Loading the data: Using pandas to read the CSV file containing the spam data. Dividing the data into training and testing parts: Utilizing train_test_split from sklearn.model_selection to split the dataset, ensuring both training and testing datasets are represented. Building and training the regression model: The code assumes the use of logistic regression (a type of regression suitable for binary classification tasks like spam detection) from sklearn.linear_model. It initializes the model, fits it to the training data, and then makes predictions on the test set. Evaluating the model and printing the confusion matrix: Using the confusion matrix from sklearn.metrics to evaluate the model's performance by comparing the actual versus predicted labels on the test set.

In [2]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix

# Load the spam data
spam_data = pd.read_csv('spam-data.csv')

# Separating features and target variable
X = spam_data.drop('Class', axis=1)
y = spam_data['Class']

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Building and training the logistic regression model
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Predicting on the test set
y_pred = model.predict(X_test)

# Calculating the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

print(conf_matrix)
```

```
[[12  0]
 [ 2 15]]
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js