Supervised Learning

경희대학교 컴퓨터공학과
2019102191 신주영

Table of contents

Supervised Learning

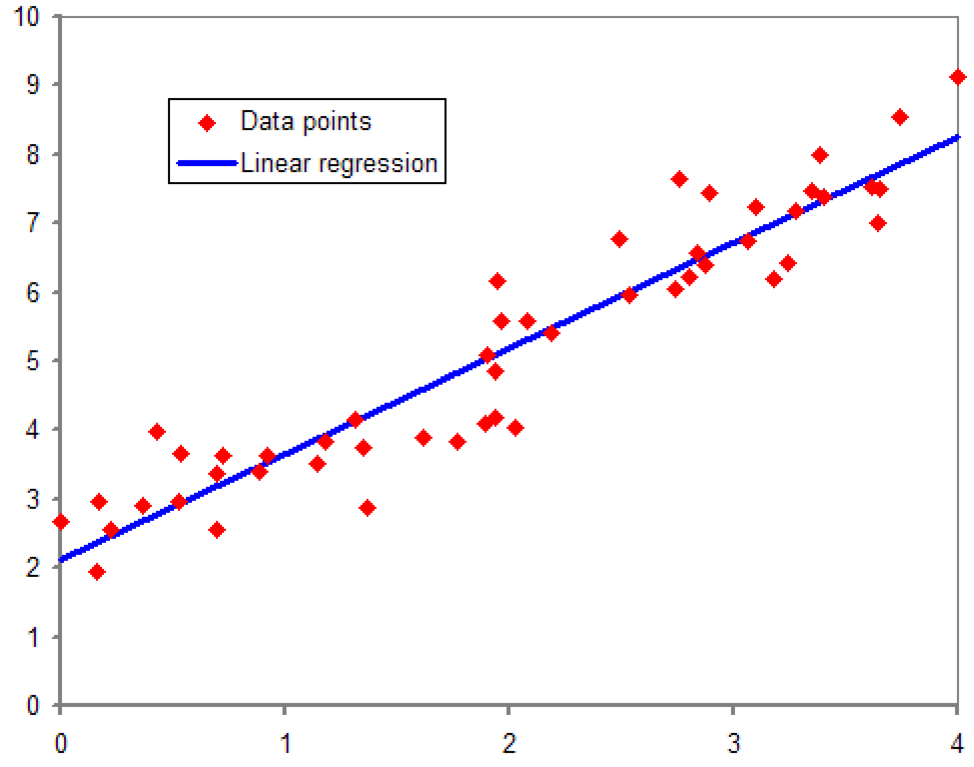
Linear Regression

경희대학교 컴퓨터공학과
2019102191 신주영

KYUNG HEE UNIVERSITY

Fundamental concept

- A technique for modeling relationships in which dependent variables change large or small as independent variables grow
- y = ax + b, a is called the inclination and b is called the y-intercept



Algorithm

- Usually, in linear regression, the learning parameters a and b of the data points that are not on a straight line must be calculated
- Create a straight line to minimize the mean square error

- Mean square error formula

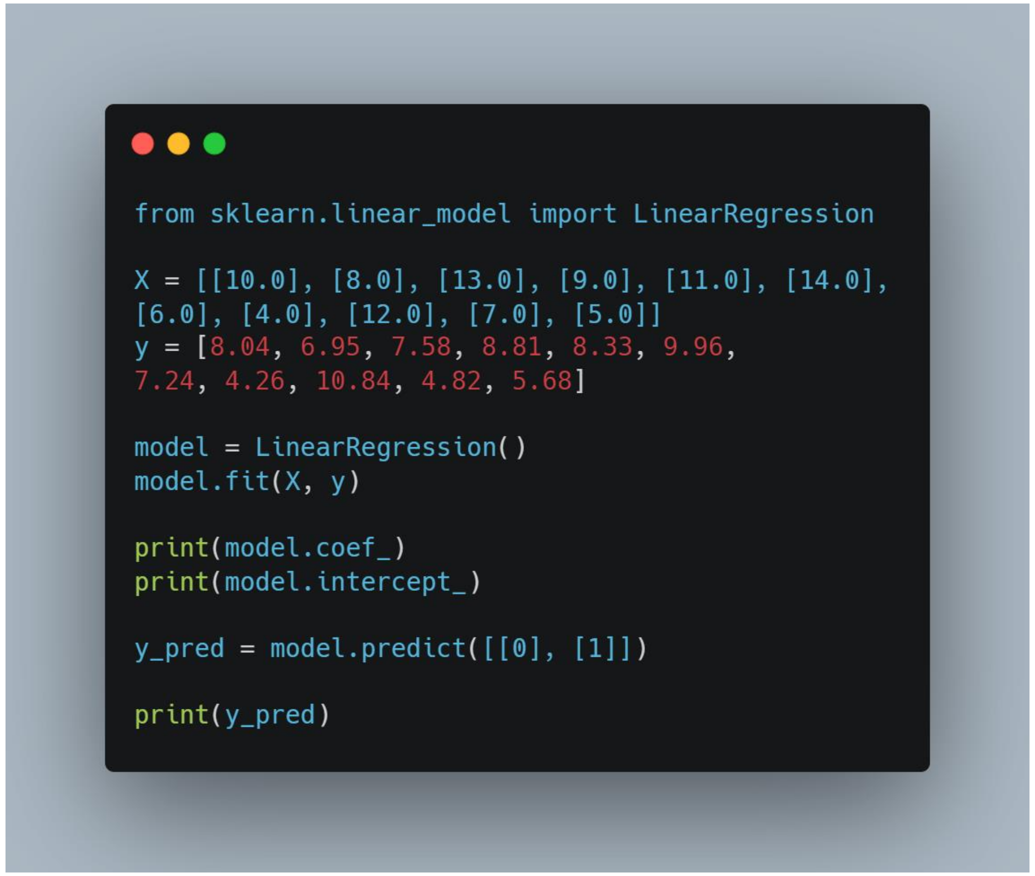
$$\sum_{i=1}^{n} \frac{y_i - (a + bx_i)^2}{n}$$

Algorithm

- A function that defines the relationship between error and learning parameters is called a loss function.
- Linear regression finds a parameter with a minimum loss function value among various straight lines.
- Finding parameters that minimize loss function values is also a concept commonly used in guidance learning algorithms.

Sample code

- From sklearn, import Linear Regression
- Set X, y variable
- Make Linear Regression variable
- Fit X, y to Linear Regression model
- Print inclination and y-intercept
- Predict y-values when x is 0 and 1 with the predict method



```python
from sklearn.linear_model import LinearRegression

X = [[10.0], [8.0], [13.0], [9.0], [11.0], [14.0],
[6.0], [4.0], [12.0], [7.0], [5.0]]
y = [8.04, 6.95, 7.58, 8.81, 8.33, 9.96,
7.24, 4.26, 10.84, 4.82, 5.68]

model = LinearRegression()
model.fit(X, y)

print(model.coef_)
print(model.intercept_)

y_pred = model.predict([[0], [1]])

print(y_pred)
```

Output

- We can get inclination by model.coef_ method
- We can get y-intercept by model.intercept_ method
- We can get y-values when x is 0 and 1 by model.predict method

```
[0.50009091]
3.0000909090909094
[3.00009091 3.50018182]
```

Method of minimizing mean square error

- Let x and y as shown on the right.
- The table below shows the average square error values according to the values a and b.
- Substituting the x and y values into the mean square error formula, the following equation can be obtained.
- Therefore, the minimum value of the equation according to the values a and b may be obtained.

i	x	y
0	2	1
1	3	5
2	6	3
3	7	7

선형회귀	a	b	평균제곱오차
(a)	0.823	0.706	2.89
(b)	4.5	-0.125	5.83

$$\sum_{i=1}^{4} \frac{y_i - (a+bx_i)^2}{4} = a^2 + 24.5b^2 + 9ab - 8a - 42a + 21$$

Type of linear regression

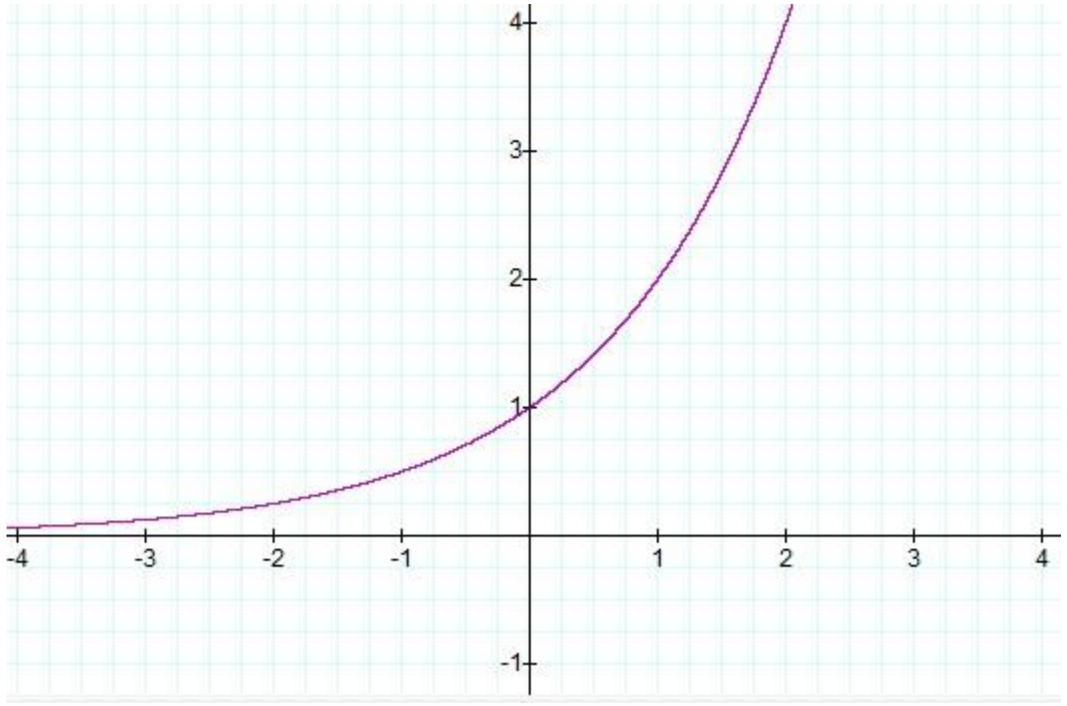
- Since polynomial regression is not linear, it may be awkward to call it linear regression, but because it is linear regression for learning parameters rather than independent variables, polynomial regression is also included in linear regression.

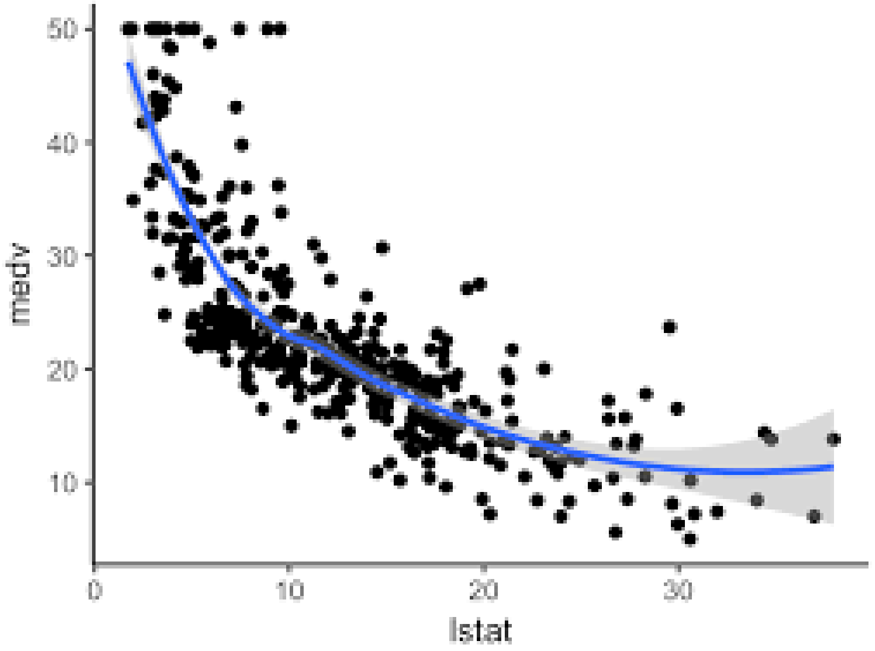
선형회귀의 종류	함수 정의 예
단순선형회귀 Simple linear regression	y = ax + b
다중선형회귀 multiple linear regression	z = ax + by + c
다항회귀 Polynomial regression	y = ax^2 + bx + c

경희대학교
KYUNG HEE UNIVERSITY

Nonlinear regression

- y = e^x
- y = 1/x





Supervised Learning

Regularization

경희대학교 컴퓨터공학과
2019102191 신주영

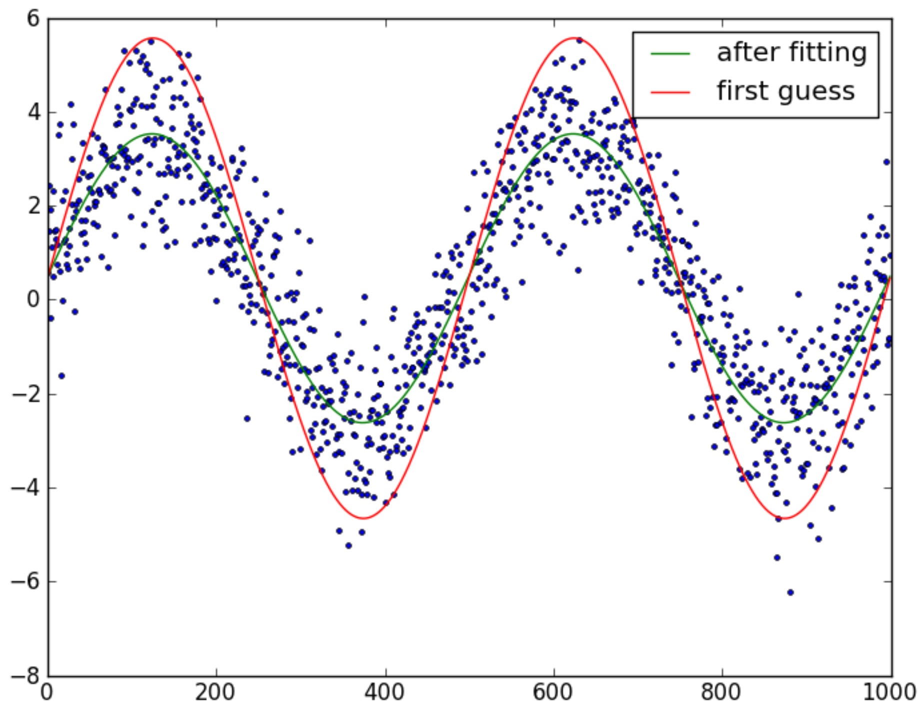
Fundamental concept

- Normalization is used as a way to prevent over-fitting of the model
- In over-fitting, the error between the execution result and the correct answer data is very small, but the error between the result of executing the model with test data and the correct answer data is very large

Fundamental concept

making practice data

- We created data by adding random numbers to the function $y = \sin(2\pi\chi)$
- We'll apply linear regression to this data to make it a machine learning model
- When the order is increased, we'll check how the learning error and test error change



Learning errors and test errors by order

- As the order increases, the learning error gradually decreases
- The test error increases as the order increases
- The sixth-order linear regression is a complex model, so learning errors are reduced, but over-fitting occurs and cannot be called a generalized model.

차수	학습오차	검정오차
1	0.412	0.618
2	0.176	0.193
3	0.081	0.492
…	…	…
6	0.024	3.472

경희대학교
KYUNG HEE UNIVERSITY

Consequences of using normalization for linear regression

- Normalization prevents over-fitting by adding a penalty term to the loss function
- When the order increased, the test error did not increase, so that over-fitting could be prevented
- We're going to regularization method as a ridge regression

차수	학습오차	검정오차
1	0.412	0.612
2	0.372	0.532
3	0.301	0.394
…	…	…
6	0.159	0.331

Algorithm

only linear regression

- As the order increases, the absolute value of the learning parameter gradually increases.

차수	w_1	w_2	w_3	w_4	w_5	w_6
1	-0.007	-0.217	-	-	-	-
2	0.978	-5.222	4.204	-	-	-
3	0.281	4.927	-17.639	12.157	-	-
…	…	…	…	…	…	…
6	1.080	-26.324	287.431	-1034.141	-1611.144	-1147.946

Algorithm
linear regression with normalization

- Normalization reduces the increase in learning parameters.

차수	w_1	w_2	w_3	w_4	w_5	w_6
1	-0.055	-0.149	-	-	-	-
2	-0.066	-0.493	0.421	-	-	-
3	-0.001	-0.716	-0.042	0.670	-	-
…	…	…	…	…	…	…
6	0.191	-0.751	-0.497	-0.182	0.109	0.370

Algorithm

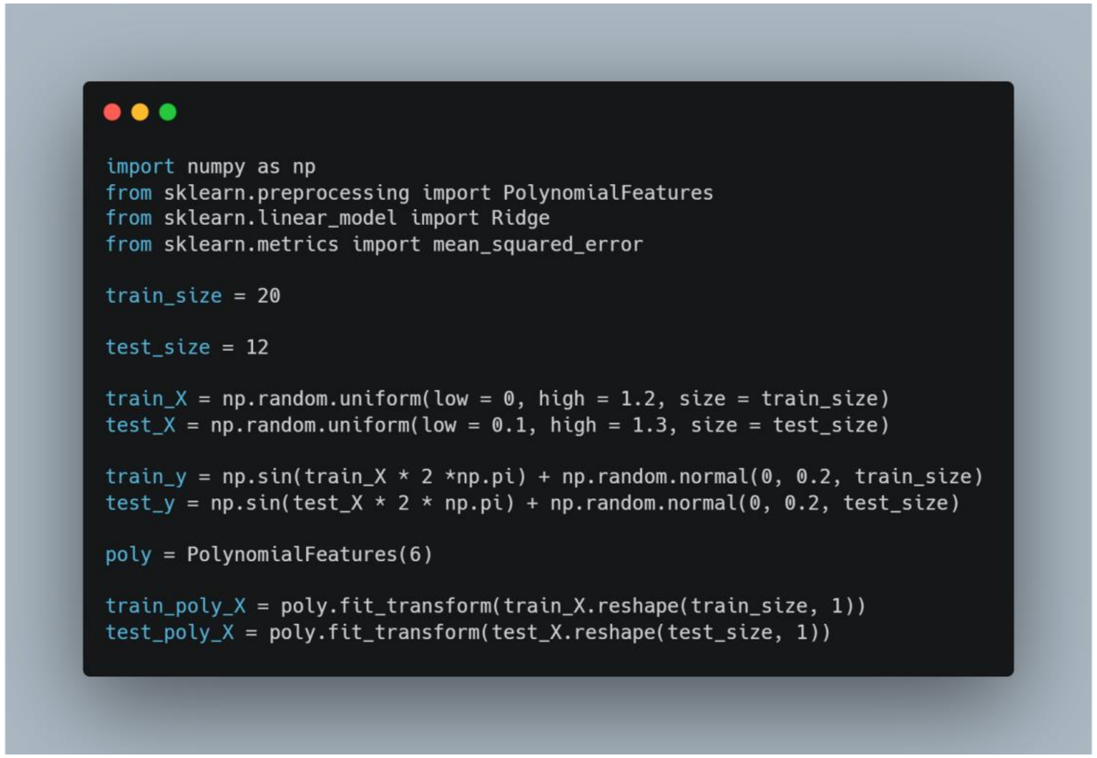
loss function and penalty term

- $\sum_{i=1}^{n}\{y_i - (c + bx_i + ax_i{}^2)\}^2 + \alpha(a^2 + b^2)$
- $\sum_{i=1}^{n}\{y_i - (c + bx_i + ax_i{}^2)\}^2$ is a linear regression loss function
- $\alpha(a^2 + b^2)$ is a penalty term which is learning parameter's square shape
- $\alpha(\geq 0)$ is a parameter that adjusts the intensity of normalization
- If a, b is increasing, $\alpha(a^2 + b^2)$ is increase, and linear regression loss function value increase
- By making a, b minimize, $\alpha(a^2 + b^2)$ is minimized and linear regression loss function minimized

경희대학교
KYUNG HEE UNIVERSITY

Sample code

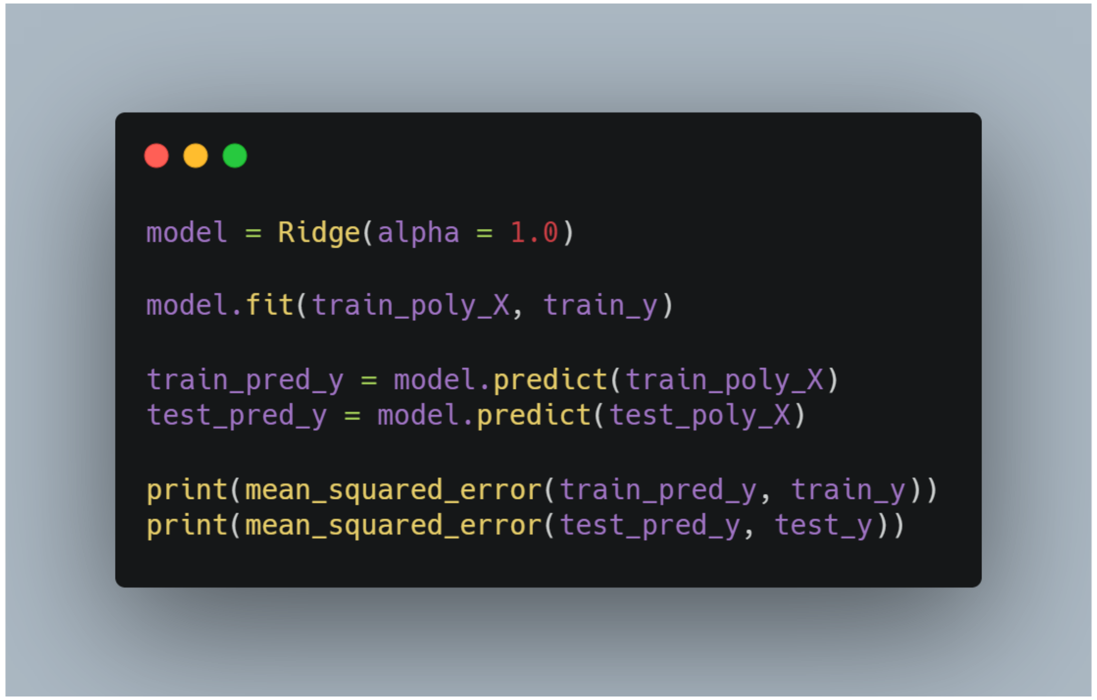
- Import numpy, PolynomialFeatures, Ridge, mean_squared_error
- Set training data and test data
- Construct a six-order linear regression model
- Conversion of the results into an array after learning the characteristics of the learning data in the sixth equation



```
import numpy as np
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import Ridge
from sklearn.metrics import mean_squared_error

train_size = 20

test_size = 12

train_X = np.random.uniform(low = 0, high = 1.2, size = train_size)
test_X = np.random.uniform(low = 0.1, high = 1.3, size = test_size)

train_y = np.sin(train_X * 2 *np.pi) + np.random.normal(0, 0.2, train_size)
test_y = np.sin(test_X * 2 * np.pi) + np.random.normal(0, 0.2, test_size)

poly = PolynomialFeatures(6)

train_poly_X = poly.fit_transform(train_X.reshape(train_size, 1))
test_poly_X = poly.fit_transform(test_X.reshape(test_size, 1))
```

Sample code

- Make ridge model and learn as a dependent variable of 6th-order learning outcomes and learning data
- Predict output results of models made by learning from rigid regression
- Calculation of mean square error



```
model = Ridge(alpha = 1.0)

model.fit(train_poly_X, train_y)

train_pred_y = model.predict(train_poly_X)
test_pred_y = model.predict(test_poly_X)

print(mean_squared_error(train_pred_y, train_y))
print(mean_squared_error(test_pred_y, test_y))
```

경희대학교
KYUNG HEE UNIVERSITY

Adjust normalization strength with α

- If the α value is large, the increase in learning parameters decreases and the graph shape becomes simple.
- Conversely, if the α value is small, a penalty when the absolute value of the learning parameter increases is hardly given, making the model complicated.
- If $\alpha = 0$, it is the same state as linear regression without normalization.

LASSO regression

- In ridge regression, the graph is circular as shown in the right picture because the penalty term is the sum of the learning parameters squared.
- In LASSO regression, the penalty term is the sum of the absolute values, so the graph is square as shown in the figure on the right.

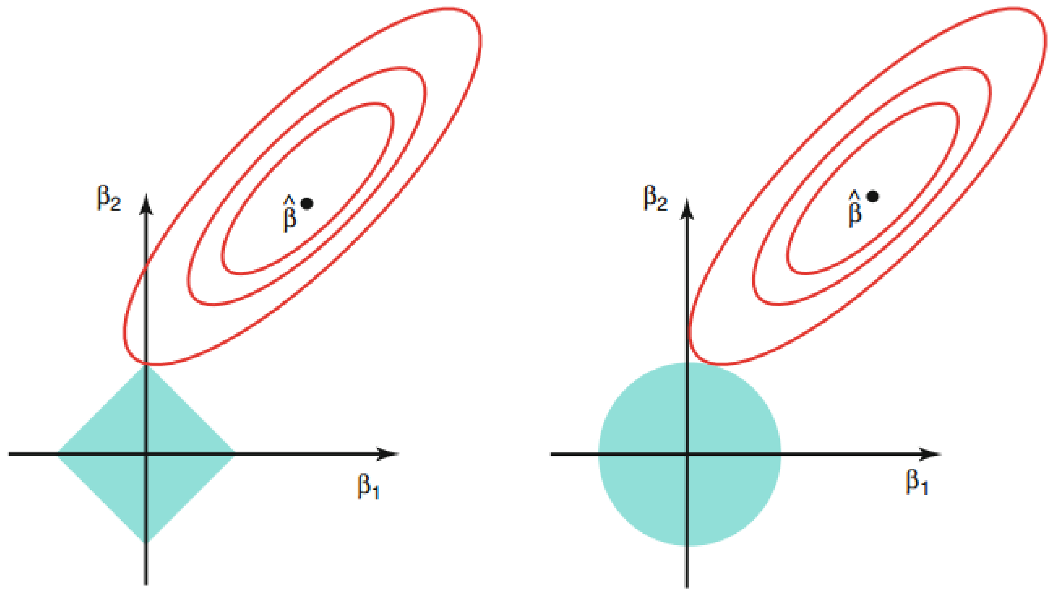


FIGURE 6.7. *Contours of the error and constraint functions for the lasso* (left) *and ridge regression* (right). *The solid blue areas are the constraint regions, $|\beta_1| + |\beta_2| \le s$ and $\beta_1^2 + \beta_2^2 \le s$, while the red ellipses are the contours of the RSS.*

LASSO regression

- $\sum_{i=1}^{n}\{y_i - (c + bx_i + a{x_i}^2)\}^2 + \alpha(|a| + |b|)$
- Ridge regression and LASSO regression are the same in that the increase in learning parameters decreases as the penalty term is added
- In LASSO regression, there is a difference that the b learning parameter is 0 at optimum solution
- Therefore, in LASSO regression, the b parameter tends to become 0

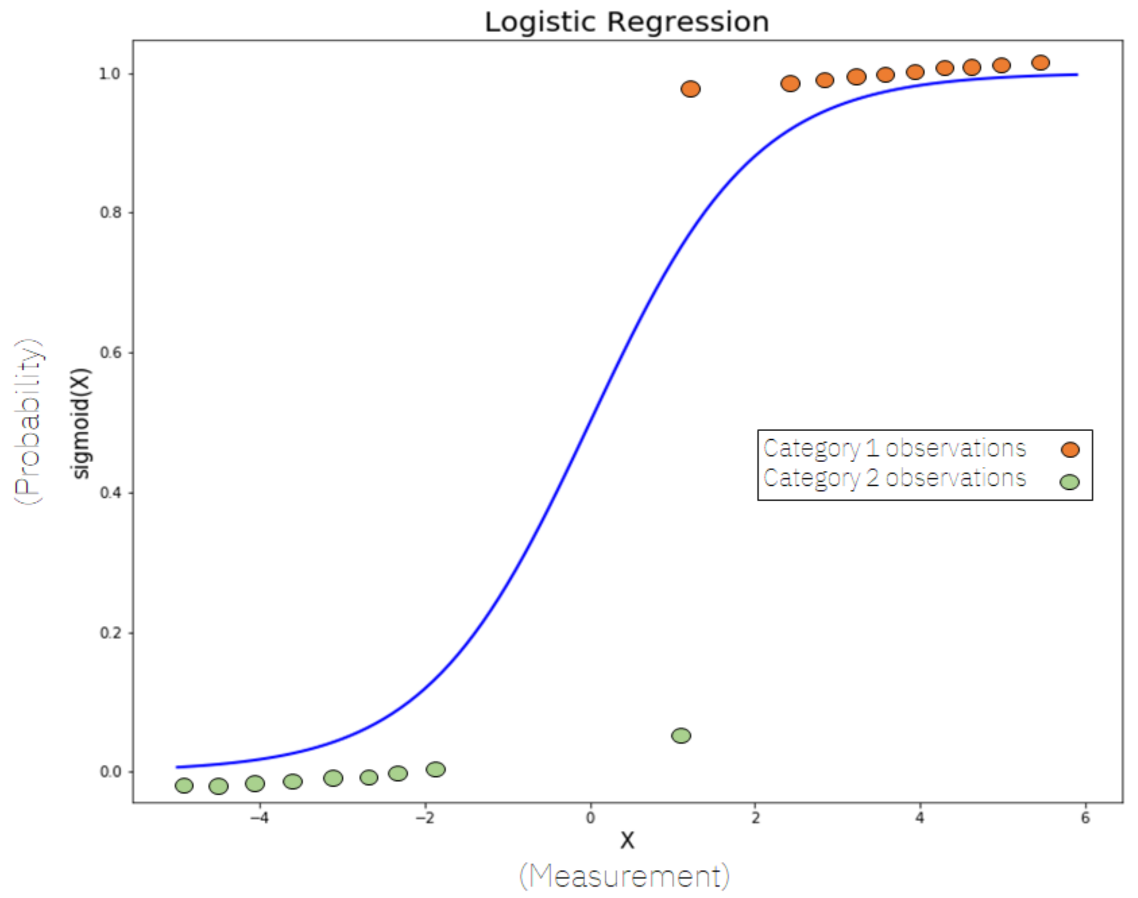
Supervised Learning

Logistic Regression

경희대학교 컴퓨터공학과
2019102191 신주영

경희대학교
KYUNG HEE UNIVERSITY

Fundamental Concept



Fundamental Concept

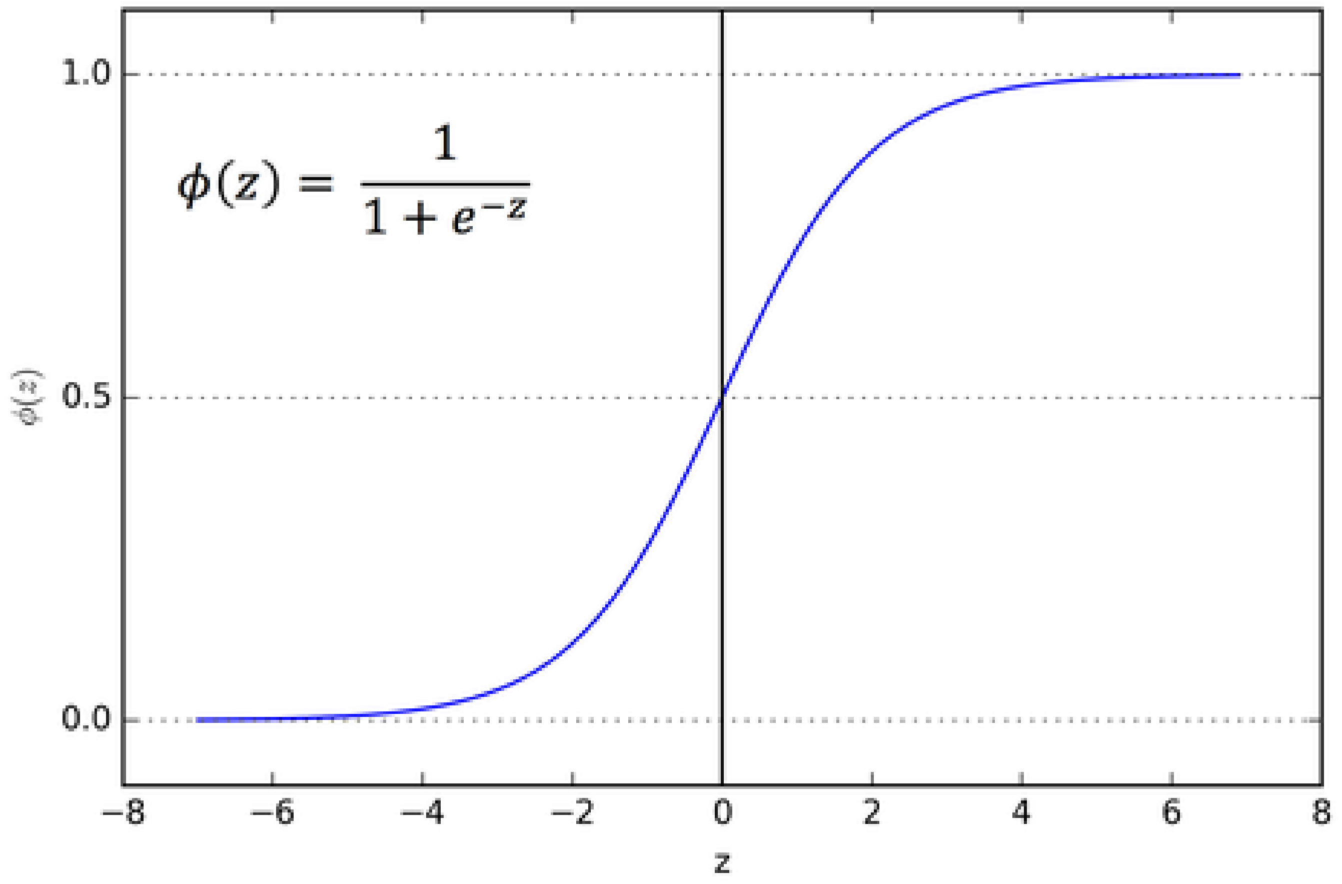
- Logistic regression is an algorithm that learns the probability of an event occurring
- Usually, binary classification is performed, but more than three classification problems can also be dealt with

Algorithm

- The deflection w_0 is added to the weight vector w corresponding to the data x to calculate $w^T x + w_0$
- The same is true of learning the weight vector w and the deflection w_0 from the data
- Unlike linear regression, the probability is calculated, so the range of output results should be between 0 and 1
- Therefore, a value between 0 and 1 is returned using the sigmoid function
- Sigmoid function: $\partial(z) = \frac{1}{1+e^{-z}}$

Algorithm

sigmoid function graph



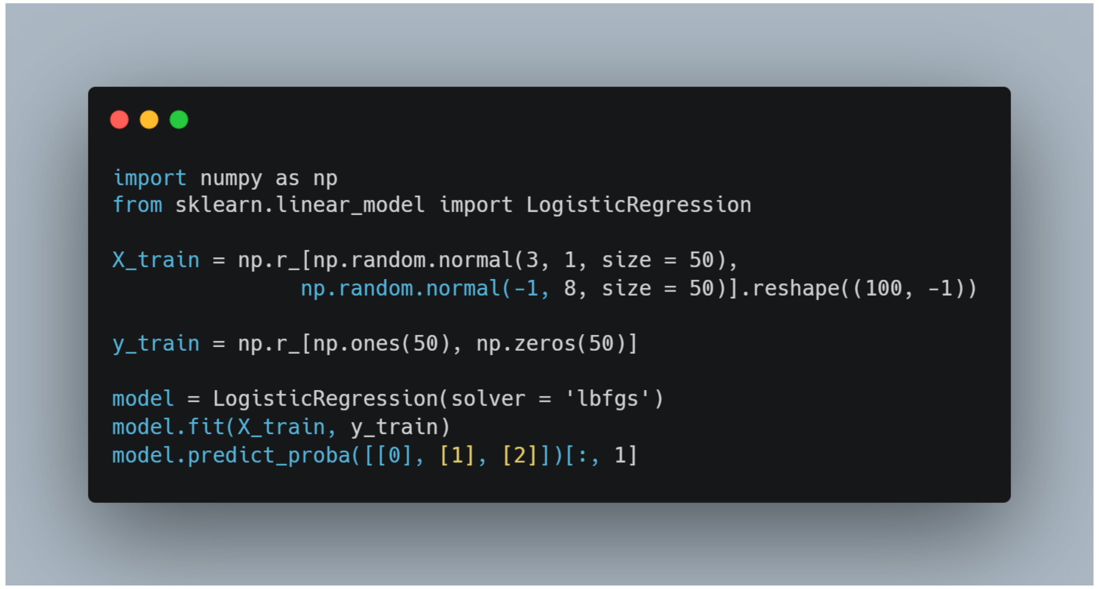
Algorithm

- $\partial(w^T x + w_0)$ is calculated as a probability p in which a label corresponding to data x is y as a sigmoid function.
- Binary classification usually takes the probability of a prediction result of 0.5 as a threshold.
- When learning, the error is minimized with the loss function of logistic regression.
- Find the minimum value of the loss function value while calculating the slope of the logistic regression function value.

Sample Code

random number example

- Import numpy and sklearn
- Make feature data
- Make label data
- generation and learning and prediction of logistic regression models



```python
import numpy as np
from sklearn.linear_model import LogisticRegression

X_train = np.r_[np.random.normal(3, 1, size = 50),
                np.random.normal(-1, 8, size = 50)].reshape((100, -1))

y_train = np.r_[np.ones(50), np.zeros(50)]

model = LogisticRegression(solver = 'lbfgs')
model.fit(X_train, y_train)
model.predict_proba([[0], [1], [2]])[:, 1]
```

경희대학교
KYUNG HEE UNIVERSITY

About solvers

- For small datasets, ‘liblinear’ is a good choice, where as ‘sag’ and ‘sega’ are faster for large ones.
- For multiclass problems, only ‘newton-cg’, ‘sag’, ‘sega’ and ‘lbfgs’ handle multinomial loss.
- ‘liblinear’ is limited to one-versus-rest schemes.

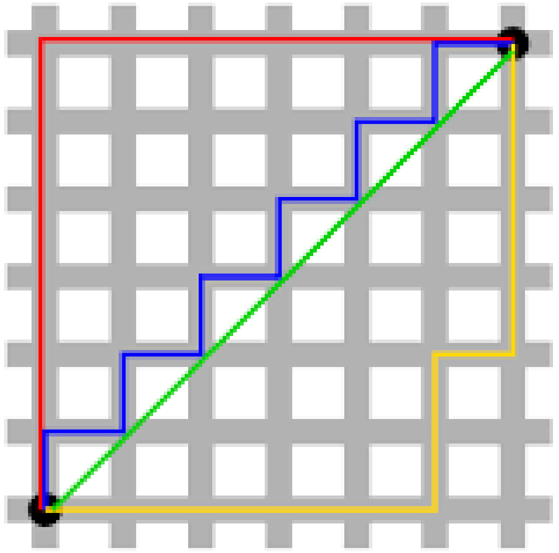
About solvers

	Newton-cg	Lbfgs	Liblinear	Sag	Sega
Penalty term	‘L2’, ‘none’	‘L2’, ‘none’	‘L1’, ‘L2’	‘L2’, ‘none’	‘elasticnet’, ‘L1’, ‘L2’, ‘none’
Dataset size	Does not care	Does not care	Small dataset	Fast at large dataset	Fast at large dataset
class	Multi class Problem	Multi class Problem	Only class Problem	Multi class Problem	Multiclass problem

경희대학교
KYUNG HEE UNIVERSITY

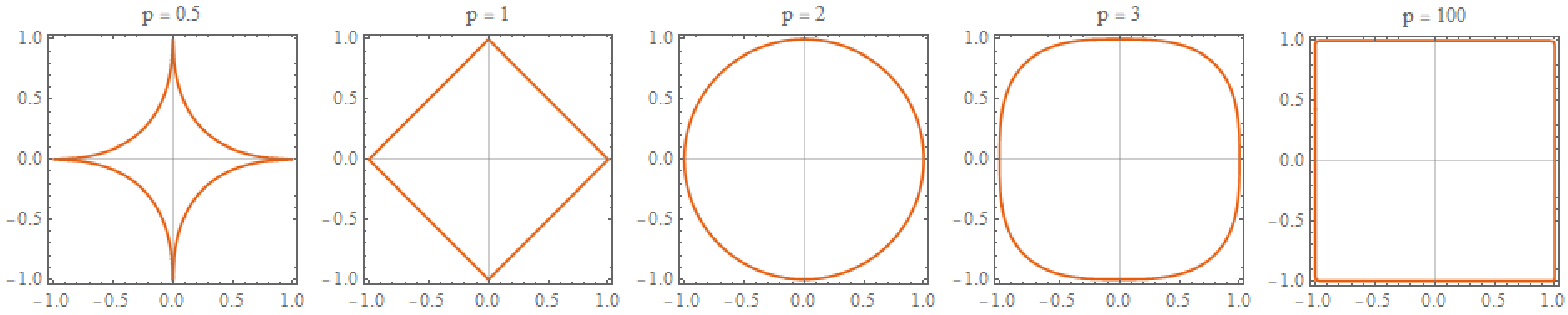
What is $norm$?

- It is norm that the vector tells how large it is.
- norm is a function that sends a set of real numbers $\mathbb{R}$ in a vector space called V.



What is $l_p - norm$?

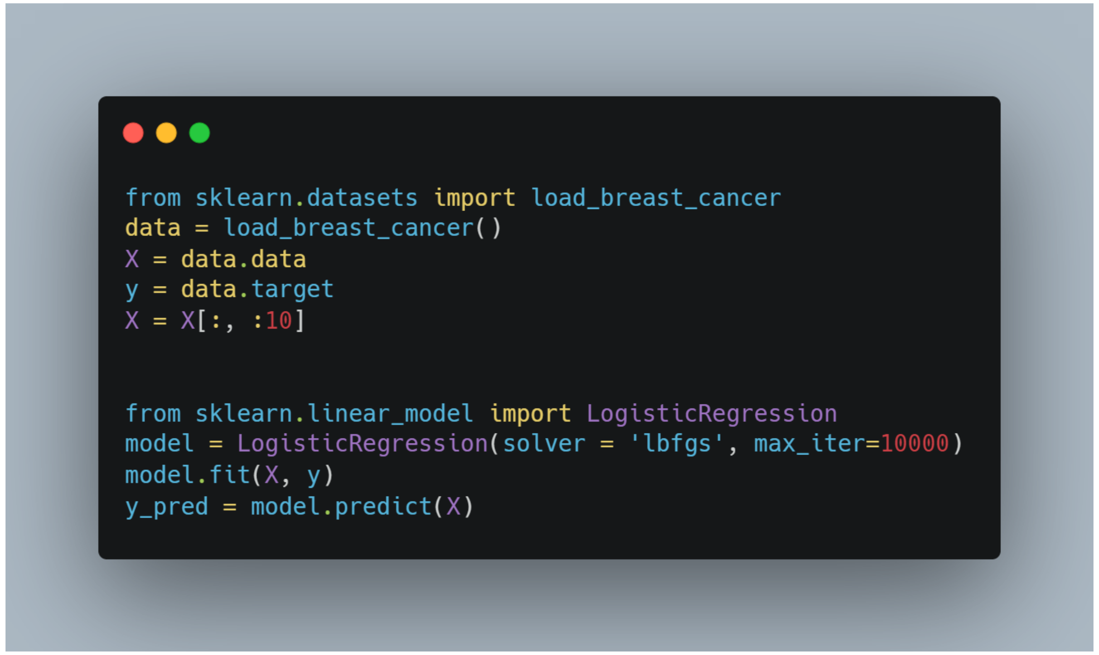
$$||x||_p = (\sum_{i=1}^{n} |x_i|^p)^{\frac{1}{p}}$$



Sample code

breast cancer dataset example

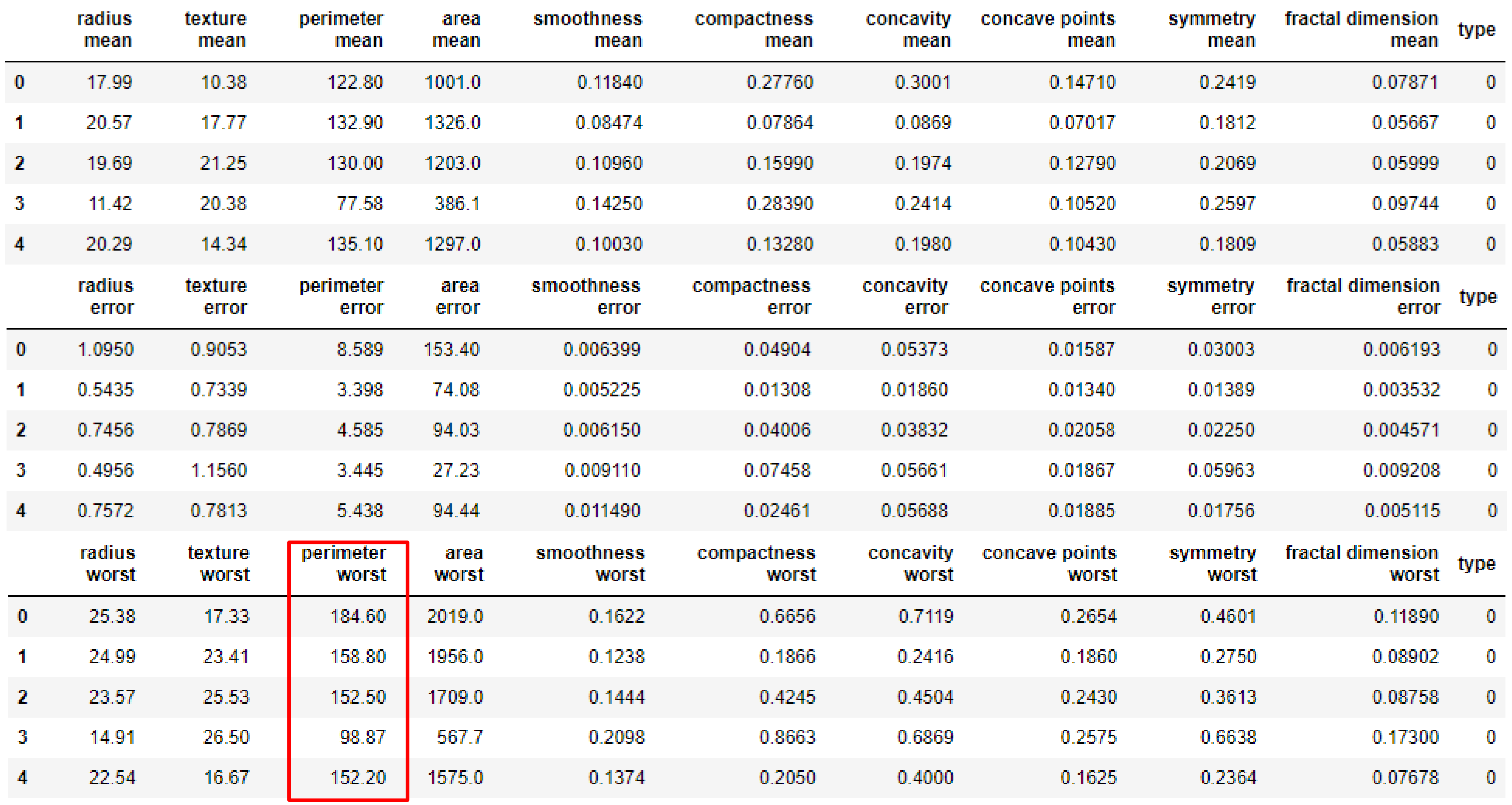
- Load dataset
- Get only the data we want
- Import logistic regression library
- Make logistic regression instance
- Predict the result



```python
from sklearn.datasets import load_breast_cancer
data = load_breast_cancer()
X = data.data
y = data.target
X = X[:, :10]


from sklearn.linear_model import LogisticRegression
model = LogisticRegression(solver = 'lbfgs', max_iter=10000)
model.fit(X, y)
y_pred = model.predict(X)
```

경희대학교
KYUNG HEE UNIVERSITY

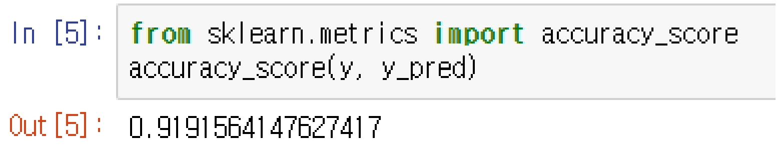
Breast cancer diagnosis dataset table



	radius mean	texture mean	perimeter mean	area mean	smoothness mean	compactness mean	concavity mean	concave points mean	symmetry mean	fractal dimension mean	type
0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	0.07871	0
1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	0.05667	0
2	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	0.05999	0
3	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	0.09744	0
4	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	0.05883	0

	radius error	texture error	perimeter error	area error	smoothness error	compactness error	concavity error	concave points error	symmetry error	fractal dimension error	type
0	1.0950	0.9053	8.589	153.40	0.006399	0.04904	0.05373	0.01587	0.03003	0.006193	0
1	0.5435	0.7339	3.398	74.08	0.005225	0.01308	0.01860	0.01340	0.01389	0.003532	0
2	0.7456	0.7869	4.585	94.03	0.006150	0.04006	0.03832	0.02058	0.02250	0.004571	0
3	0.4956	1.1560	3.445	27.23	0.009110	0.07458	0.05661	0.01867	0.05963	0.009208	0
4	0.7572	0.7813	5.438	94.44	0.011490	0.02461	0.05688	0.01885	0.01756	0.005115	0

	radius worst	texture worst	perimeter worst	area worst	smoothness worst	compactness worst	concavity worst	concave points worst	symmetry worst	fractal dimension worst	type
0	25.38	17.33	184.60	2019.0	0.1622	0.6656	0.7119	0.2654	0.4601	0.11890	0
1	24.99	23.41	158.80	1956.0	0.1238	0.1866	0.2416	0.1860	0.2750	0.08902	0
2	23.57	25.53	152.50	1709.0	0.1444	0.4245	0.4504	0.2430	0.3613	0.08758	0
3	14.91	26.50	98.87	567.7	0.2098	0.8663	0.6869	0.2575	0.6638	0.17300	0
4	22.54	16.67	152.20	1575.0	0.1374	0.2050	0.4000	0.1625	0.2364	0.07678	0

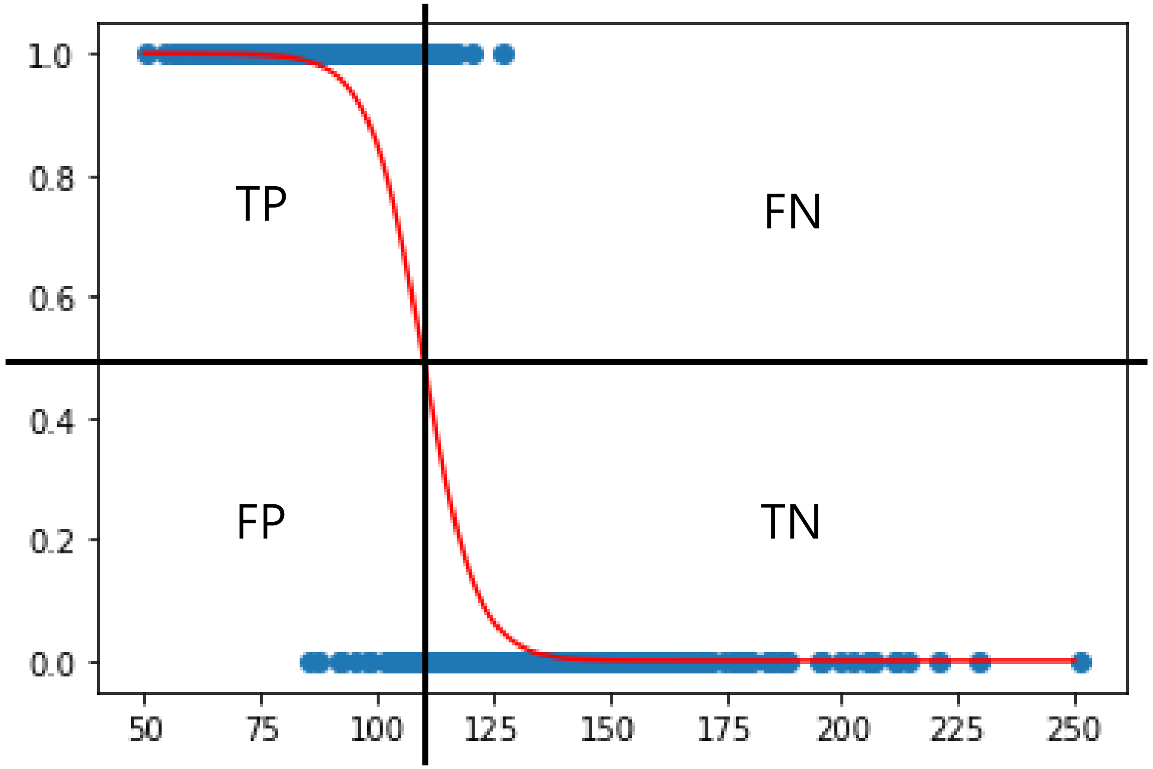


```
In [5]:  from sklearn.metrics import accuracy_score
         accuracy_score(y, y_pred)

Out[5]:  0.9191564147627417
```

$$\frac{TP + TN}{TP + TN + FP + FN}$$

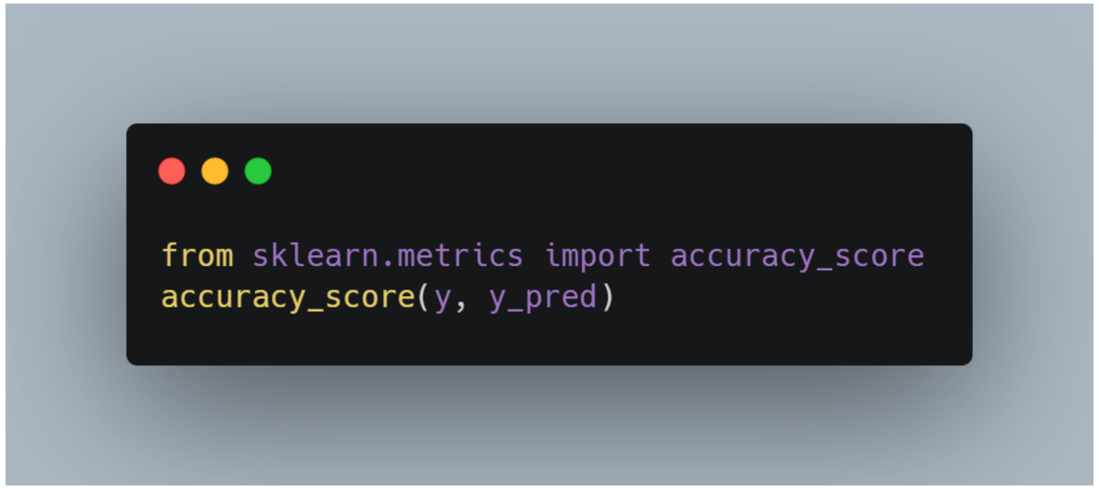
경희대학교
KYUNG HEE UNIVERSITY

Only the perimeter worst value visualization



Accuracy variation according to dataset

- Only the average: 0.9121
- Only the error: 0.8928
- Only the worst: 0.9561
- All of them: 0.9578



```
from sklearn.metrics import accuracy_score
accuracy_score(y, y_pred)
```

Cross Validation according to dataset

- Only the average

```
Out[5]:  array([0.88596491, 0.92105263, 0.92105263, 0.92982456, 0.90265487])
```

- Only the error

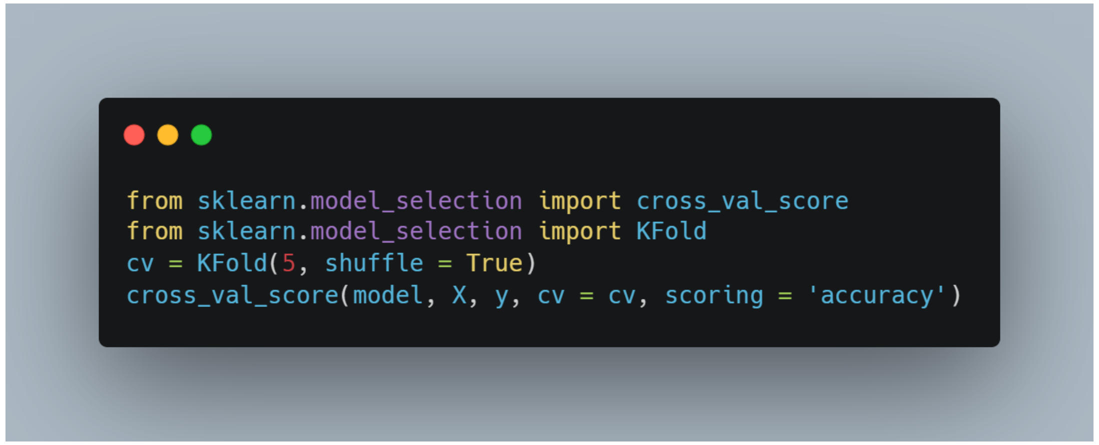
```
Out[5]:  array([0.89473684, 0.87719298, 0.90350877, 0.89473684, 0.85840708])
```

- Only the worst

```
Out[7]:  array([0.96491228, 0.94736842, 0.94736842, 0.93859649, 0.95575221])
```

- All of them

```
Out[9]:  array([0.93859649, 0.97368421, 0.93859649, 0.92105263, 0.99115044])
```

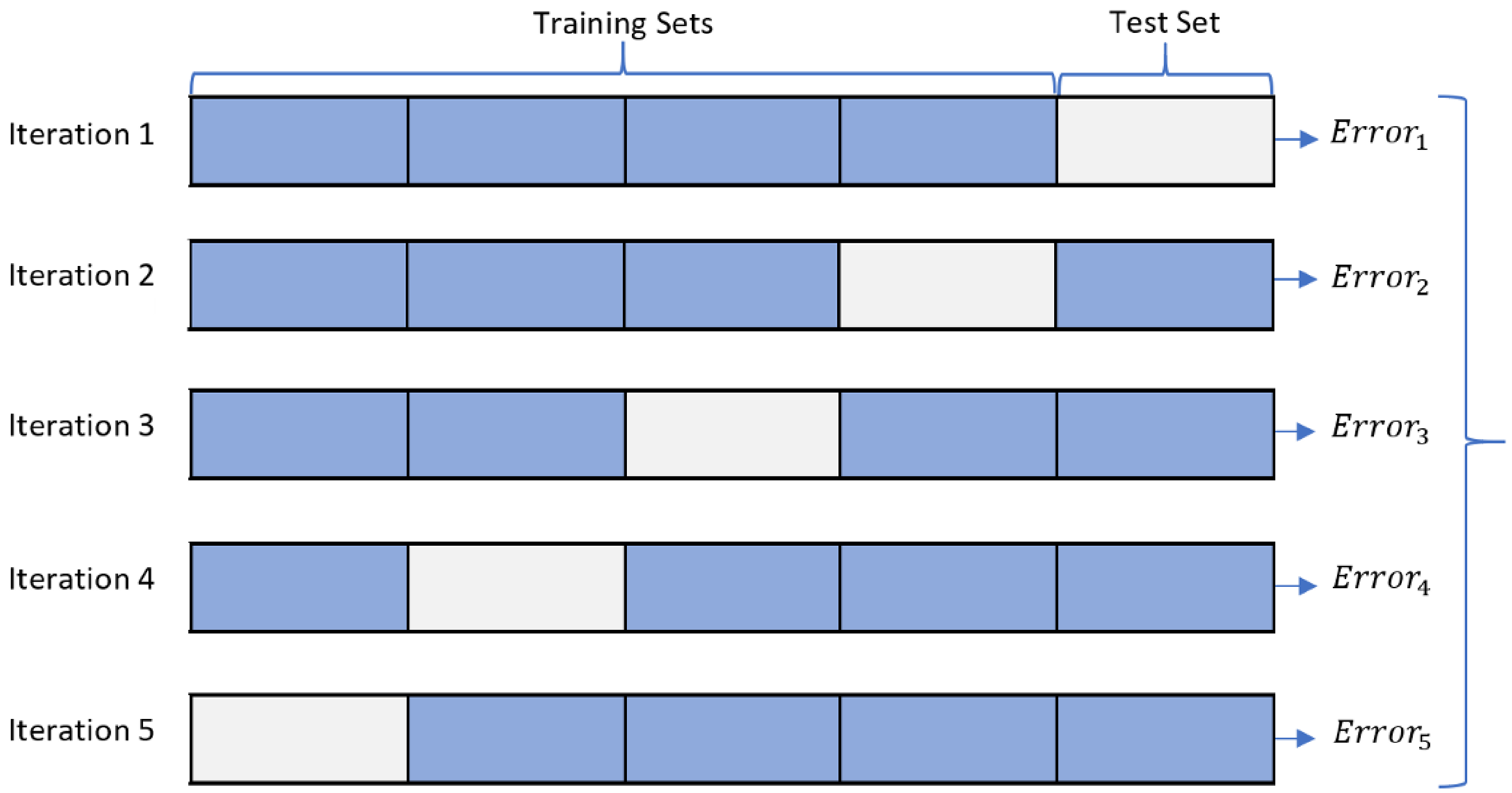


```python
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
cv = KFold(5, shuffle = True)
cross_val_score(model, X, y, cv = cv, scoring = 'accuracy')
```

경희대학교
KYUNG HEE UNIVERSITY

Ways to prevent over-fitting

Cross Validation



$$Error = \frac{1}{5}\sum_{i=1}^{5} Error_i$$

Determination boundary

- When unknown data is put into the model trained to solve the classification problem and classified, the classification result changes to a boundary of some data.
- The boundary in which the classification results change is called the determination boundary.
- In logistic regression, the decision boundary is where the result of calculated probability is 50%.

Supervised Learning

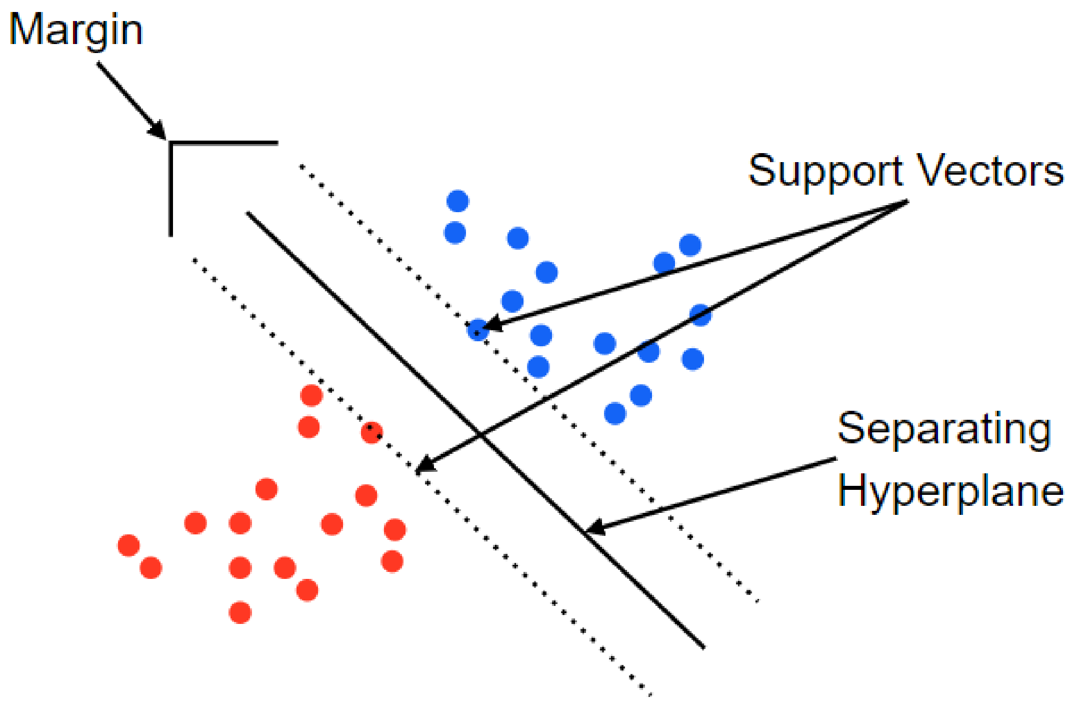
SVM (Support Vector Machine)

경희대학교 컴퓨터공학과
2019102191 신주영

경희대학교
KYUNG HEE UNIVERSITY

Fundamental Concept

- The decision boundaries obtained by SVM maximize margin between groups, so learn to have no points belonging to the boundaries and classify them more clearly than logistic regression.

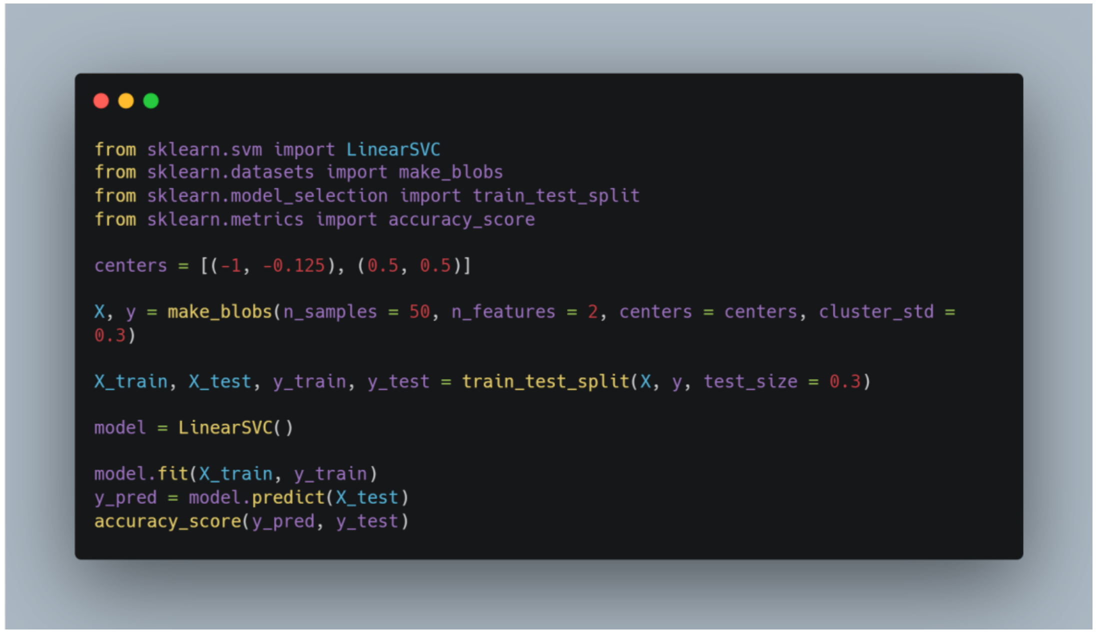


Algorithm

- SVM deals with the problem of binary classification of planes assuming linear and complete classification is possible.
- Maximize the margin between groups to obtain a good decision boundary to classify.

Sample Code

- Import libraries
- Set center coordinate of data
- Make random numbers based on center coordinate
- Make SVM model
- Learning data
- Get accuracy



```
from sklearn.svm import LinearSVC
from sklearn.datasets import make_blobs
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

centers = [(-1, -0.125), (0.5, 0.5)]

X, y = make_blobs(n_samples = 50, n_features = 2, centers = centers, cluster_std =
0.3)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)

model = LinearSVC()

model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy_score(y_pred, y_test)
```

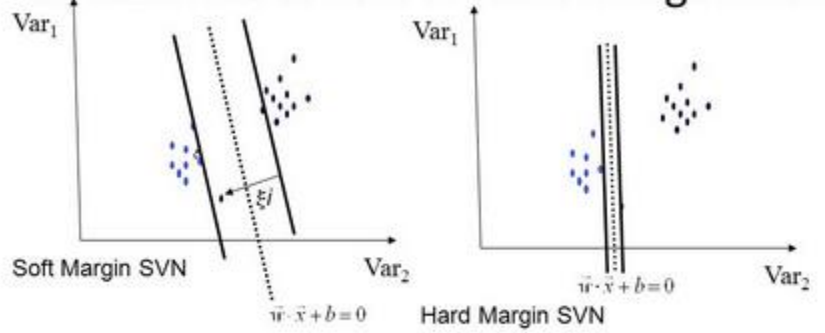
Margin and Support vector

- If the margin does not contain data, it is called hard margin.
- Allowing some data to be contained in margin is called soft margin.
- Learning data is divided into three categories.
    - data outside margin: data outside the margin based on the decision boundary
    - margin data: data for determining margin based on decision boundaries
    - data inside margin: data that are in margin or misclassified based on the boundary of the boundaries

Hard margin and Soft margin

- The left graph using soft margin is not affected by the fact that the learning results are far away.
- The graph on the right using hard margin is greatly influenced by the point far away.
- How much data in the margin is allowed in soft margin follows the hyperparameters set by the person himself.

Robustness of Soft vs Hard Margin SVMs



- Soft margin – underfitting
- Hard margin – overfitting

Trade-off: width of the margin vs. no. of training errors committed by the linear decision boundary

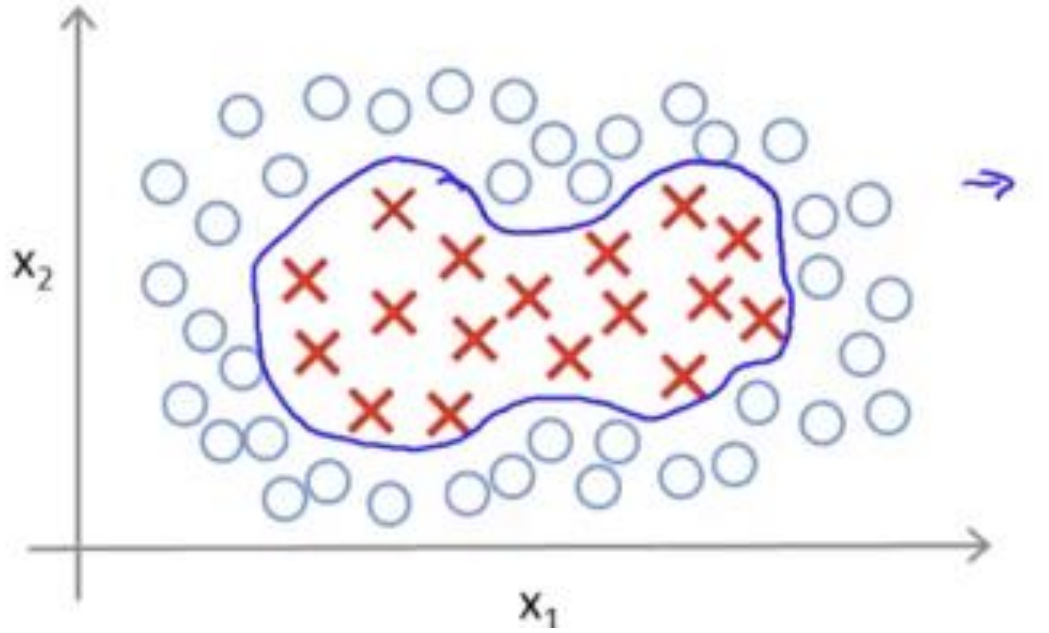
Supervised Learning

SVM(Support Vector Machine) Using Kernel

경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

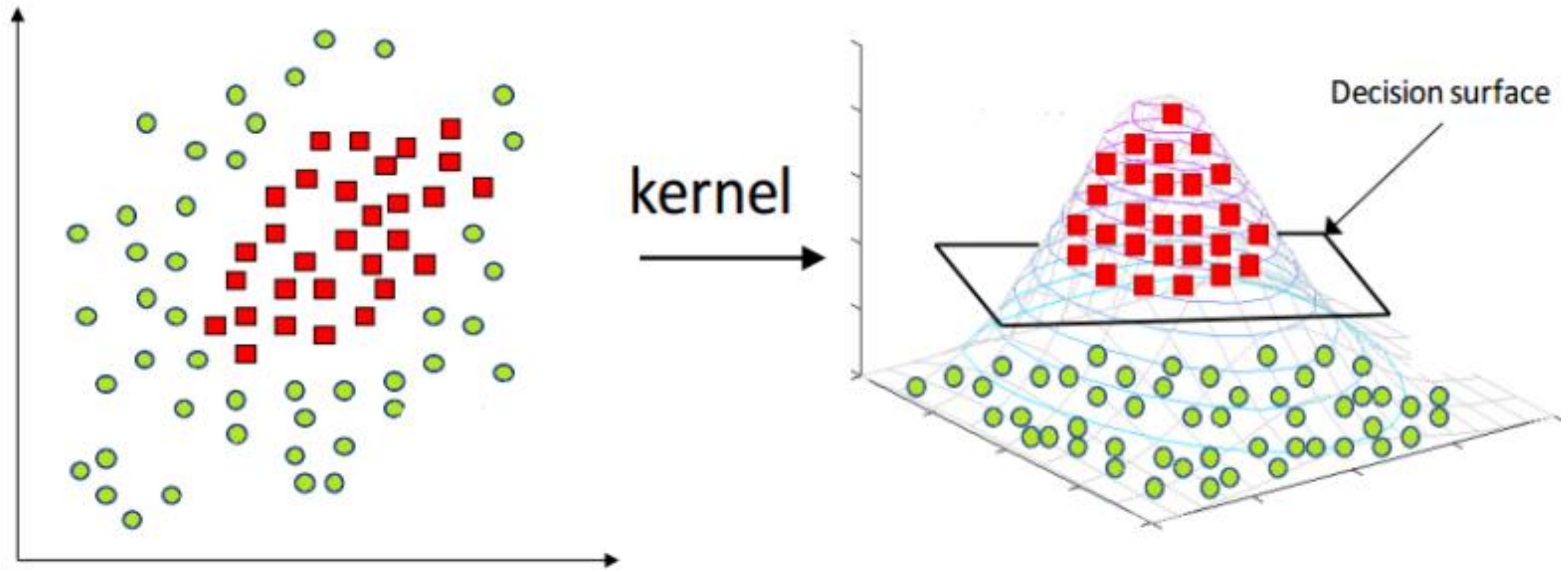
- In SVM, it is difficult to classify data in which the boundary of each label is curved because the decision boundary is a straight line.
- Complex decision boundaries can be learned by applying kernel techniques to SVM.



Algorithm

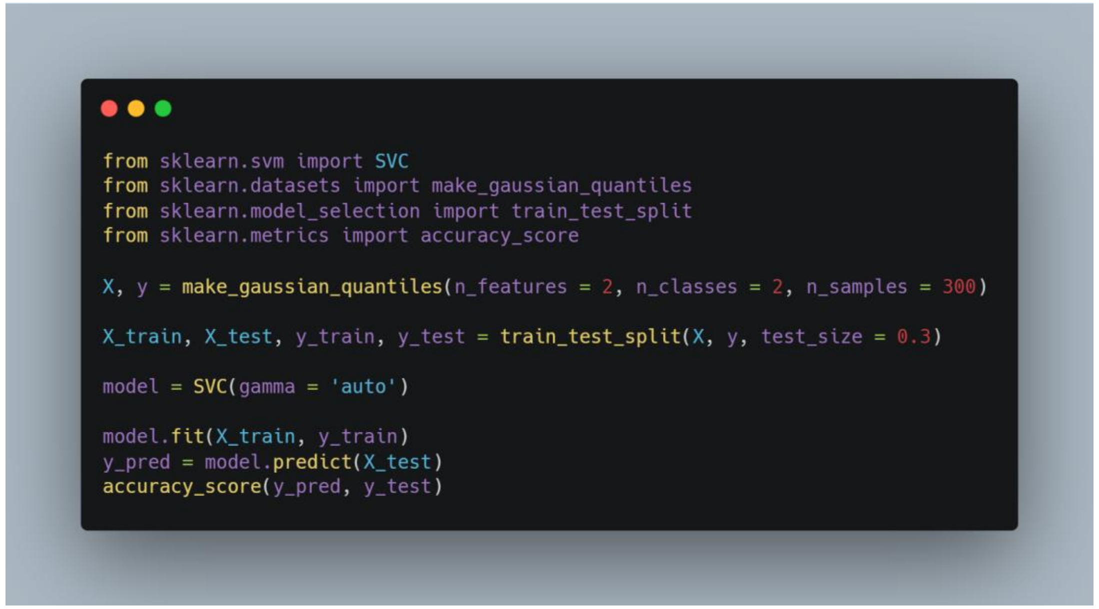
- Kernel technique: linear regression after transferring data to another feature space
- Conditions required to allow linear separation of data that cannot be linearly separated
  - There is a higher dimension space than the learning data, and each point of the learning data corresponds to a point of the higher dimension space.
  - Points corresponding to learning data in a high-dimensional space can be linearly separated, and actual learning data is projection in a high-dimensional space.

Algorithm



Sample Code

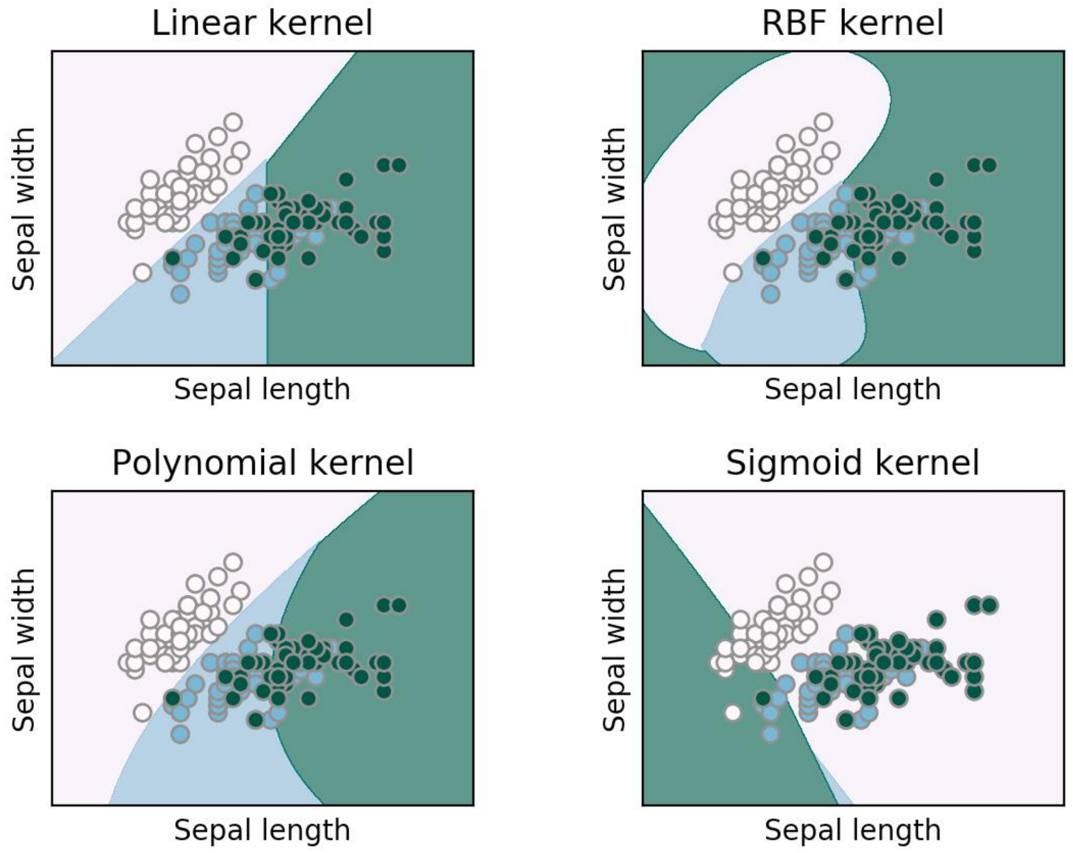
- Import libraries
- Make gaussian quantiles
- Set train, test dataset
- Make SVM model
- Learn and predict data
- Check accuracy



```
from sklearn.svm import SVC
from sklearn.datasets import make_gaussian_quantiles
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

X, y = make_gaussian_quantiles(n_features = 2, n_classes = 2, n_samples = 300)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)

model = SVC(gamma = 'auto')

model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy_score(y_pred, y_test)
```

경희대학교
KYUNG HEE UNIVERSITY

Difference in learning outcomes using kernel functions



Points to note

- The larger the hyperparameter value, the more complex the decision boundary is learned from the data.
- Kernel techniques are recommended to be used to require accuracy rather than feature analysis.
- It should be noted that meaningful decision boundaries are not obtained as a result of learning.

Supervised Learning

Naïve Bayes Classification

경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

- The Naïve Bayes Classification is one of the algorithms for predicting results based on probability.
- Calculate the probability of which label the data belongs and classify the data into the label with the highest probability.
- We're going to use The Naïve Bayes classification going to be used to classify virtual news article titles into '영화' and '우주'.

학습 데이터	카테고리
감동을 준 명작 영화가 부활	영화
화려한 액션 영화가 개봉	영화
명작의 부활에 세계가 감동	영화
모래 폭풍이 화성을 덮다	우주
마침내 화성 탐사 재개	우주
VR로 보는 화성의 모래 폭풍과 감동	우주

검정 데이터	카테고리
부활한 명작에서 보여주는 액션에 감동	??

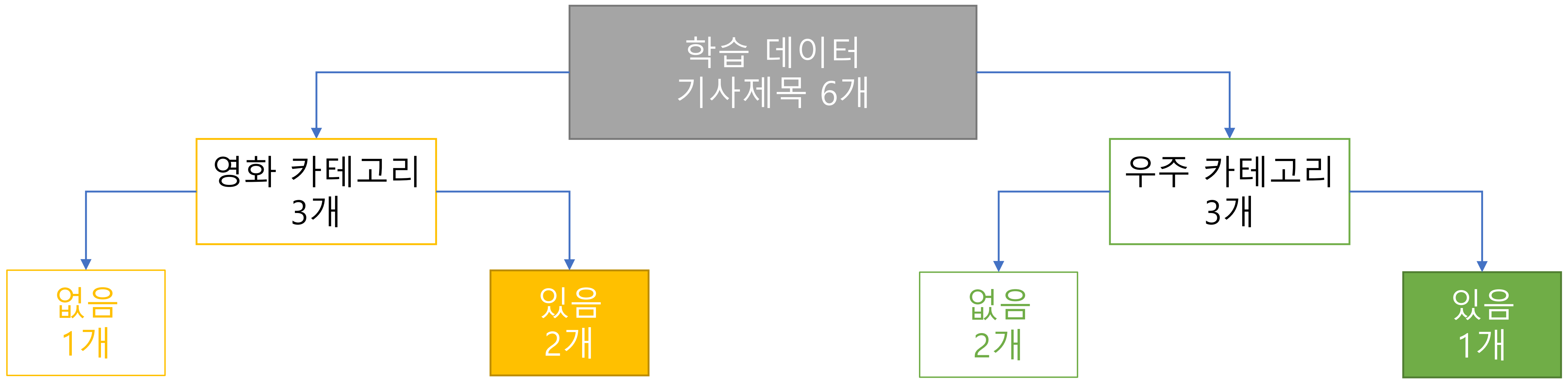
경희대학교
KYUNG HEE UNIVERSITY

Fundamental Concept

- According to the table, there are three categories of articles in the learning data, one for 영화 and one for 우주.
- Simply put, there is a 50 % chance that a title is a movie or a universe.
- We will take an example by paying attention to the word '감동' included in the test data.

경희대학교
KYUNG HEE UNIVERSITY

Fundamental Concept



카테고리	'감동'이 나타날 조건부 확률
영화	2/3
우주	1/3

경희대학교
KYUNG HEE UNIVERSITY

Algorithm

Data Preprocessing

- When dealing with the classification problem of natural language processing into the Naïve Bayes classification, input data must be converted into a vector composed of features.
- The preprocessing in this example is to convert the article title to BoW (Bag of Words) and then create a vector consisting of feature and label pairs.

Algorithm

Data Preprocessing

- First, only nouns are extracted from the article title of the learning data. At this time, the word order is ignored and treated as a set.

학습데이터	카테고리
{"감동", "명작", "영화"}	영화
{"화려", "액션", "영화"}	영화
{"명작", "세계", "감동"}	영화
{"모래 폭풍", "화성"}	우주
{"화성", "탐사", "재개"}	우주
{"VR", "탐사", "재개"}	우주

경희대학교
KYUNG HEE UNIVERSITY

Algorithm

Data Preprocessing

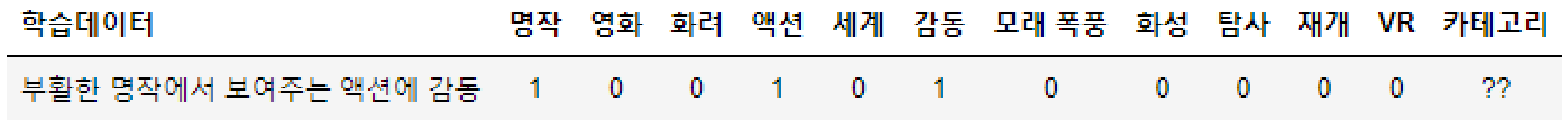
- Second, Change the word set and category of the feature to be easy to handle.
- When the learning data includes words in a set of words, add 1 and 0 when not included.
- As shown in the table above, a pair of features and labels is called BoW depending on whether there is a word in a natural language sentence.

학습데이터	명작	영화	화려	액션	세계	감동	모래 폭풍	화성	탐사	재개	VR	카테고리
감동을 준 명작 영화가 부활	1	1	0	0	0	1	0	0	0	0	0	1
화려한 액션 영화가 개봉	0	1	1	1	0	0	0	0	0	0	0	1
명작의 부활에 세계가 감동	1	0	0	0	1	1	0	0	0	0	0	1
모래 폭풍이 화성을 덮다	0	0	0	0	0	0	1	1	0	0	0	0
마침내 화성 탐사 재개	0	0	0	0	0	0	0	1	1	1	0	0
VR로 보는 화성의 모래 폭풍과 감동	0	0	0	0	0	1	0	1	1	0	1	0

Algorithm

Data Preprocessing

- Third, The test data is also converted to BoW.



학습데이터	명작	영화	화려	액션	세계	감동	모래 폭풍	화성	탐사	재개	VR	카테고리
부활한 명작에서 보여주는 액션에 감동	1	0	0	1	0	1	0	0	0	0	0	??

Algorithm

Calculate Probability

- When learning by the Naïve Bayes classification, two types of probabilities are calculated.
  - a. probability of each label appearing
  - b. conditional probability of each word appearing on each label
  - calculate $a \times b$

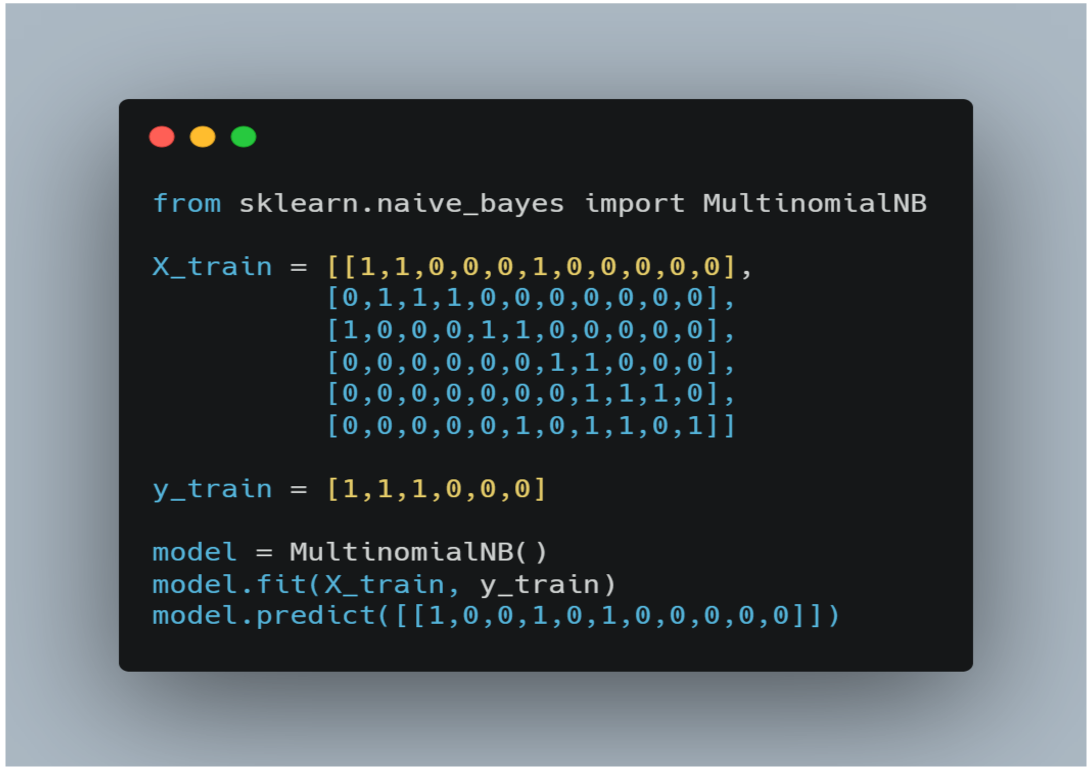
	명작	영화	…	감동	…	VR
영화	2/3	2/3	…	2/3	…	0.01

- The part allocated 0.01 is the part allocated with the original probability 0, which means that the word is not in the learning data.

경희대학교
KYUNG HEE UNIVERSITY

Sample Code

- Import libraries
- Make X, y data
- Make model and training
- Predict result



```python
from sklearn.naive_bayes import MultinomialNB

X_train = [[1,1,0,0,0,1,0,0,0,0,0],
           [0,1,1,1,0,0,0,0,0,0,0],
           [1,0,0,0,1,1,0,0,0,0,0],
           [0,0,0,0,0,0,1,1,0,0,0],
           [0,0,0,0,0,0,0,1,1,1,0],
           [0,0,0,0,0,1,0,1,1,0,1]]

y_train = [1,1,1,0,0,0]

model = MultinomialNB()
model.fit(X_train, y_train)
model.predict([[1,0,0,1,0,1,0,0,0,0,0]])
```

```
Out[2]:  array([1])
```

Supervised Learning

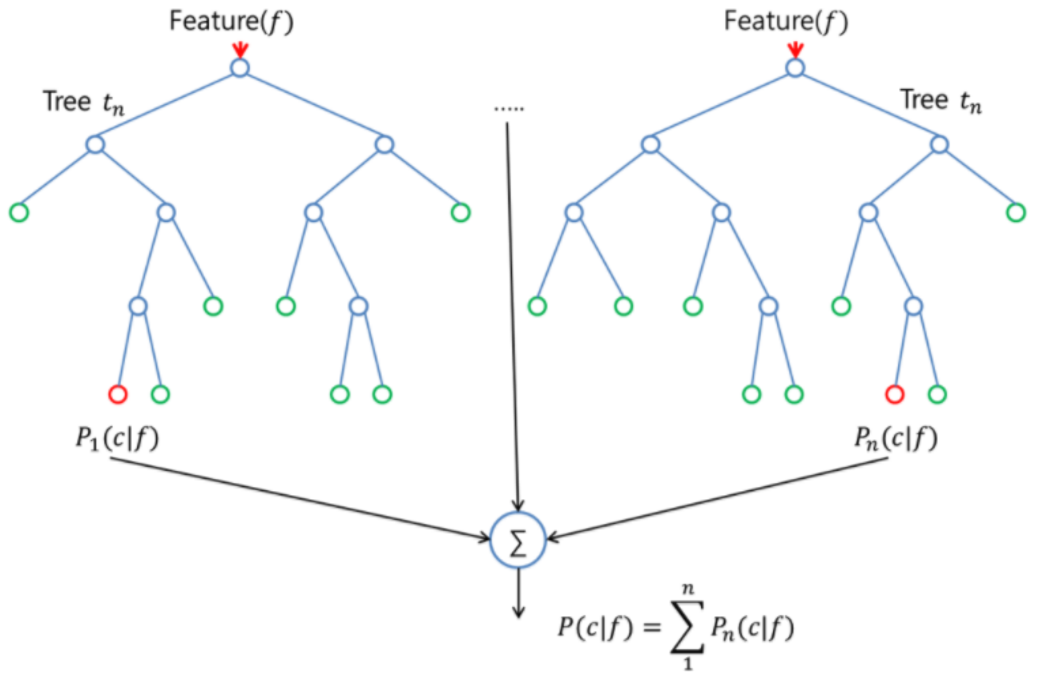
Random Forest

경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

- Random forest is a method of combining various models to create a high-performance model.
- It can be used for both regression and classification problems.
- Similar algorithms include gradient boosting.
- Random forest uses multiple models called decision trees to increase accuracy than using a single decision tree.
- Random forest is used for both classification and regression problems, but this section exemplifies the classification problem.

Fundamental Concept



- The most predictive results are obtained as the final classification result by collecting the predictive results from each decision tree.
- Random forest should be different for each decision tree.

Algorithm

Decision Tree

- When dividing learning data, impurity, which represents the imbalance of data, is digitized and used.
- The decision tree divides the data so that the impurity is reduced.
- If there are many of the same labels in the divided group, the impurity decreases.
- If there are many other labels in the divided group, the impurity increases.
- There are several indicators of impurities, where Gini index will be used.
- Gini index: $-\sum_{i=1}^{c} p_i^2$

경희대학교 KYUNG HEE UNIVERSITY

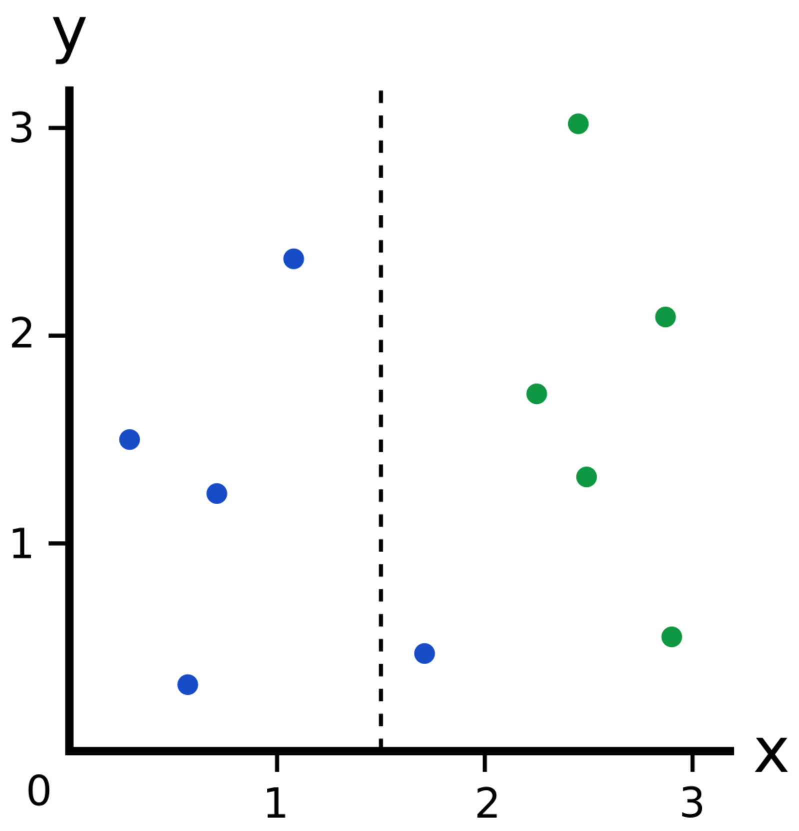
Algorithm

Decision Tree

- In this case, c is the number of labels and p_i is the number of labels divided by the number of data.
- The Gini coefficient and the weighted average using it are calculated according to the method of dividing the data.
    1. Calculate all features of one area and the impurity of the object to be divided
    2. Divide areas to minimize impurities.
    3. After dividing, repeat 1~2 steps in each area.

Algorithm

Decision Tree



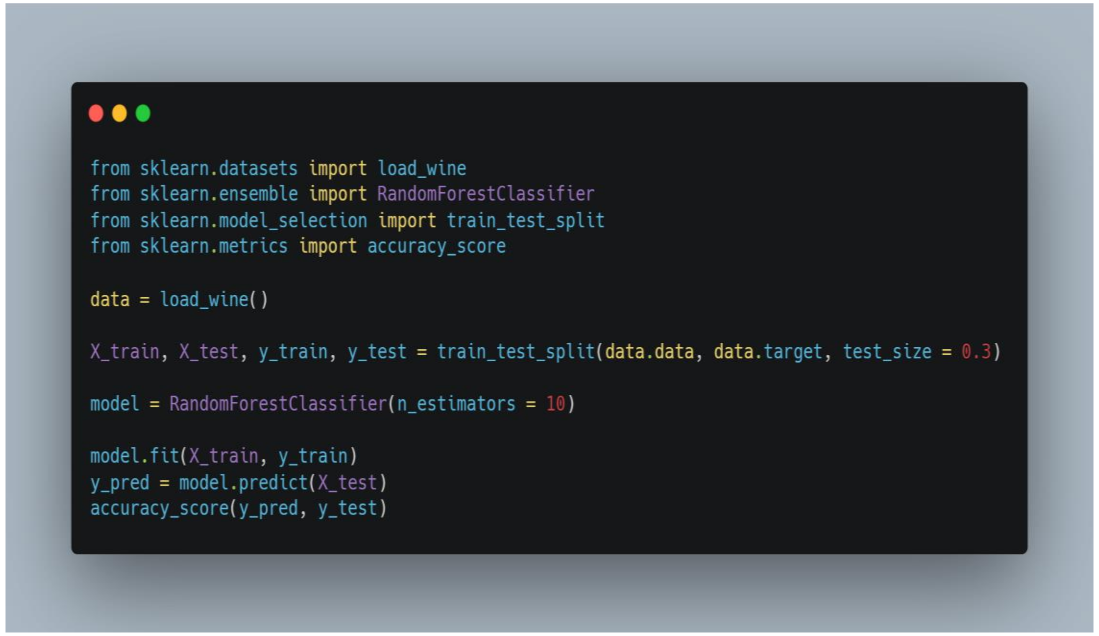
Algorithm
random forest

- Bootstrap: A method of "increasing" learning data by performing random recovery extractions from learning data several times. In one learning data, several learning data with different contents are generated and the learning data placed in each decision tree is different.
- Certain optional choices: Choose and use only a few features randomly when learning the decision tree with bootstrap learning data.
- Random forest learns to have different classification results using the above method.

경희대학교
KYUNG HEE UNIVERSITY

Sample Code

- Import libraries
- Make dataset
- Set train and test data
- Make Random Forest model
- Train data into model
- Calculate accuracy



```python
from sklearn.datasets import load_wine
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

data = load_wine()

X_train, X_test, y_train, y_test = train_test_split(data.data, data.target, test_size = 0.3)

model = RandomForestClassifier(n_estimators = 10)

model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy_score(y_pred, y_test)
```

```
Out[2]:  0.9444444444444444
```

경희대학교
KYUNG HEE UNIVERSITY

Importance of feature

- Feature of high importance are expected to significantly reduce impurities by using them as criteria for dividing areas.
- The feature of low importance is that it is difficult to reduce impurities even when used as a criterion for dividing areas.
- In this way, attention should be paid to the importance of features and unnecessary features should be reduced.

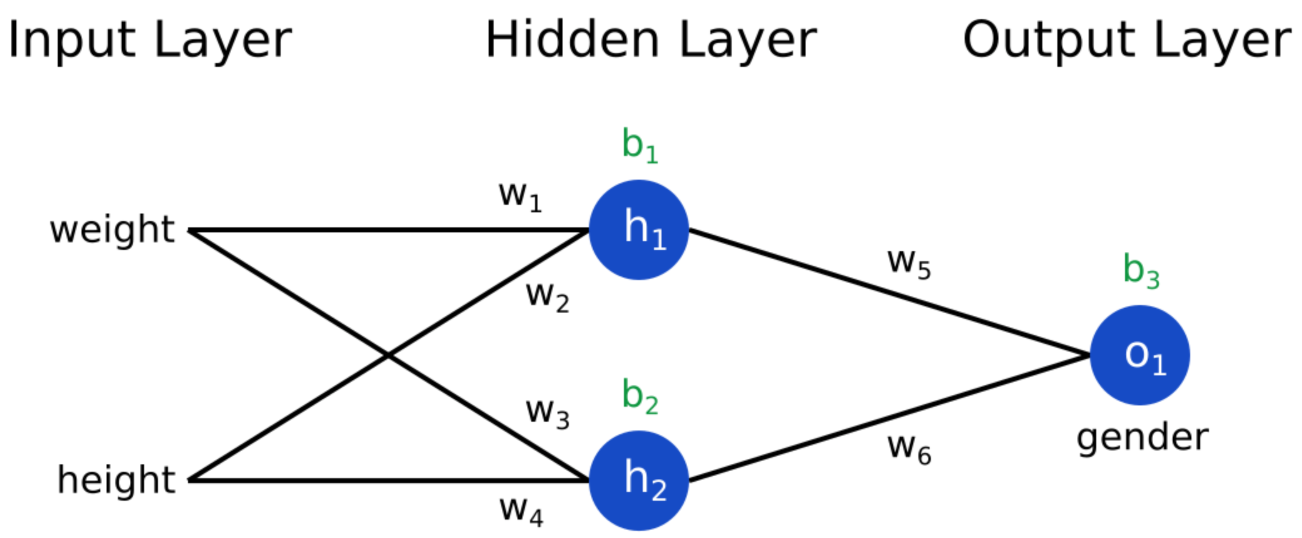
Supervised Learning

Neural Network

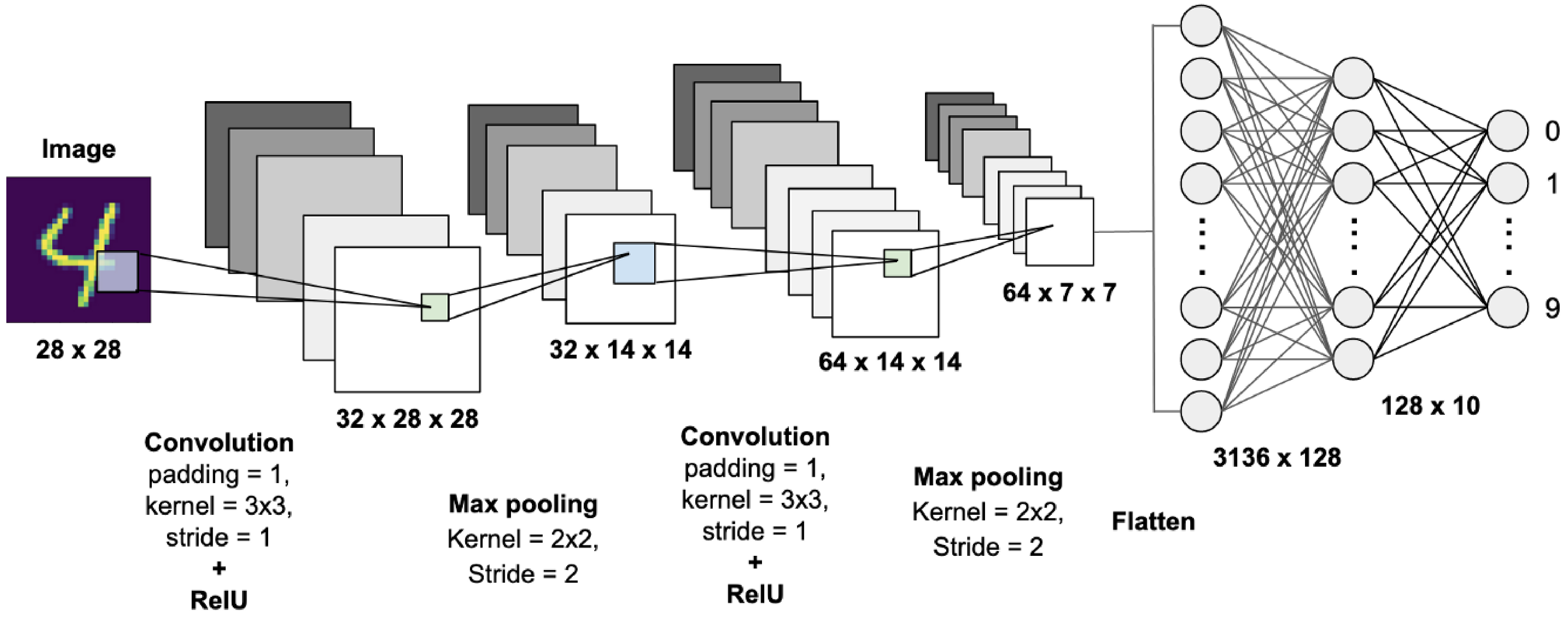
경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

- A neural network is a model that learns complex decision lines by defining a hidden layer between the input layer and the output layer
- It can be applied to both regression and classification, but it is mainly applied to classification problems.
- The above figure is a neural network in which the input layer is 2D, the hidden layer is 2D, and the output layer is 1D.



Fundamental Concept



경희대학교
KYUNG HEE UNIVERSITY

Algorithm

fault perceptron

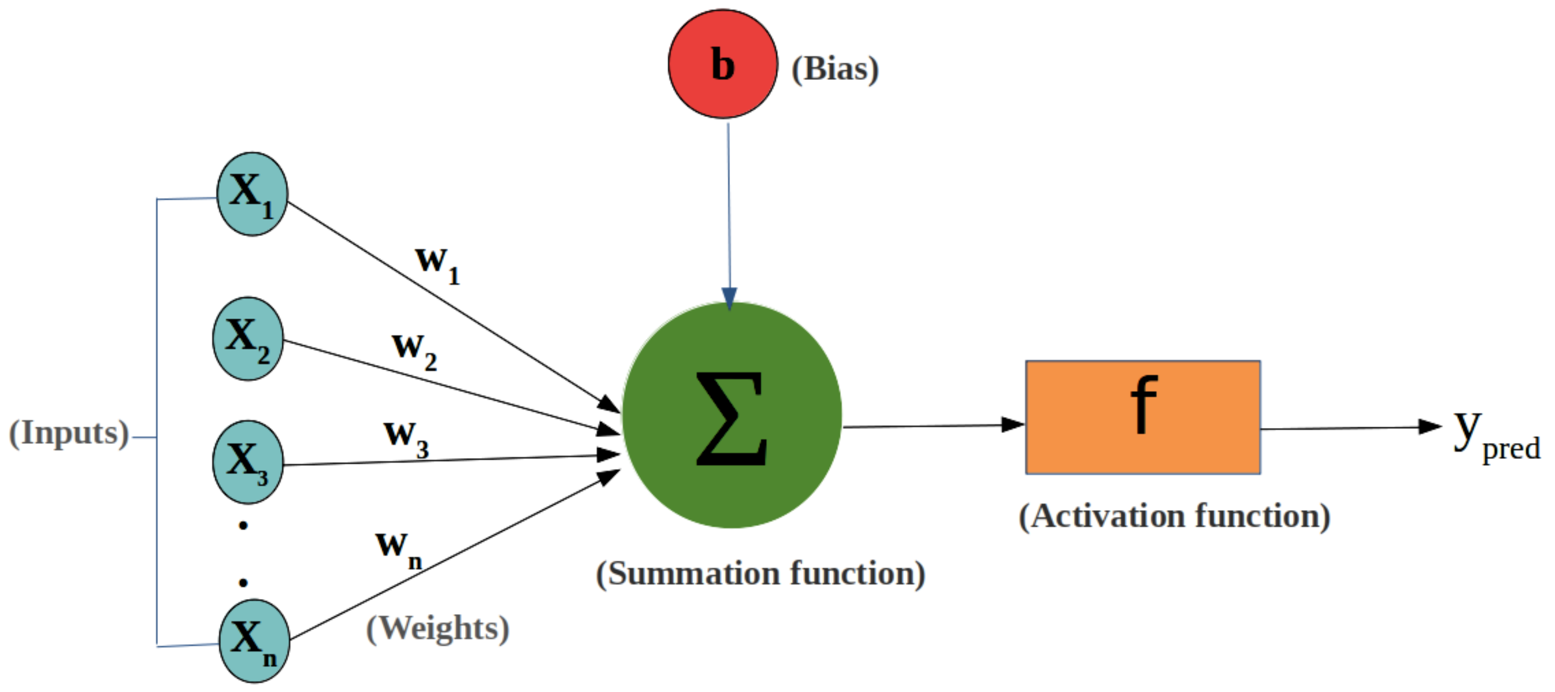
- After applying a neural network to classifying NIST, a dataset composed of cursive letters, we will check the classification results.
- NIST has 10 cursive letters ranging from 0 to 9.
- The input layer represents the input cursive numeric image itself.
- The pixel values of each point are stored in a 1D column vector.
- The hidden layer calculates a value received from the input layer with a nonlinear function such as a sigmoid function.
- The larger the hyperparameters, the more complex decision lines are learned, but overfitting is likely to occur.
- The output layer also calculates the value received from the hidden layer as a nonlinear function.
- The output layer outputs 10 probabilities that the input cursive numerical image belongs to one of 0 to 9.

Algorithm
fault perceptron

- A fault perceptron is a model that identifies data by applying a nonlinear function to a result obtained by multiplying a feature by a weight.
- For example, if feature is 2D, input feature is $(x_1,\ x_2)$.
- $y = f(w_0 + w_1x_1 + w_2x_2)$
- we call w_1, w_2 as a weight, w_0 as a bias.
- Weight and bias are parameters used for learning.
- non-linear function f is called activation function
- The probability y is calculated using the sum of the weights and features. In this case, the activation function uses a sigmoid function or the like.

Algorithm

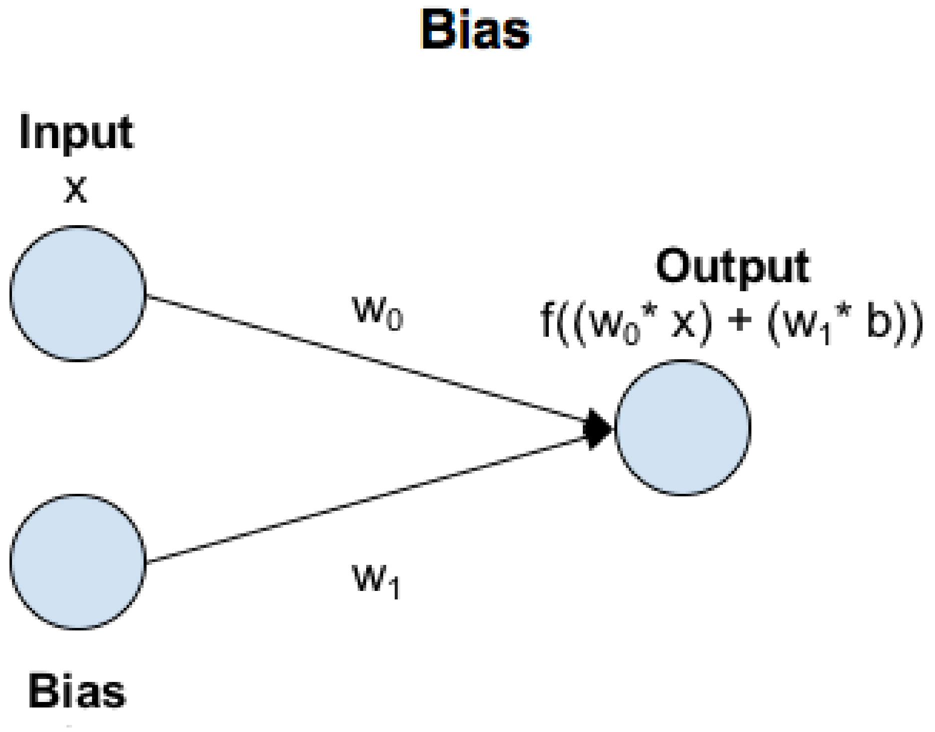
fault perceptron



Algorithm

fault perceptron

- It also simply shows only the input and output parts.
- Fault perceptron have similar characteristics to logistic regression.
- If activation function f is only sigmoid function, fault perception is same as logistic regression



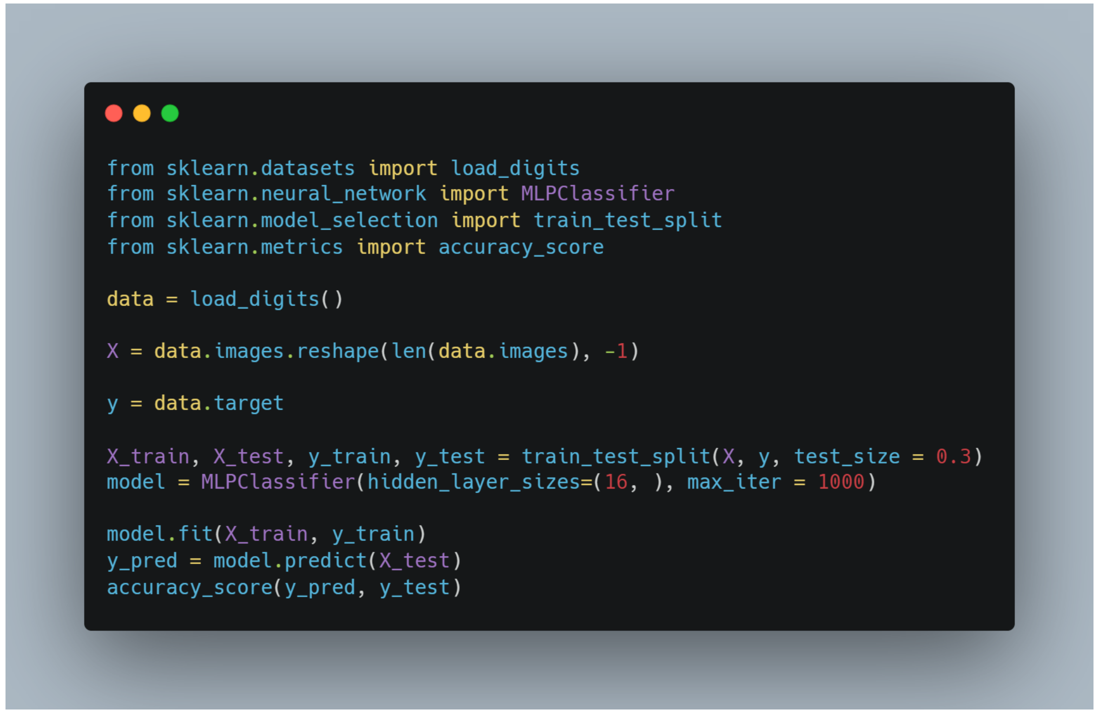
Algorithm
Neural Network

- A neural network is a model that defines several fault perceptron to represent complex decision line.
- fault perceptron can't learn decision line correctly.

1. Set a hidden layer between the input and output.
2. Set the decision to make the final decision by combining the output results of the two layers.

Sample Code

- Import libraries
- Load data
- Reshape images
- Set label data
- Set training, test data
- put training data to model
- Predict result
- Check accuracy

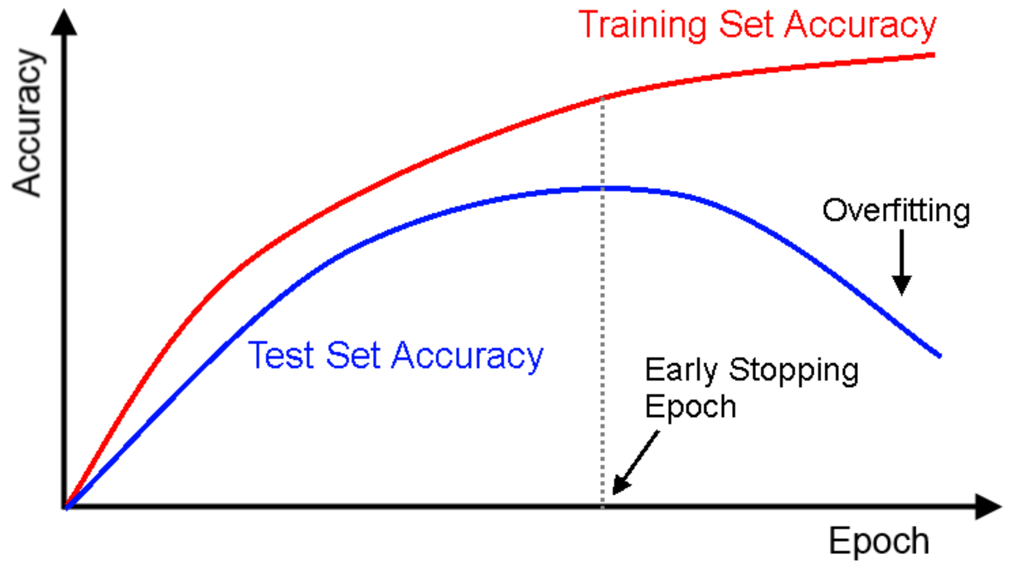


```
from sklearn.datasets import load_digits
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

data = load_digits()

X = data.images.reshape(len(data.images), -1)

y = data.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)
model = MLPClassifier(hidden_layer_sizes=(16, ), max_iter = 1000)

model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy_score(y_pred, y_test)
```

경희대학교 KYUNG HEE UNIVERSITY

Early stopping

- Some of the learning data is further divided into evaluation data used during learning.
- It stores evaluation indicators such as loss in order with evaluation data used during learning so that the degree of learning progress can be known.
- If there are signs of overfitting during learning, the learning is terminated in the middle.



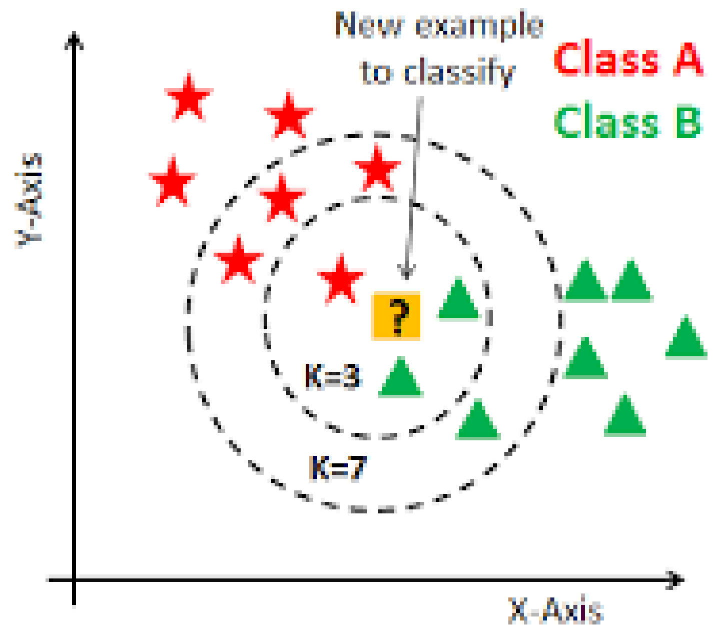
Supervised Learning

kNN (k-Nearest Neighbors Algorithm)

경희대학교 컴퓨터공학과
2019102191 신주영

Fundamental Concept

- kNN can be applied to both classification and regression problems.
- When classifying unknown data, the distance to the learning data is calculated to determine how the data is classified up to k points nearby, and then the data is classified by majority vote 2.



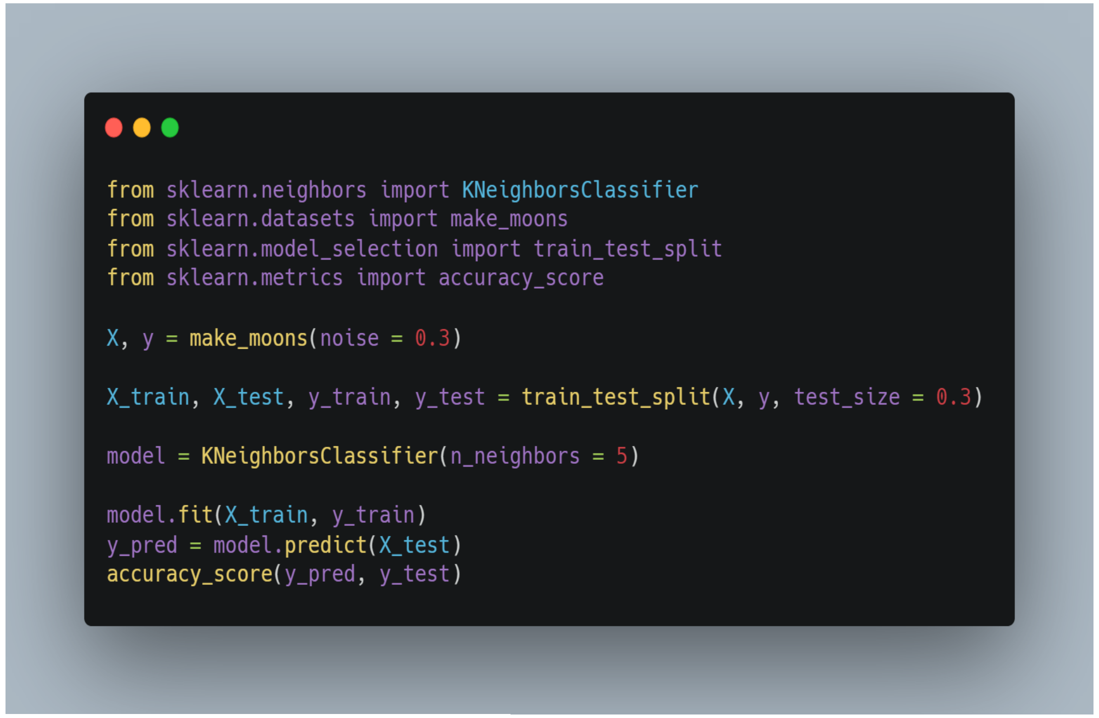
출처: 데이터 캠프

Algorithm

1. calculating the distance of learning data from input data
2. Search for k learning data close to input data
3. Categorize input data that is not known by majority vote in the label of learning data.

Sample Code

- Import libraries
- Make sample data
- Set training, test data
- Train data
- Predict result
- Calculate accuracy



```python
from sklearn.neighbors import KNeighborsClassifier
from sklearn.datasets import make_moons
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

X, y = make_moons(noise = 0.3)

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)

model = KNeighborsClassifier(n_neighbors = 5)

model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy_score(y_pred, y_test)
```

```
Out[1]:  0.8666666666666667
```

경희대학교
KYUNG HEE UNIVERSITY

Difference in decision boundary according to k value

- The smaller the k value, the more likely over-fitting occurs.
- Conversely, if the value of k is too large, the decision boundary cannot be accurately learned.

Points to note

- The kNN is basically not suitable for large amounts of data processing.
- High-level data is difficult to learn. This is because asymptotic assumptions of the kNN algorithm are not necessarily established in high-dimensional data.