**65. Comparison with Logistic Regression in Business Analytics**

**Predicting Customer Churn with Heart Data**

Now that I have covered Support Vector Machines (SVMs) in their full generality, it's time to see how they work in a practical example. Let’s take a business scenario where I want to predict customer churn based on customer data. Here, I use the heart dataset as a proxy, which consists of about ten variables, and I need to classify customers into two categories: "high risk of churn" or "low risk of churn." I’ll apply Support Vector Machines to classify these categories.

In the following analysis, I focus on the performance of SVMs on the training data. Later, I will compare it to the test data. To visualize the performance, I use **ROC (Receiver Operating Characteristic) curves**, which are a standard tool in classification problems to evaluate model performance.

When fitting an SVM model, which starts with a linear decision boundary, I must decide on a threshold to classify a customer as high risk or low risk. Initially, this threshold is set at zero. If the decision function is greater than zero, I classify it as "high risk"; otherwise, it is classified as "low risk." With this classification, I will have both false positives (predicting high risk when it’s not) and false negatives (predicting low risk when it’s actually high).

The ROC curve helps trace the trade-off between true positives (correctly predicting high risk) and false positives as I change this threshold. Ideally, the curve should "hug" the upper left corner of the plot, indicating high true positive rates and low false positive rates. The closer a curve is to this corner, the better the classifier performs.

**Comparing Classifiers: SVM vs. Linear Discriminant Analysis**

To provide a comparative analysis of classifiers, I start by comparing a **linear SVM classifier** (red curve) with **Linear Discriminant Analysis (LDA)** (blue curve) on the training data. Both classifiers perform similarly, but the SVM appears to have a slight edge in certain areas. The performance is measured using the **Area Under the Curve (AUC)** metric, which quantifies the overall ability of the model to distinguish between the two classes. An AUC of 1 would indicate perfect classification, while an AUC of 0.5 would suggest random guessing.

Next, I compare the linear SVM classifier with an **SVM using a Radial Basis Function (RBF) kernel** at different gamma values. Gamma controls the flexibility of the decision boundary; higher gamma values lead to more complex boundaries. The plot shows that with gamma set to , the SVM performs the best, but as I reduce gamma (e.g., to ), the performance drops. This demonstrates that gamma is a tuning parameter in SVMs that needs to be carefully selected to avoid overfitting or underfitting.

However, I must note that comparing models based on training data alone is misleading since more complex models (higher gamma) are likely to fit better on training data, potentially leading to overfitting. To address this, I move to evaluating performance on test data.

**Evaluating SVMs and Logistic Regression on Test Data**

To avoid overfitting, I set aside 80 observations as test data. After fitting the classifiers on the training data, I evaluate their performance using ROC curves on this unseen test data. Comparing the linear SVM classifier with LDA, the SVM slightly outperforms, but the difference is marginal. Notably, the ROC curve on the test data is not as impressive as on the training data, indicating some level of overfitting.

When comparing different SVM models with varying gamma values, I observe that the model with the highest gamma value (which performed best on the training data) actually performs the worst on the test data. This reinforces the importance of choosing gamma carefully and validating the choice using cross-validation or a validation dataset. With SVMs, I also need to tune the cost parameter (C), meaning there are two key tuning parameters for kernel-based SVMs: gamma and C. For linear SVMs, I only need to tune C.

**SVMs: Handling More Than Two Classes**

Up until now, I’ve been focusing on binary classification (two classes), but what happens when I need to classify more than two categories? For instance, I might want to segment customers into "high risk," "moderate risk," and "low risk" of churn. SVMs handle multi-class problems using two general approaches:

1. **One-vs-All (OVA):** This approach involves fitting KKK separate two-class SVM classifiers, each one distinguishing a particular class from all others. For example, if I have three segments (high, moderate, low), I fit three SVMs: one to distinguish "high" from "not high," another for "moderate" vs. "not moderate," and so on. When classifying a new customer, I evaluate all three models and assign the customer to the class with the highest score.
2. **One-vs-One (OVO):** This method involves fitting a classifier for every possible pair of classes. If I have 10 segments, this results in classifiers. For a new customer, I evaluate all classifiers and use a majority voting scheme to determine the final classification.

While both methods are somewhat ad hoc, they work reasonably well in practice. For large numbers of classes, OVA tends to be favored due to computational efficiency.

**Support Vector Machines vs. Logistic Regression**

Now, I’ll compare Support Vector Machines (SVMs) with **Logistic Regression (LR)**, which is another common method for binary classification problems. Logistic Regression models the probabilities of the classes and provides decision boundaries by optimizing for these probabilities. In contrast, SVMs focus on optimizing the decision boundary directly.

Despite these differences, they are not as distinct as they might seem. I can reframe the SVM optimization problem to look quite similar to that of logistic regression, but with a different loss function known as the **hinge loss**. The hinge loss penalizes points that are on the wrong side of the decision boundary, leading to a "support vector" property where only certain points (support vectors) directly influence the decision boundary.

On the other hand, logistic regression uses a **log-likelihood loss function**, which has a smoother curve compared to hinge loss. This smoothness can be thought of as a “soft margin” that focuses more on points near the decision boundary rather than those further away. This difference allows logistic regression to estimate probabilities more naturally than SVMs.

**Choosing Between SVM and Logistic Regression**

So, when should I use SVMs versus logistic regression in business analytics?

1. **When Classes Are Nearly Separable:** SVMs, particularly with a linear kernel, tend to perform better when the classes are almost separable. Logistic regression may break down in these scenarios unless some regularization is applied.
2. **When Classes Overlap Significantly:** If there’s significant overlap between the classes, logistic regression with regularization (like Lasso or Ridge penalties) may perform better because it models the probabilities and provides more useful estimates for business decision-making.
3. **For Non-linear Decision Boundaries:** Kernel-based SVMs (e.g., with RBF kernels) are well-suited for non-linear decision boundaries. These can be used in logistic regression and LDA too, but they are computationally more intensive.
4. **Probability Estimates Are Important:** If I need to understand the likelihood of an outcome (e.g., the probability that a customer will churn), logistic regression is more straightforward and interpretable.

**Trade-offs and Considerations in Business Analytics**

While SVMs provide a powerful tool, especially with kernel methods, they come with trade-offs. One limitation is that SVMs do not naturally perform feature selection, unlike Lasso regression, which can zero out less important features. This can be a drawback in high-dimensional problems, where interpretability is crucial.

Furthermore, in situations where probability estimates are more critical than just classification (such as predicting customer lifetime value or conversion rates), logistic regression is more appropriate. While the SVM community has developed some post hoc methods, like recursive feature elimination and techniques to estimate probabilities after fitting, these are less straightforward than directly using logistic regression.

In summary, the choice between SVMs and logistic regression in business analytics will depend on the specific problem, the need for probability estimates, and the computational resources available. Each method has its strengths and is suited for different types of data and business contexts.