26. **Python Logistic Regression for Predicting Shipping Outcomes in a Warehouse**

I will be diving into the application of various classifiers to shipping data in a warehouse environment. This project focuses on classification techniques, where I'll use logistic regression to predict whether a shipment will be delivered on time or late. If you'd like more details on the dataset or further documentation, you can refer to resources such as the ISLP package documentation available on the statlearning.com website.

As with any machine learning project, I begin by importing all the necessary libraries at the start. Familiar tools like pandas, numpy, and matplotlib are essential, but today's focus will be on machine learning classifiers available through scikit-learn (or sklearn). The classifiers I will use include Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Gaussian Naive Bayes, k-Nearest Neighbors, and logistic regression.

For this exercise, I begin by loading a warehouse shipping dataset. The goal is to predict the OnTime variable, which is a binary outcome indicating whether a shipment is delivered on time ("Yes") or late ("No"). The dataset contains several predictor variables, such as the shipping distance, weight of the packages, time of the day, and warehouse processing speed. Before diving into modeling, it’s always good practice to explore the dataset using a correlation matrix to see if any relationships stand out. If predicting whether a shipment is on time were straightforward, I’d expect to see some strong correlations between predictors like distance or time of day and the OnTime outcome. However, most correlations here are relatively low, except for a noticeable positive correlation between Distance and ShippingTime, indicating that longer distances tend to have higher shipping times.

Next, I fit the first model: **Logistic Regression**. As discussed in previous labs, logistic regression is a standard classifier that belongs to the Generalized Linear Model (GLM) family. In this case, I use the GLM object from the statsmodels package and specify a binomial family to indicate that this is a logistic regression. The fitting process is very similar to that of linear regression, but logistic regression is specifically suited for binary outcomes. After fitting the model, I examine the coefficients for each variable and their statistical significance to assess their predictive power. Unfortunately, I find that no single variable provides strong predictive power for determining whether a shipment will be on time or late. If it were that easy to predict, we could easily optimize warehouse operations, but real-world logistics is often more complex.

The results suggest a weak negative relationship between the shipping distance (Distance) and the OnTime variable, meaning longer distances might slightly increase the likelihood of a late delivery. However, this effect is not particularly strong. I use the results object from statsmodels to extract the predicted probabilities for each shipment. Because this is a binary classification problem, the predictions are probabilities that a shipment will be on time. I create predicted labels based on a threshold of 50%. If the predicted probability is greater than 50%, I label it as "Yes" (on time); otherwise, it is labeled "No" (late). I then use these predictions to create a confusion matrix, comparing predicted outcomes with actual outcomes. Ideally, I want high accuracy, which would be reflected by high values along the diagonal of the matrix. The initial accuracy based on the training data is around 52%, which is only slightly better than random guessing (50%). Even a small improvement over random chance could mean a significant operational advantage in logistics.

However, it's important to note that this accuracy reflects **training error**, which can often be overly optimistic. To get a more realistic estimate of how well this model might perform in practice, I split the data into training and test sets. Since shipping data often involves time-based patterns, I split the dataset by time, using earlier shipments for training and more recent ones for testing. I refit the logistic regression model on the training set and evaluate its performance on the test set. When predicting on-time delivery for the test set, the accuracy drops to around 48%, which is slightly worse than random guessing. This indicates that the model might not have strong predictive power, highlighting the challenge of forecasting shipping outcomes.

Given the poor performance of the initial model, I decide to simplify it. The original model used several predictors, including distance, weight, and time of day. I reduce this to a model that uses only the two most relevant predictors, such as Distance and TimeOfDay. This adjustment aims to balance the bias-variance trade-off better. After fitting this simpler model, the test accuracy improves to around 56%, which is better than random guessing. This suggests that a more straightforward model might provide a small but meaningful improvement in predicting shipping outcomes. It could serve as a basic decision-support tool in a warehouse setting, though further refinement and additional data would be needed for a production-level solution.

To extend this analysis further, I could use the fitted model to make new predictions on future shipments by updating it with observed data on distance and time of day for each new shipment. This would provide a more dynamic model capable of real-time forecasting in a warehouse management system.