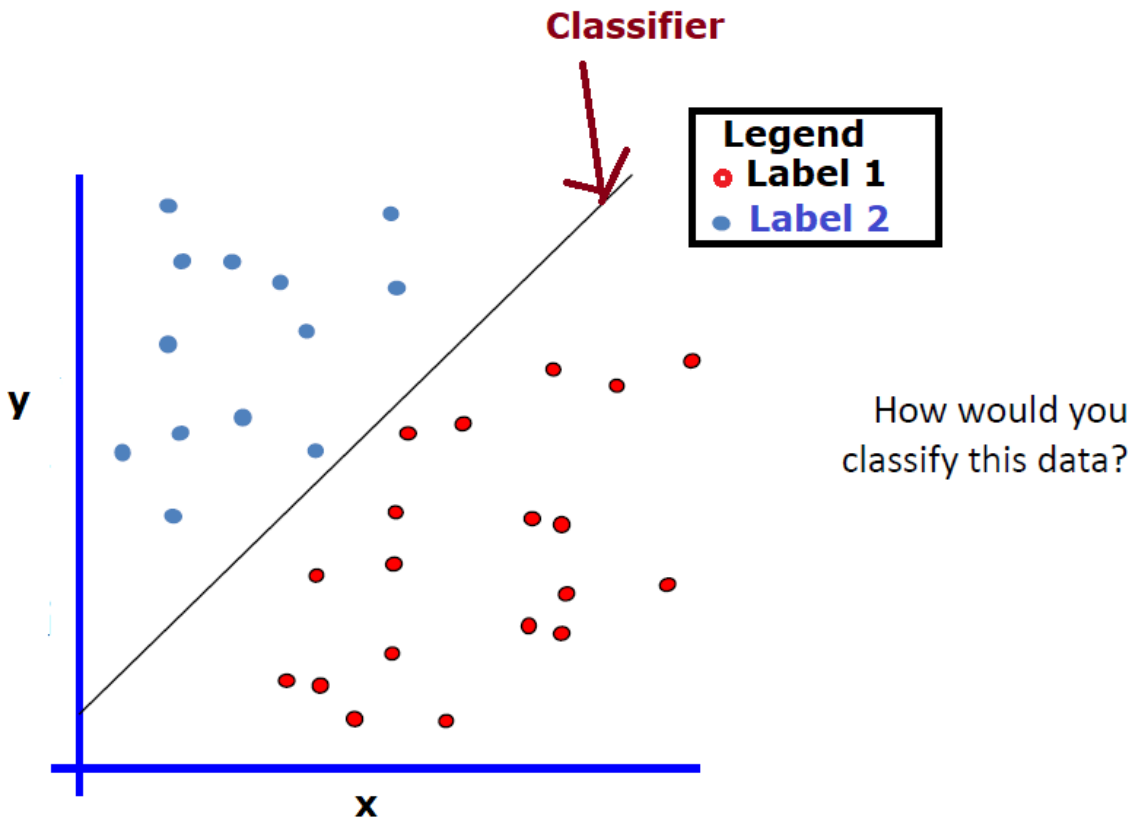
- Show sample of examples for learning
- Not for the kid to be fixed to the material
- Able to apply
- Able to extrapolate
- Able to adapt to similar cases
- Able to innovate


Not to fit the train data
AI
Generalize

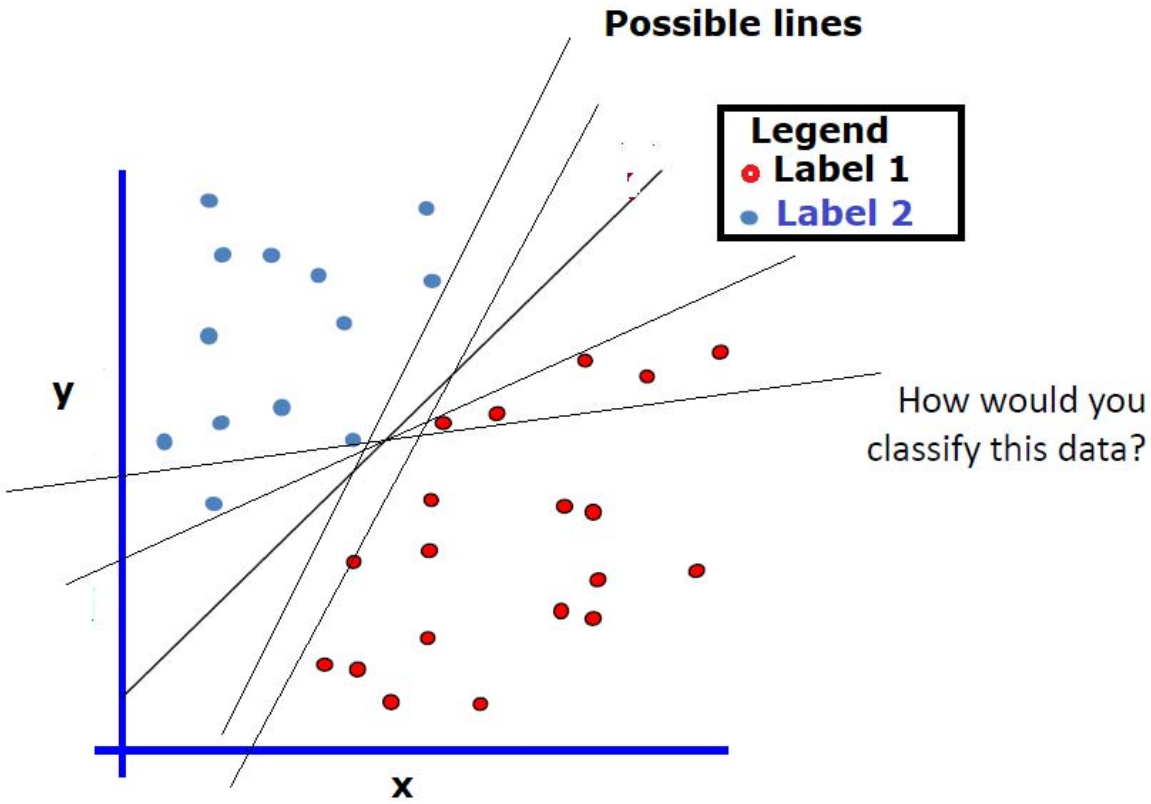
The Learning Process



Linear Classifiers



Linear Regression Y=ax+b where a is the slope and b is the y-intercept

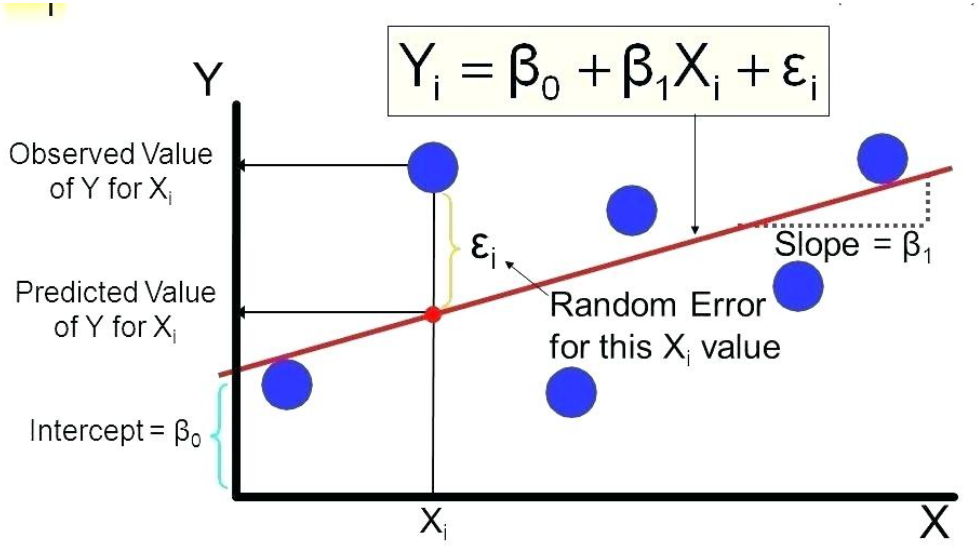


By changing the value of **a** and **b**, we could have many possible lines as classifier, which one could achieve the best performance?

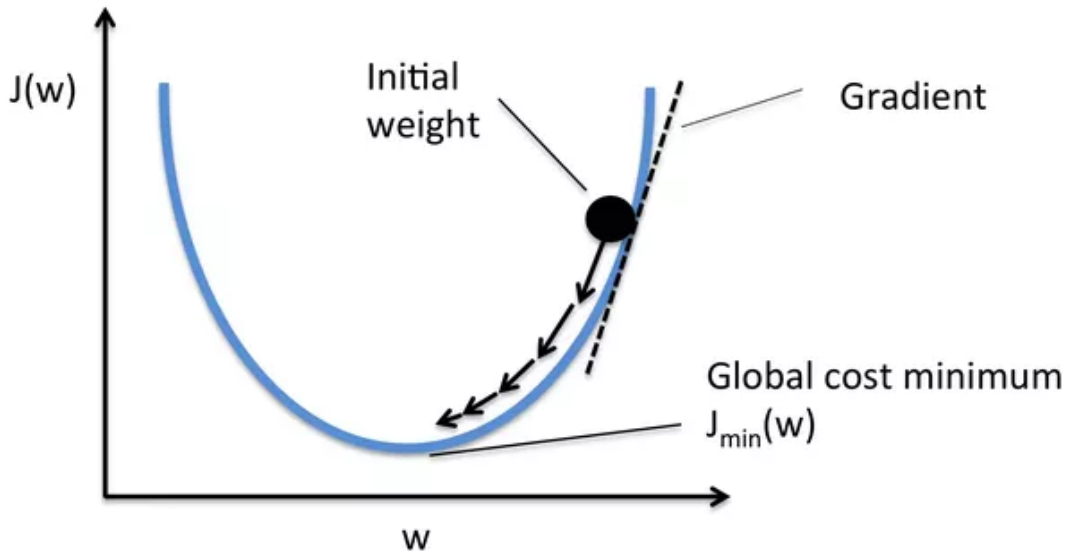
Cost Function

- Linear regression tries to fit points to a line generated by an algorithm. This optimized line (the model) is capable of predicting values for certain input values and can be plotted. The square error is the commonly cost function to optimize and find the best line that have produce the lowest square error.

$$J(w) = \frac{1}{2}\sum_{i}\left(Target^{i} - Actual^{i}\right)^{2}$$



Gradient Descent (GD)



$$J(w) = \frac{1}{2}\sum_i \left(Target^i - Actual^i\right)^2$$

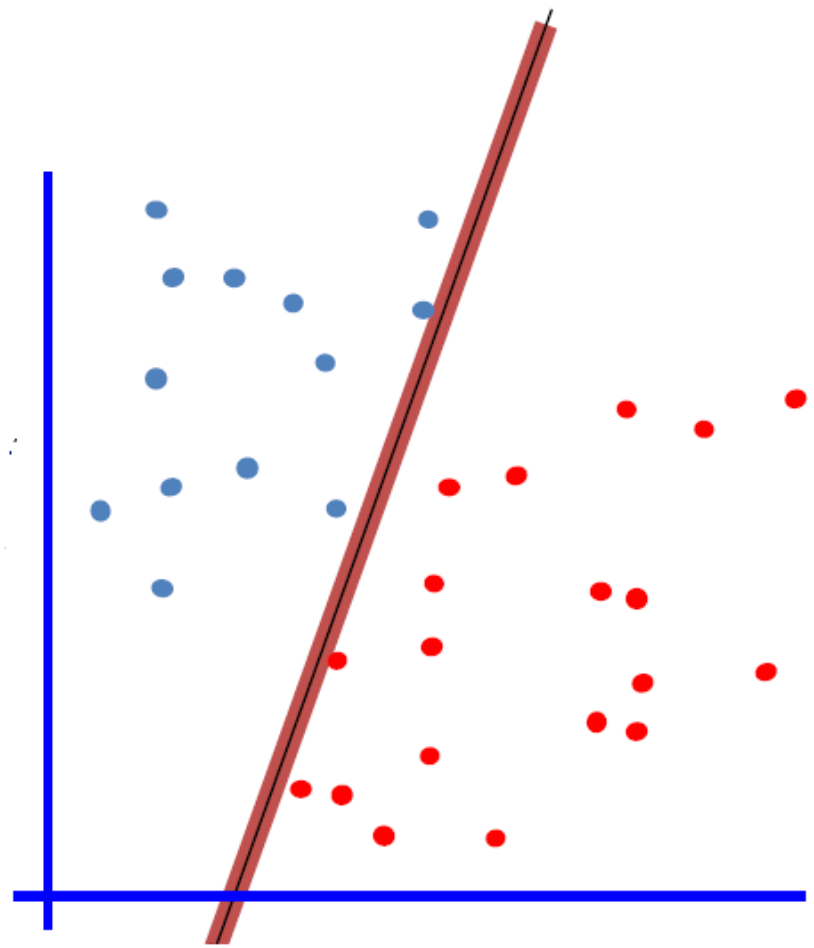
$$\Delta w_j = -\mu \frac{\partial J}{\partial w_j}$$

$$w_j = w_j + \Delta w_j$$

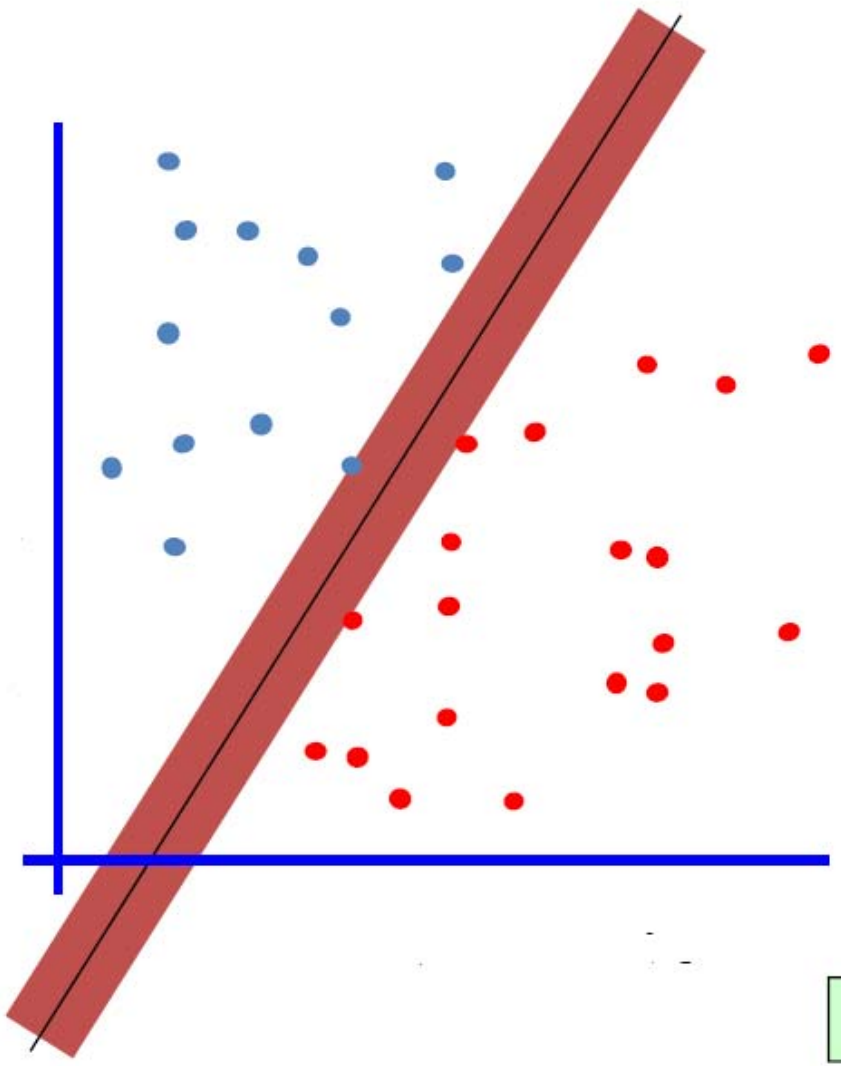
Where μ is the learning rate

Source: https://www.quora.com/Whats-the-difference-between-gradient-descent-and-stochastic-gradient-descent

- Gradient Descent is a effective yet simple optimization method commonly adapted in machine learning.

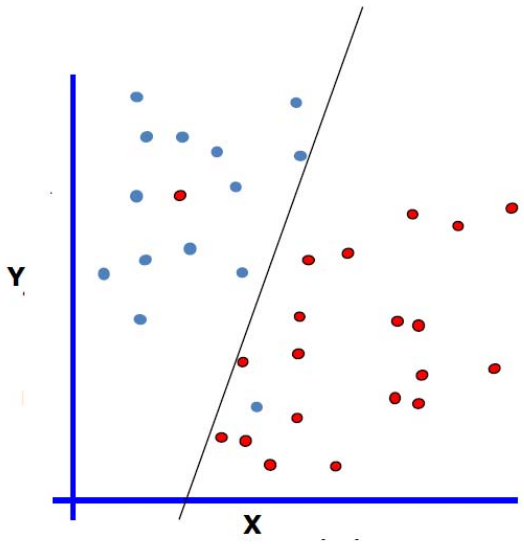


Define the margin of a linear classifier as the width that the boundary could be increased by before hitting a datapoint.



The maximum margin linear classifier is the linear classifier with the, maximum margin.
This is the simplest kind of SVM (Called an LSVM)

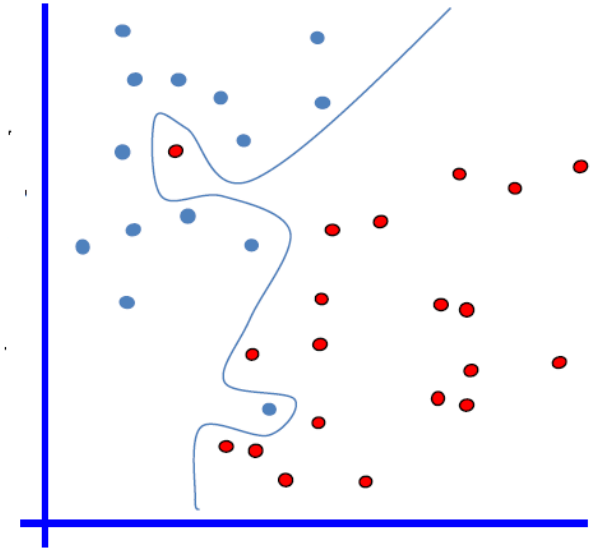
Linear SVM



How would you classify this data?

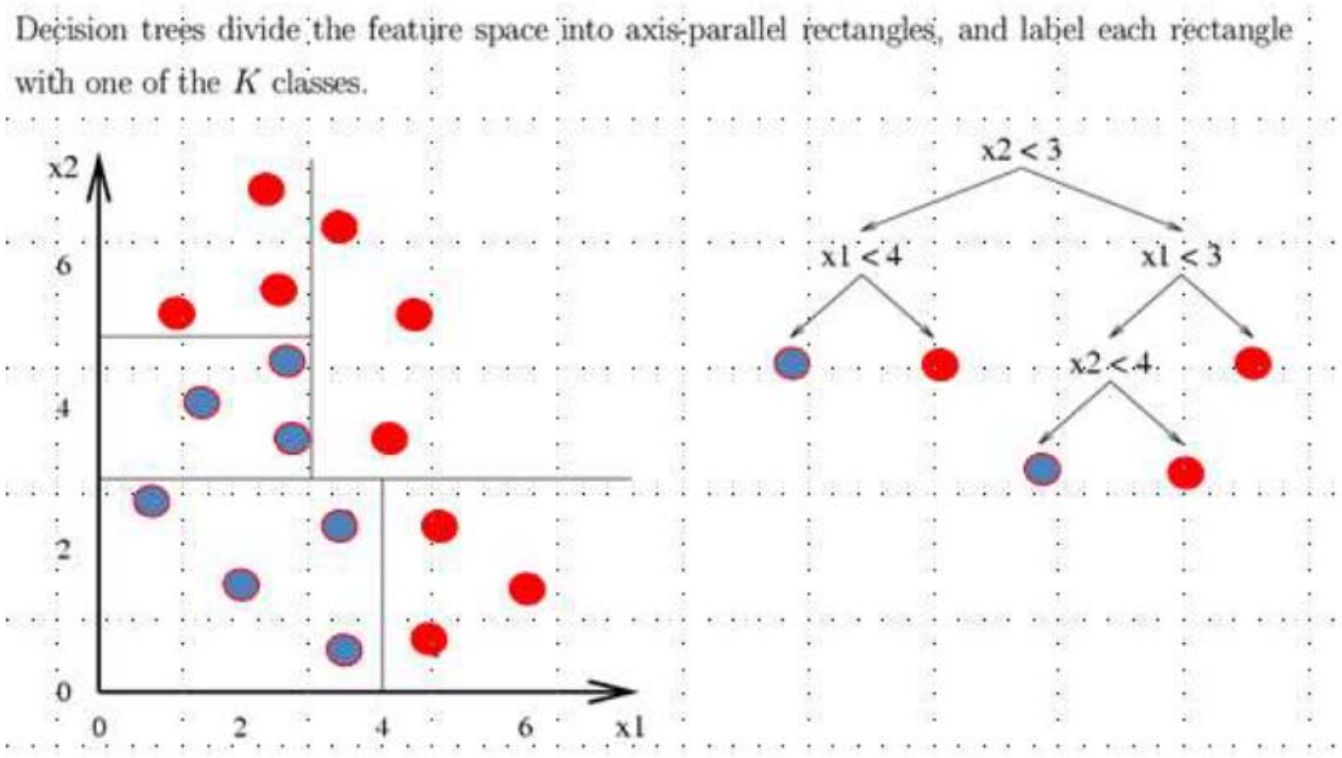
No linear classifier can cover all instances.

Non-Linear classifier is required



Linear Regression

https://www.youtube.com/watch?v=CtKeHnfK5uA



- A flow-chart like tree structure
- Internal node denotes a test on an attribute
- Branch represents an outcome of the test
- Leaf nodes represents an outcome of the test
- Leaf nodes represent class labels or class distribution

Data Split

Idea #1: Choose hyperparameters that work best on the data

BAD: always works perfectly on training data

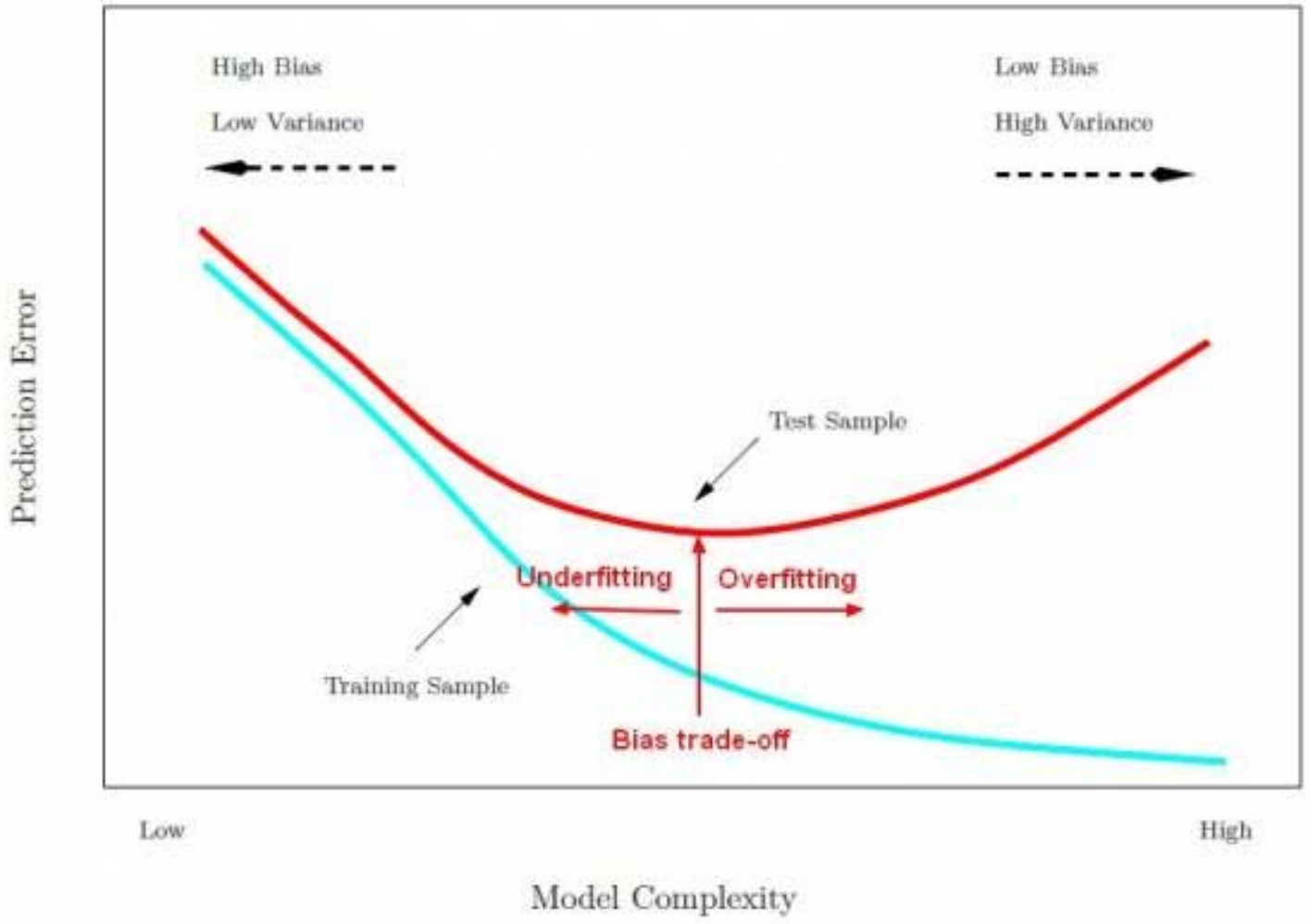
Idea #2: Split data into **train** and **test**, choose hyperparameters that work best on test data

BAD: No idea how algorithm will perform on new data

Idea #3: Split data into **train**, **val**, and **test**; choose hyperparameters on val and evaluate on test

Better!

Under fitting and Overfitting



Type I & II Errors

Figure 1

		Reality	
		H_o Is True	H_1 Is True
Conclusion	Do Not Reject H_o	Correct Conclusion	Type II Error
	Reject H_o	Type I Error	Correct Conclusion

scikit-learn
Machine Learning in Python

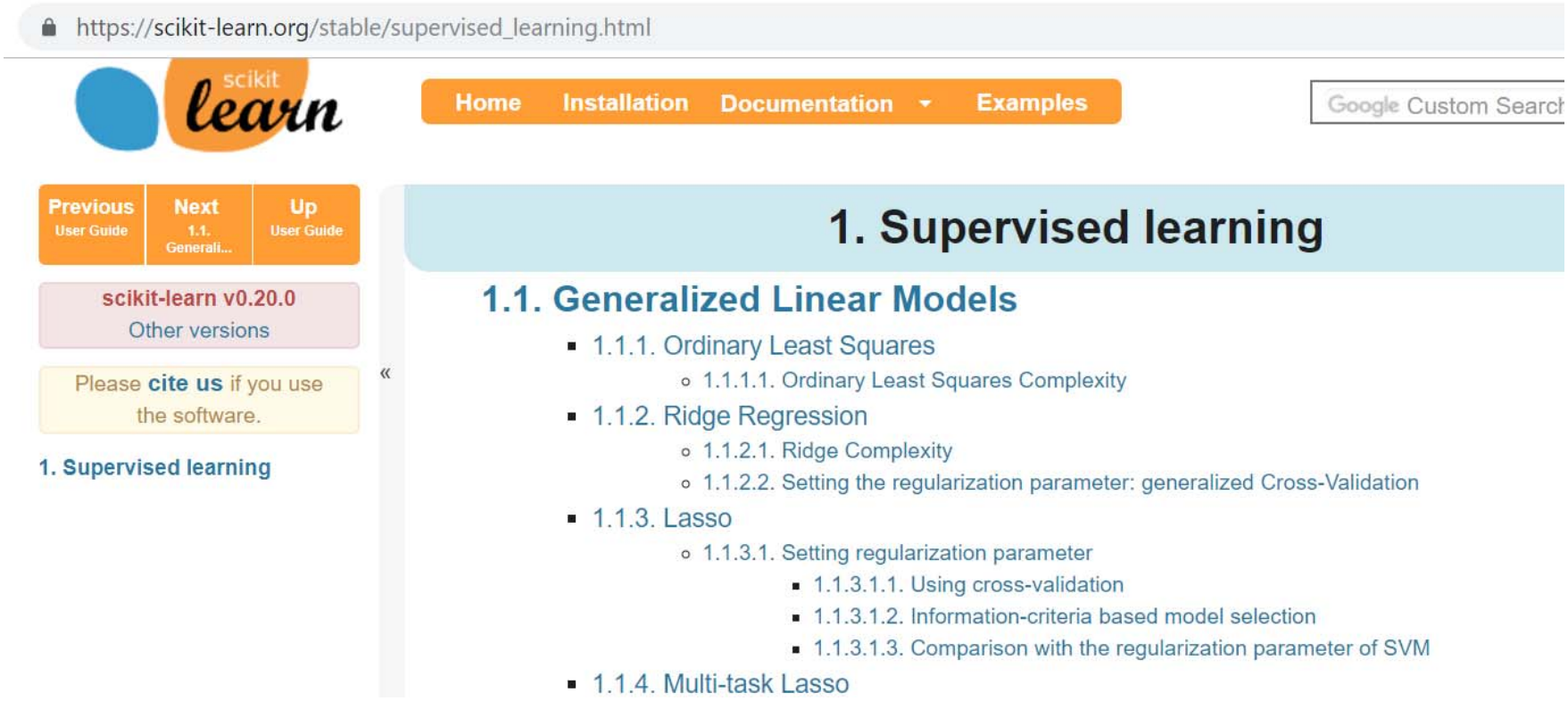
https://scikit-learn.org/stable/index.html

scikit-learn commonly used steps

Commonmly steps in using the Scikit-Learn estimator API:

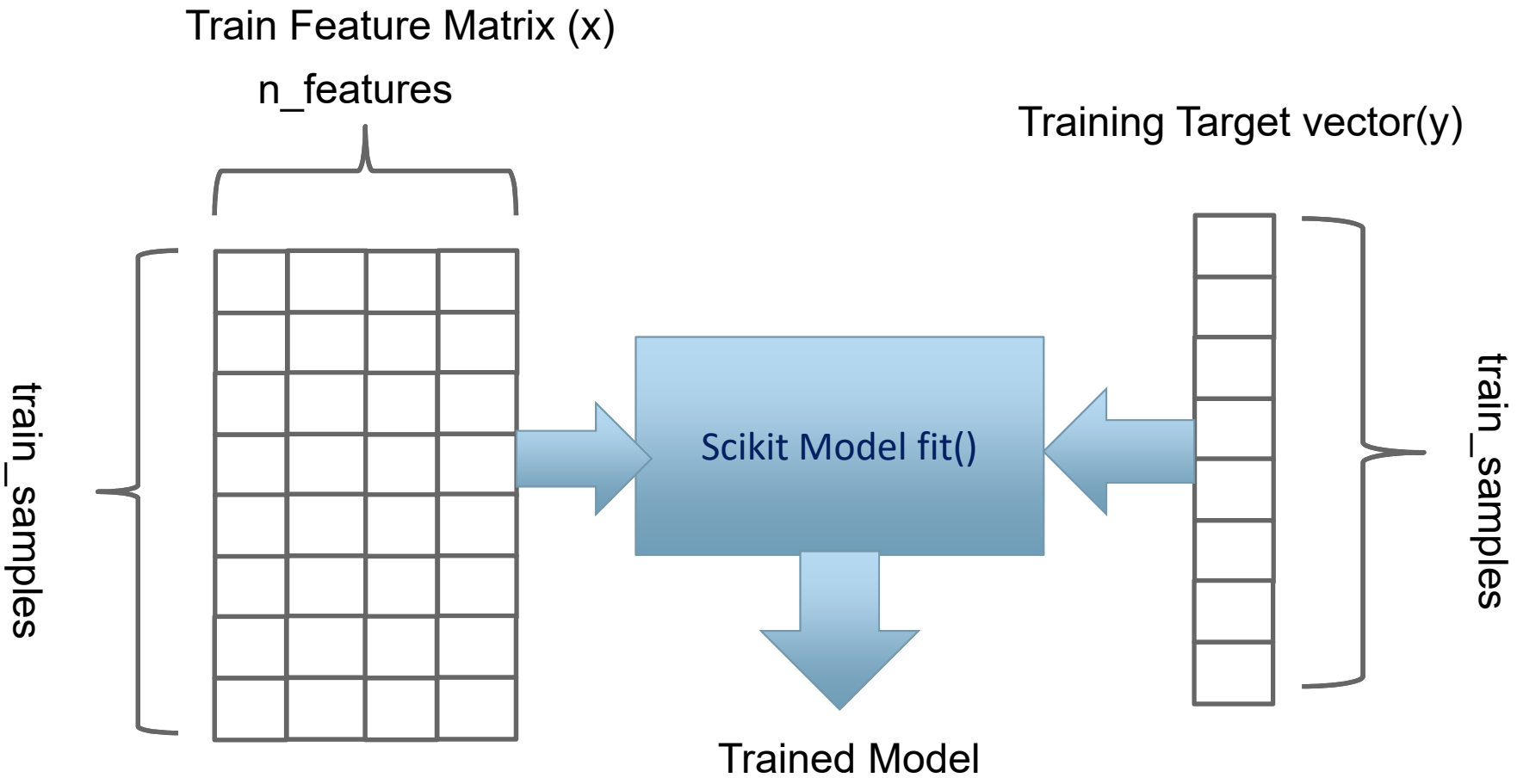
- Choose a class of model by importing the appropriate estimator class from Scikit-Learn
- Choose model hyperparameters by instantiating this class with desired values
- Arrange data into a feature matrix and target vector
- Fit the model to data by call the fit() method of the model instance.
- Apply the model to new data:
  - For supervised learning, we predict labels for unknown data using the predict() methhod
  - For unsupervised learning, we often transform or infer properties of the data using the transform or predict() methods

Choosing a class of model

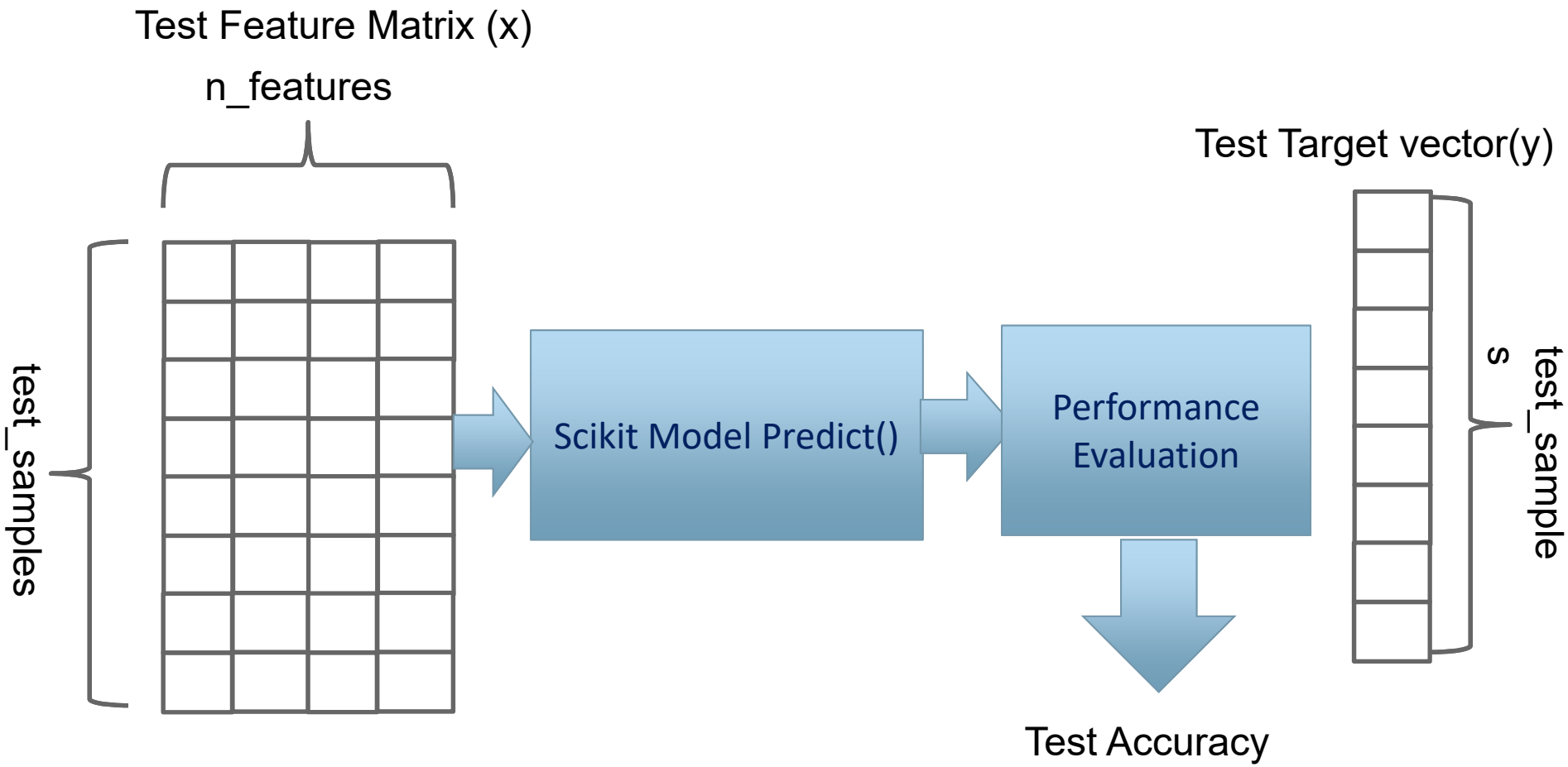


https://scikit-learn.org/stable/supervised_learning.html#supervised-learning

SCKIT Model Fit



SCKIT Model Predict



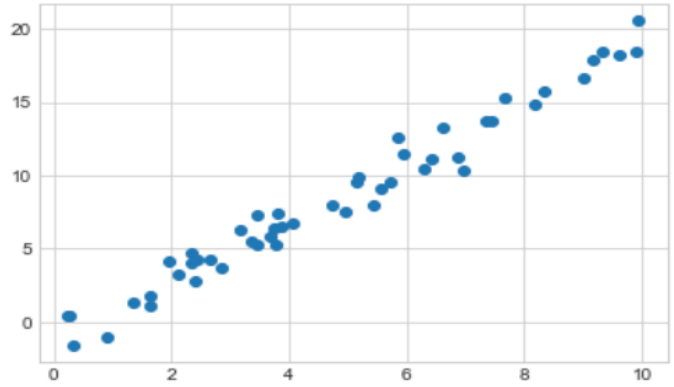
Simple Linear Example

In [1]:
```
%matplotlib inline
#import library required
import matplotlib.pyplot as plt
plt.style.use('seaborn-whitegrid') #set t he stype to use
import numpy as np

rn=np.random.RandomState(30) #set the seed for reproducibility
x=10*rn.rand(50) #produce 10 reandom number
a=2  # the slope of the line
c=5 #y insterseciton of the line

y=a*x+-1+rn.randn(50)
ax=plt.axes()
ax.scatter(x,y)
```

Out[1]: <matplotlib.collections.PathCollection at 0x1609dc83f28>



- Generate a 2-dimensional data x and y
- The relationship is linear y=ax+b
- X is randomly generated

Simple Linear Example

```
1  #use a sklearn linear model to fit the data
2  from sklearn.linear_model import LinearRegression      #step 1
3  model=LinearRegression(fit_intercept=True)             #step 2
4  print(x.shape)
5  X=x[:,np.newaxis]                                      #step 3
6  print(X.shape)
7  model.fit(X,y)                                         #step 4
8  print(model.coef_)          #print the coeffficents
9  print(model.intercept_)     #print the incept value
10 xfit=np.linspace(-1,11)
11 xfit=xfit[:,np.newaxis]
12 yfit=model.predict(xfit)                               #step 5
13 plt.scatter(x,y)
14 plt.plot(xfit,yfit,c='r')
```

```
(50,)
(50, 1)
[2.02341135]
-1.2628188060722358
```

Import chosen Linear Regression model

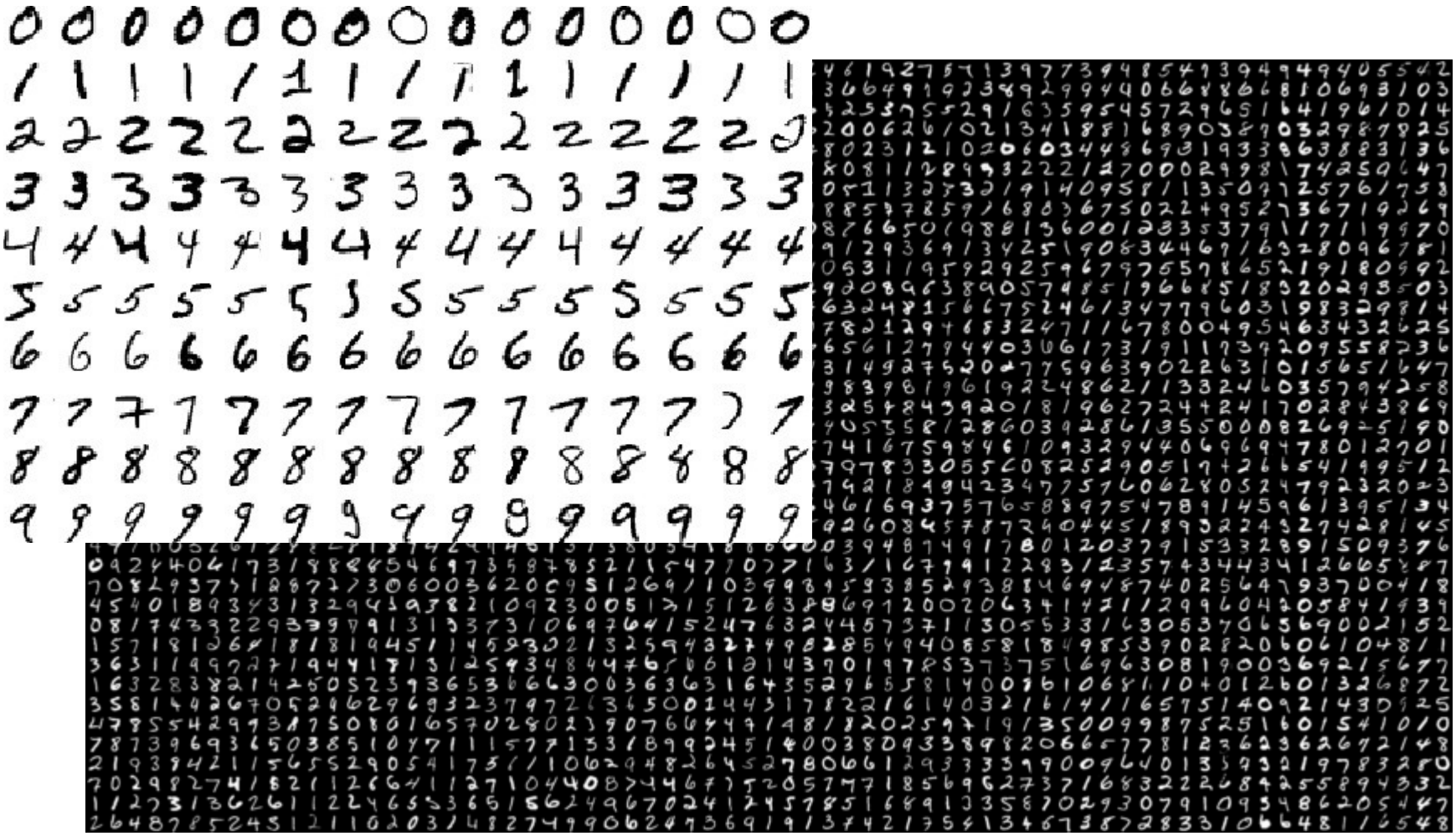
initialize the class with desired hyperparameter

Arrange data into a feature matrix

Fit the model to data by call the fit() method

Apply the model for evaluation

MNIST Handwritten Dataset



MNIST Handwritten Dataset

- **The MNIST database (Modified National Institute of Standards and Technology database)** is a large database of handwritten digits that is commonly used for training various image processing systems.

- The database is also widely used for training and testing in the field of machine learning.

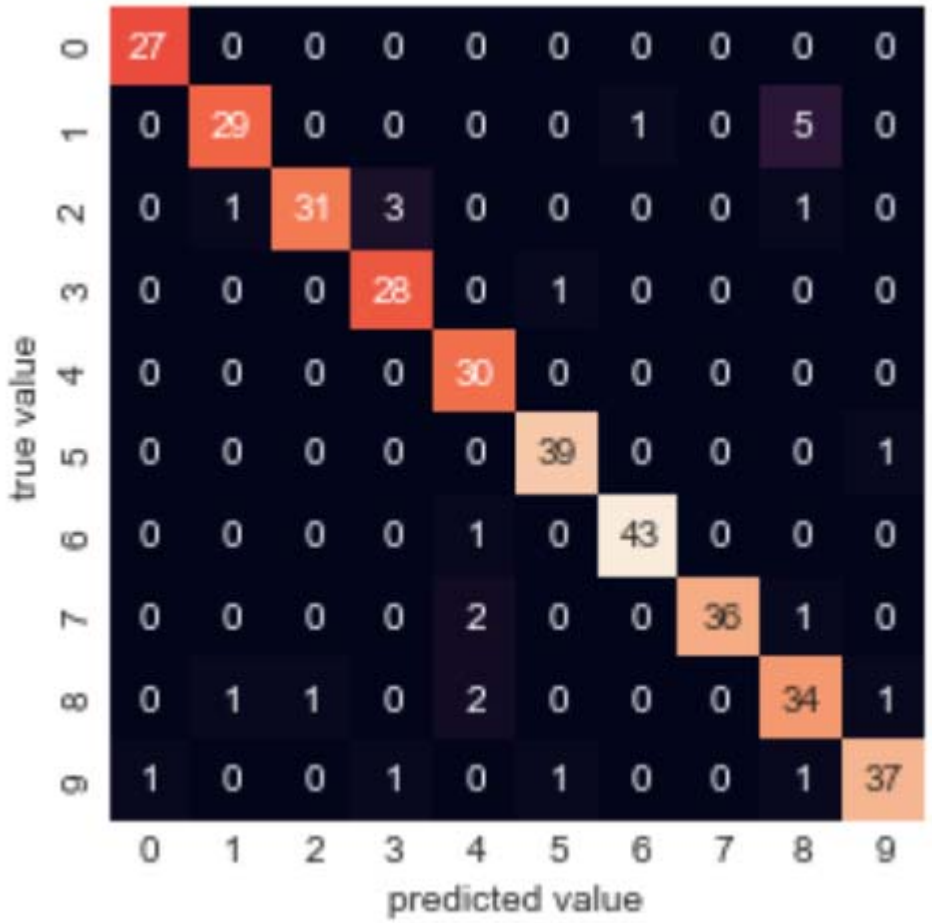
- It was created by "re-mixing" the samples from NIST's original datasets. Half of the training set and half of the test set were taken from NIST's training dataset, while the other half of the training set and the other half of the test set were taken from NIST's testing dataset.

- The MNIST database contains 60,000 training images and 10,000 testing images.

More information on the digit dataset from Sickit:
http://scikit-learn.org/stable/auto_examples/datasets/plot_digits_last_image.html
Detail information on the digit dataset from UCI:
http://archive.ics.uci.edu/ml/datasets/PenBased+Recognition+of+Handwritten+Digits

0.9277777777777778
0.9277777777777778

Confusion Matrix for Classification



Confusion Matrix plotted as heat-map for SGDClassifier for MNIST Handwritten Dataset with classification accuracy of 92%