THE HONG KONG
POLYTECHNIC UNIVERSITY
香港理工大學

AMA 2233

Data Analytics and Visualization

Yancheng Yuan

Department of Data Science and Artificial Intelligence

Opening Minds • Shaping the Future
啟迪思維 • 成就未來

Recap of last lecture

Linear Regression

Linear Regression using a weighted sum of the independent variables as a predictor to approximate the independent variable.

$$y \approx h(x) = w_0 + w_1 x_1 + \cdots + w_n x_n.$$

If we denote

$$x = (1, x_1, \ldots, x_n)^T \text{ and } w = (w_0, w_1, \ldots, w_n)^T,$$

we can write

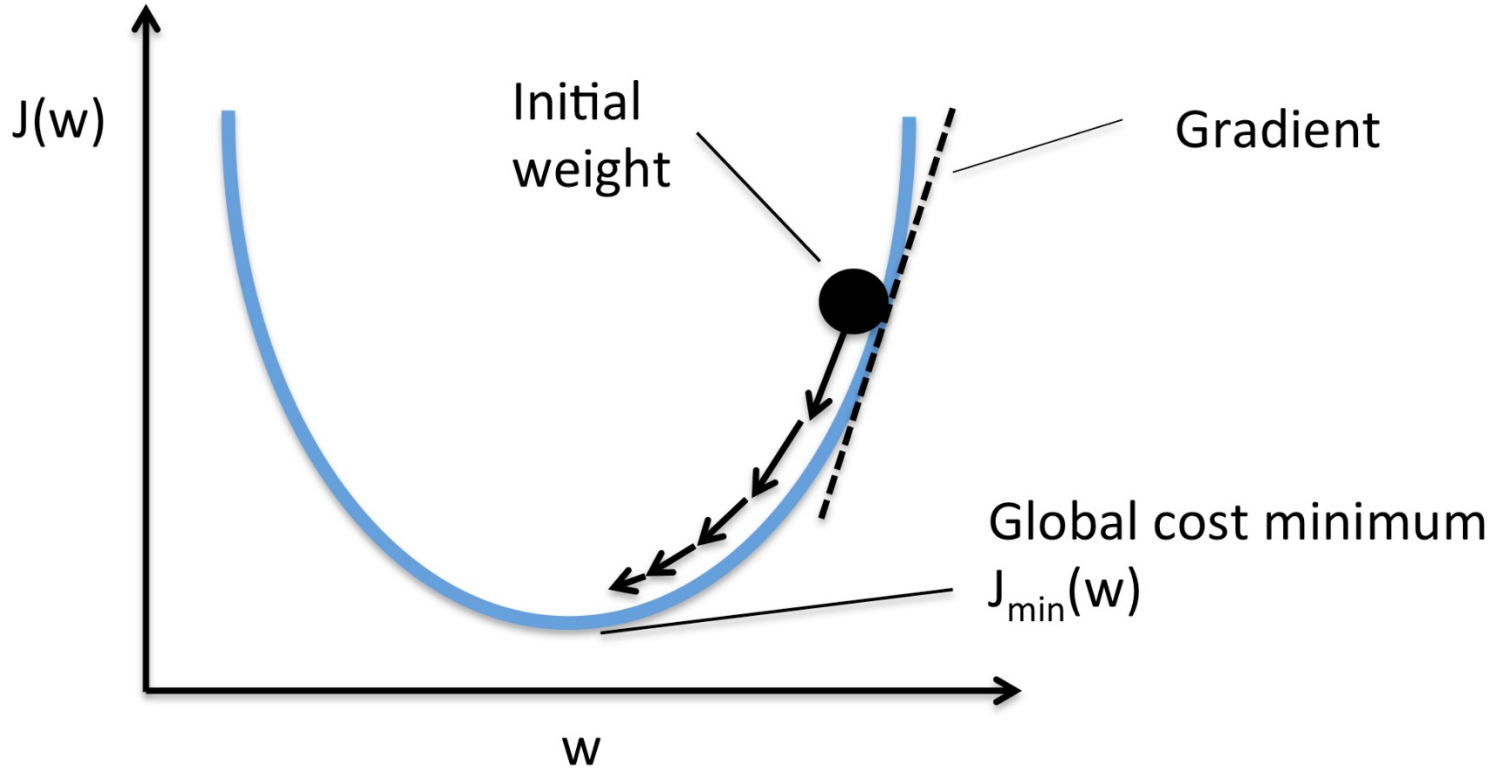
$$h(x) = w^T x.$$

How to Get a Better Model? (Error Measure)

Mean Squared Error (MSE): If we have a data set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, the mean squared error of the linear predictor on the data set is defined as

$$J(w) = \frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} - h(x^{(i)})\right)^2$$

How to Get a Better Model? (Model Training)



Gradient Descent Algorithm

Start from some $w^0 \in \mathbb{R}^{n+1}$, gradient descent (GD) algorithm updates as:

$$w^{k+1} = w^k - \alpha_k \nabla J(w^k),$$

until

$$||\nabla J(w^{k+1})|| \leq \varepsilon,$$

for some tolerance $\varepsilon > 0$.

Key points:

1. Compute $\boldsymbol{\nabla J(w^k)}$.
2. Choose step size $\alpha_k > 0$ satisfying

$$\mathbf{J}(\boldsymbol{w^{k+1}}) < \boldsymbol{J(w^k)}.$$

Compute the Gradient of J(w)

Denote

$$X = \left(x^{(1)}, x^{(2)}, \ldots, x^{(m)}\right)^T \text{ and } y = \left(y^{(1)}, y^{(2)}, \ldots, y^{(m)}\right)^T,$$

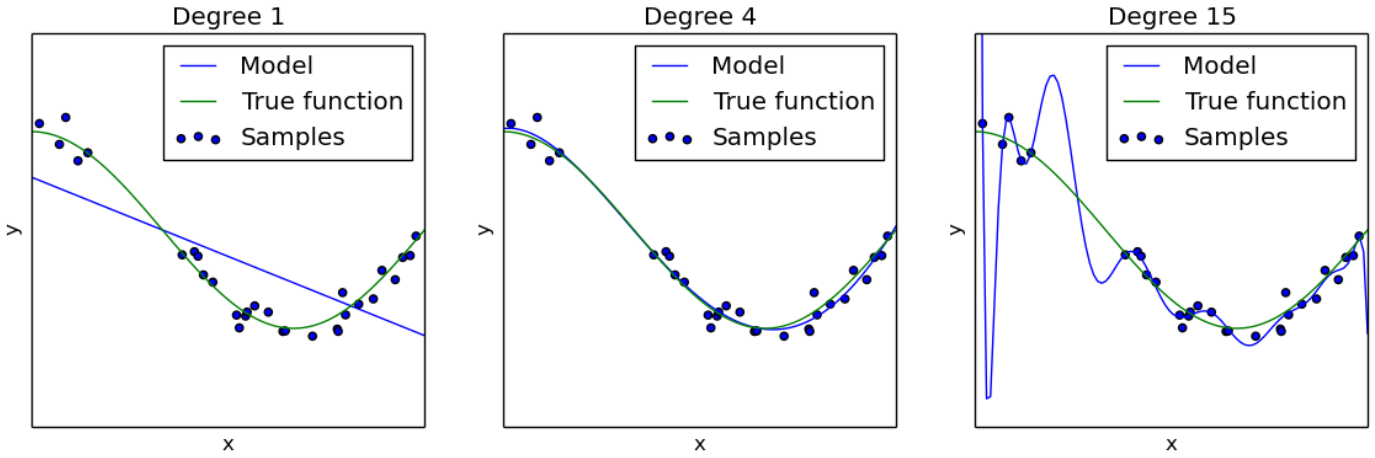
we can rewrite

$$J(w) = \frac{1}{m}||y - Xw||^2$$

Gradient of J(w):

$$\nabla J(w) = \frac{2}{m} X^T (Xw - y)$$

Overfitting



Regularization

A potential issue for the model overfitting is that the model is **too complicated**. One idea to prevent overfitting is to reduce the model complexity by adding regularization.

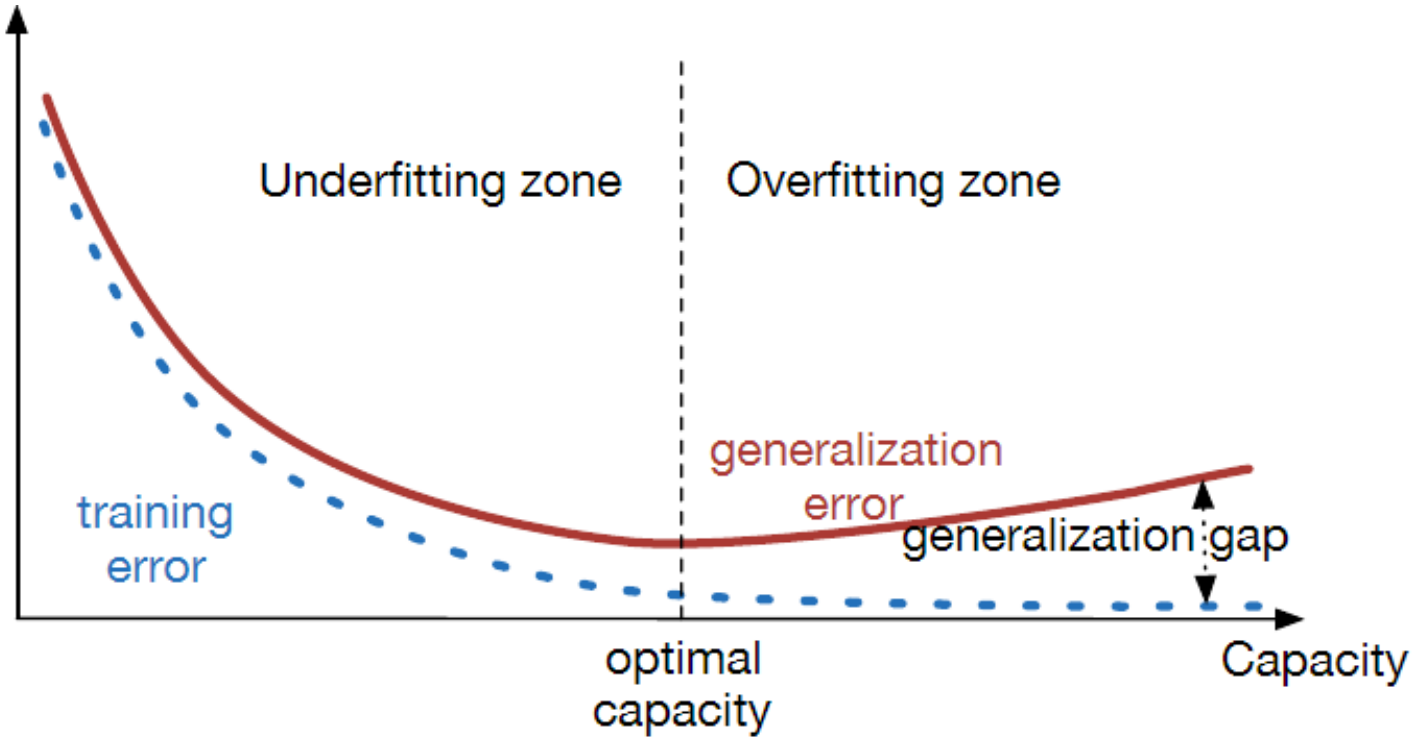
L2 Regularization (Ridge Regression)

$$J(\boldsymbol{w}) + \frac{\boldsymbol{\lambda}}{\boldsymbol{2}} ||\boldsymbol{w}||^{\boldsymbol{2}}$$

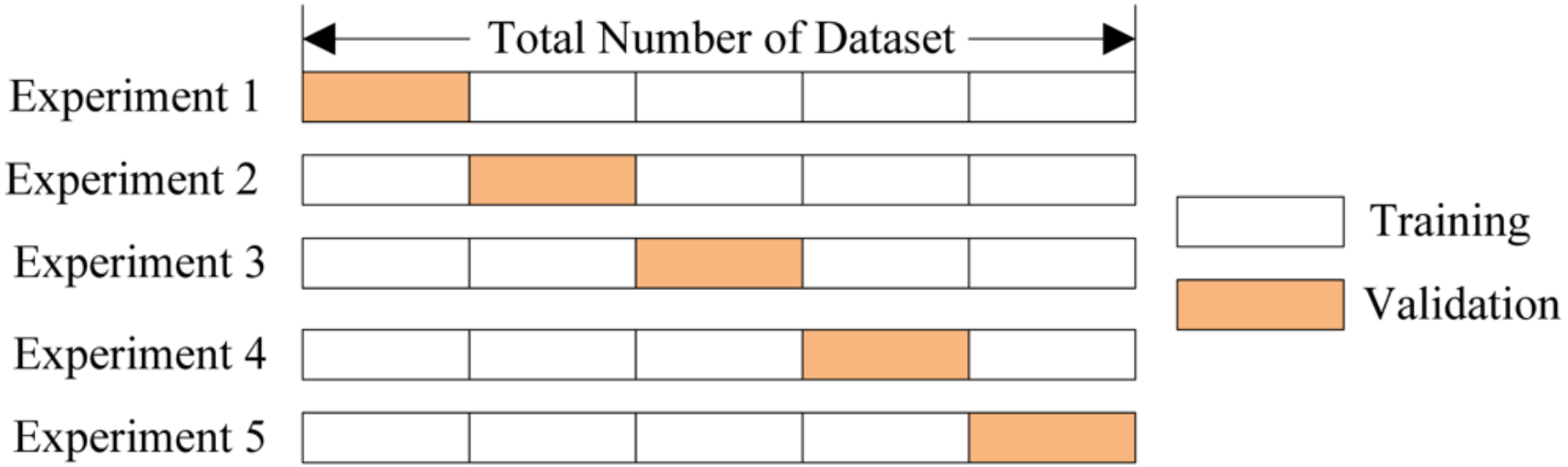
L1 Regularization (Lasso Regression)

$$J(\boldsymbol{w}) + \boldsymbol{\lambda}||\boldsymbol{w}||_{\boldsymbol{1}}$$

Regularization: Early Stopping



Model Selection: Cross Validation



Agenda



1. Linear Regression: Probability Interpretation
2. Logistic Regression
3. Application: Spam Email Classification

Linear Regression: Probability Interpretation

Problem Settings

Problem settings: We assume that the data is generated by

$$y = w^T x + \epsilon,$$

where $\epsilon \sim N(0, \sigma^2)$ is the noise, in other words,

$$p(\epsilon) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\epsilon^2 / 2\sigma^2)$$

Remark: The noise $\epsilon \sim N(0, \sigma^2)$ captures either unmodeled effects or random noise.

Likelihood Function

Likelihood function: For a given data set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, the likelihood function is given by

$$L(w) = \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{\left(y^{(i)} - w^T x^{(i)}\right)^2}{2\sigma^2}\right) = \frac{1}{\left(\sqrt{2\pi}\sigma\right)^2} \exp\left(-||y - Xw||^2 / 2\sigma^2\right)$$

Maximize Log Likelihood Function

Log Likelihood function: For a given data set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, the log likelihood function is given by

$$l(w) = \log(L(w)) = nlog\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{1}{2\sigma^2}||y - Xw||^2$$

Linear regression: For a given data set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, we find the best model parameter w by maximizing the log likelihood function, or equivalently, minimize $-l(w)$ (modify the constant):

$$w^* = argmin_w \; -l(w) = argmin_w \frac{1}{m}||y - Xw||^2$$

Logistic Regression

Binary Classification

Binary classification: We call it a binary classification problem if the variable y only takes values 0 or 1.

Model settings: For a given variable x, we assume the probability that y takes the value 1 is

$$P(y = 1 \mid x; w) = h_w(x) = \frac{1}{1 + e^{-w^T x}}$$

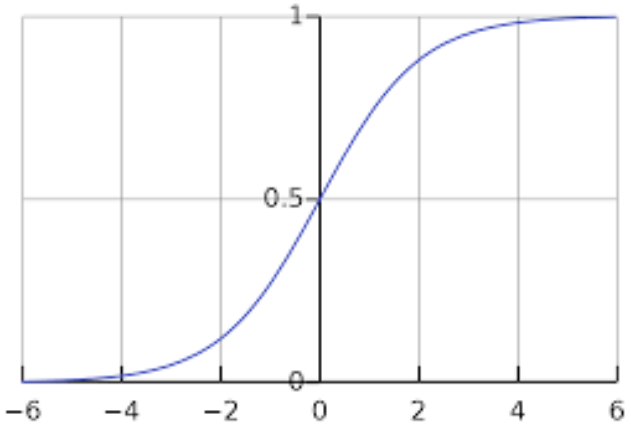
Remark: The probability that y takes the value 0 is

$$P(y = 0 \mid x; w) = 1 - P(y = 1 \mid x; w) = \frac{e^{-w^T x}}{1 + e^{-w^T x}}$$

Sigmoid Function

Sigmoid function (or logistic function):

$$g(z) = \frac{1}{1 + e^{-z}}$$



Likelihood Function

Likelihood function: For a given data set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, the likelihood function is given by

$$L(w) = \prod_{i=1}^{m} h_w\left(x^{(i)}\right)^{y^{(i)}} \left(1 - h_w\left(x^{(i)}\right)\right)^{1-y^{(i)}}$$

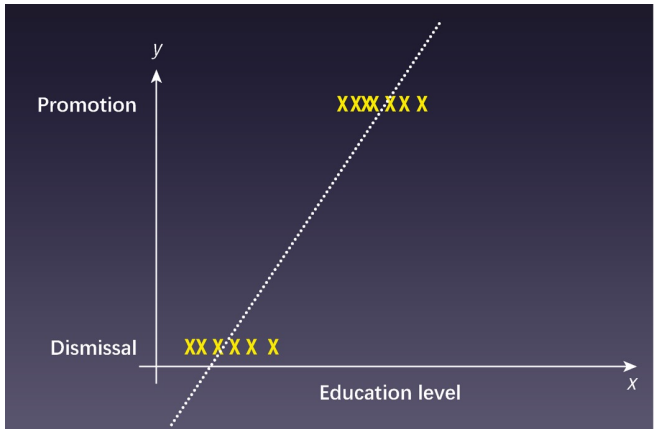
Log Likelihood function: The log likelihood function is given by

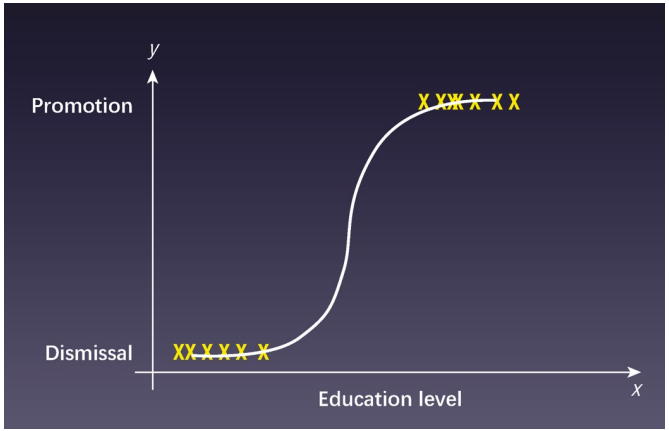
$$l(w) = \log\left(L(w)\right) = \sum_{i=1}^{m} y^{(i)} \log\left(h_w\left(x^{(i)}\right)\right) + (1 - y^{(i)}) \log\left(1 - h_w\left(x^{(i)}\right)\right)$$

Logistic Regression

Logistic regression: For a given data set $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, we find the best model parameter w by maximizing the log likelihood function,

$$w^* = argmax_w \sum_{i=1}^{m} y^{(i)} \log\left(h_w\left(x^{(i)}\right)\right) + (1 - y^{(i)}) \log\left(1 - h_w\left(x^{(i)}\right)\right)$$





Gradient of $l(w)$

$$\frac{\partial}{\partial w_j} \mathrm{y}^{(i)} \log(h_w(x^{(i)})) = y^{(i)} x_j^{(i)} (1 - h_w(x^{(i)}))$$

$$\frac{\partial}{\partial w_j} (1 - \mathrm{y}^{(i)}) \log(1 - h_w(x^{(i)})) = -(1 - y^{(i)}) x_j^{(i)} h_w(x^{(i)})$$

$$\frac{\partial}{\partial w_j} l(w) = \sum_{i=1}^{m} \left(y^{(i)} - h_w(x^{(i)}) \right) x_j^{(i)}$$

$$\nabla l(w) = \left(\frac{\partial}{\partial w_0} l(w), \ldots, \frac{\partial}{\partial w_n} l(w) \right)^T$$

Gradient Ascent Algorithm for Logistic Regression

Start from some $w^0 \in \mathbb{R}^{n+1}$, gradient descent (GD) algorithm updates as:

$$w^{k+1} = w^k + \alpha_k \nabla\, l(w^k),$$

until

$$||\nabla\, l\left(w^{k+1}\right)|| \leq \varepsilon,$$

for some tolerance $\varepsilon > 0$.

Key points:

1. Compute $\boldsymbol{\nabla\, l\left(w^k\right)}$.
2. Choose step size $\alpha_k > 0$ satisfying

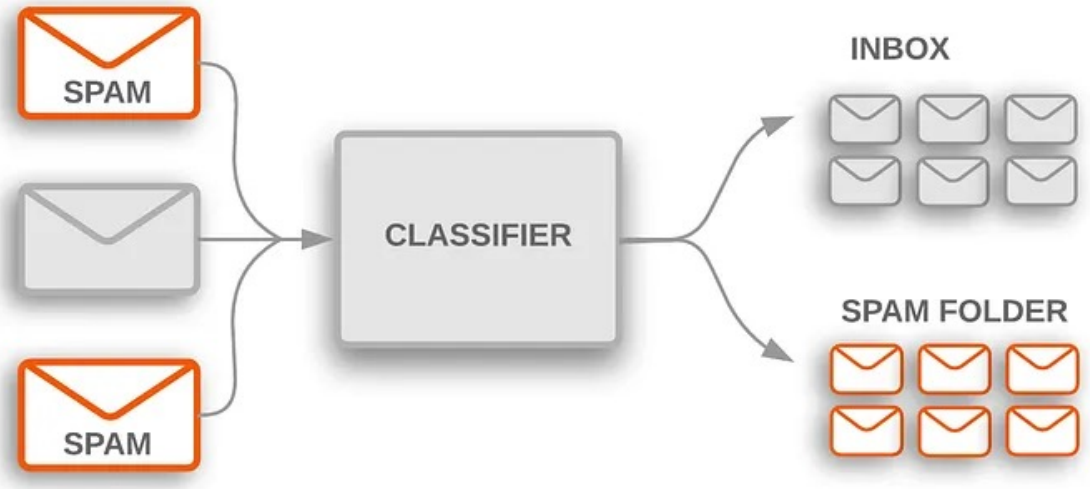
$$\boldsymbol{l\left(w^{k+1}\right) > l(w^k).}$$

Application

Spam Email Detection

Problem settings: We will build a logistic regression model to detect spam emails. Given the content of an email x, If the email is spam, we expect that

$$P(y = 1|x) > P(y = 0\ |x)$$



Data Preparation: Clean the Raw Data

We usually

1. Lowering case.

2. Remove of special characters.

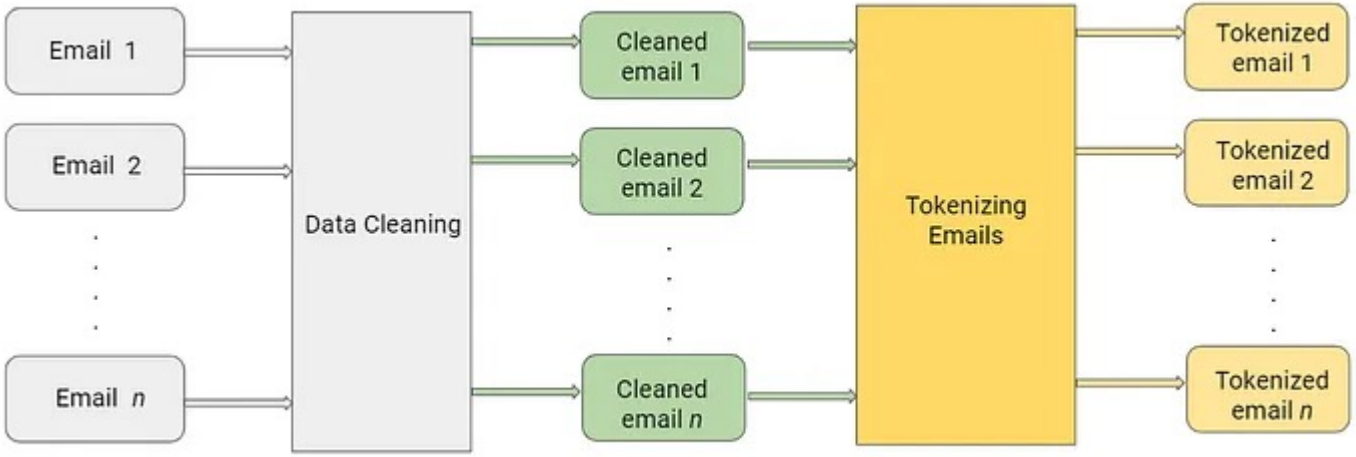
3. Remove stop words.

……

Tokenizing the cleaned data

Step 1: Choose a dictionary with K words.

Step 2: Build a vector v of length K such that v_j is the frequency of the jth word in the dictionary.

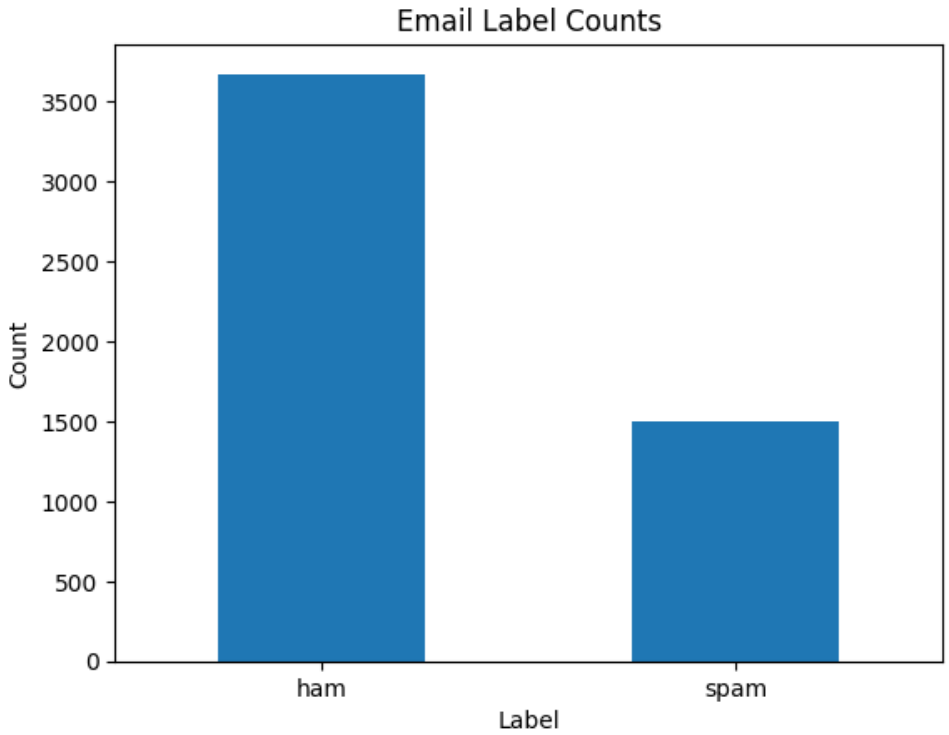


Data

(5171, 3)

	label	text	label_num
605	ham	Subject: enron methanol ; meter # : 988291\r\n...	0
2349	ham	Subject: hpl nom for january 9 , 2001\r\n(see...	0
3624	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
4685	spam	Subject: photoshop , windows , office . cheap ...	1
2030	ham	Subject: re : indian springs\r\nthis deal is t...	0

Data


Email Label Counts
Count
3500
3000
2500
2000
1500
1000
500
0
ham
spam
Label

Data

Spam email subject word cloud



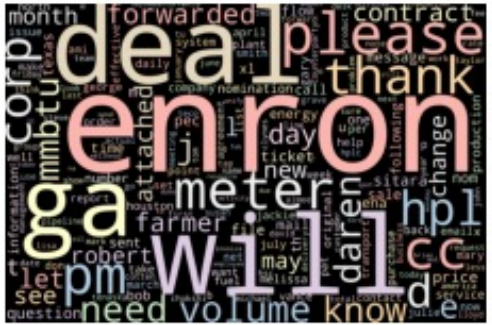
Spam email text word cloud



Ham email subject word cloud



Ham email text word cloud



Results of Logistic Regression



```
# only use subject features
spam_clf = LogisticRegression(penalty=None)
spam_clf.fit(X_subject_train, y_train)
print('Train accuracy:', spam_clf.score(X_subject_train, y_train))
print('Test  accuracy:', spam_clf.score(X_subject_test, y_test))
```

```
Train accuracy: 0.9966841668969328
Test  accuracy: 0.9304123711340206
```

```
1 # only use text features
2 spam_clf = LogisticRegression(penalty=None)
3 spam_clf.fit(X_text_train, y_train)
4 print('Train accuracy:', spam_clf.score(X_text_train, y_train))
5 print('Test  accuracy:', spam_clf.score(X_text_test, y_test))
```

```
Train accuracy: 0.9988947222989776
Test  accuracy: 0.9722938144329897
```

```
 1 from scipy.sparse import hstack
 2
 3 # stack the feature matrix
 4 X_train = hstack([X_text_train, X_subject_train])
 5 X_test = hstack([X_text_test, X_subject_test])
 6
 7 # use both subject and text features
 8 spam_clf = LogisticRegression(penalty=None)
 9 spam_clf.fit(X_train, y_train)
10 print('Train accuracy:', spam_clf.score(X_train, y_train))
11 print('Test  accuracy:', spam_clf.score(X_test, y_test))
```

```
Train accuracy: 1.0
Test  accuracy: 0.9774484536082474
```


Thank You!