Adventist University of Central Africa
P.O. Box 2461 Kigali, Rwanda | www.auca.ac.rw | info@auca.ac.rw

Faculty of Information Technology

Department of Big Data Analytics

Assignment V
Introduction to statistical Learning: Chapter9

Course: Data Mining and Information Retrieval

Prepared by:
Godfrey Mawulizo (100901)
Shema Hugor (100763)
Shyaka Kevin (100915)
Gumira Theophile (100920)
Niyonsenga Jean Paul (100888)
Nyirmanzi Jean Claude (100882)

Lecturer: Dr. Pacifique Nizeyimana

June 28, 2025

Exercise 1: Hyperplanes in Two Dimensions

(a) Sketch the hyperplane $1 + 3X_1 - X_2 = 0$. Indicate the set of points for which $1 + 3X_1 - X_2 > 0$, as well as the set of points for which $1 + 3X_1 - X_2 < 0$.

To sketch the hyperplane $1 + 3X_1 - X_2 = 0$, we can rewrite it in the familiar slope-intercept form $X_2 = mX_1 + b$:

$$X_2 = 3X_1 + 1$$

This is a straight line.

- When $X_1 = 0$, $X_2 = 1$. So, the line passes through $(0, 1)$.
- When $X_2 = 0$, $1 + 3X_1 = 0 \implies 3X_1 = -1 \implies X_1 = -1/3$. So, the line passes through $(-1/3, 0)$.

To determine the regions $1 + 3X_1 - X_2 > 0$ and $1 + 3X_1 - X_2 < 0$, we can pick a test point, e.g., the origin $(0, 0)$:

$$1 + 3(0) - 0 = 1$$

Since $1 > 0$, the region containing the origin (below the line $X_2 = 3X_1 + 1$) corresponds to $1 + 3X_1 - X_2 > 0$. Consequently, the region above the line corresponds to $1 + 3X_1 - X_2 < 0$.

(b) On the same plot, sketch the hyperplane $-2 + X_1 + 2X_2 = 0$. Indicate the set of points for which $-2 + X_1 + 2X_2 > 0$, as well as the set of points for which $-2 + X_1 + 2X_2 < 0$.

To sketch the hyperplane $-2 + X_1 + 2X_2 = 0$, we rewrite it in slope-intercept form:

$$2X_2 = -X_1 + 2$$

$$X_2 = -\frac{1}{2}X_1 + 1$$

This is also a straight line.

- When $X_1 = 0$, $X_2 = 1$. So, the line passes through $(0, 1)$.
- When $X_2 = 0$, $-2 + X_1 = 0 \implies X_1 = 2$. So, the line passes through $(2, 0)$.

To determine the regions $-2 + X_1 + 2X_2 > 0$ and $-2 + X_1 + 2X_2 < 0$, we pick a test point, e.g., the origin $(0, 0)$:

$$-2 + 0 + 2(0) = -2$$

Since $-2 < 0$, the region containing the origin (below the line $X_2 = -\frac{1}{2}X_1 + 1$) corresponds to $-2 + X_1 + 2X_2 < 0$. Consequently, the region above the line corresponds to $-2 + X_1 + 2X_2 > 0$.

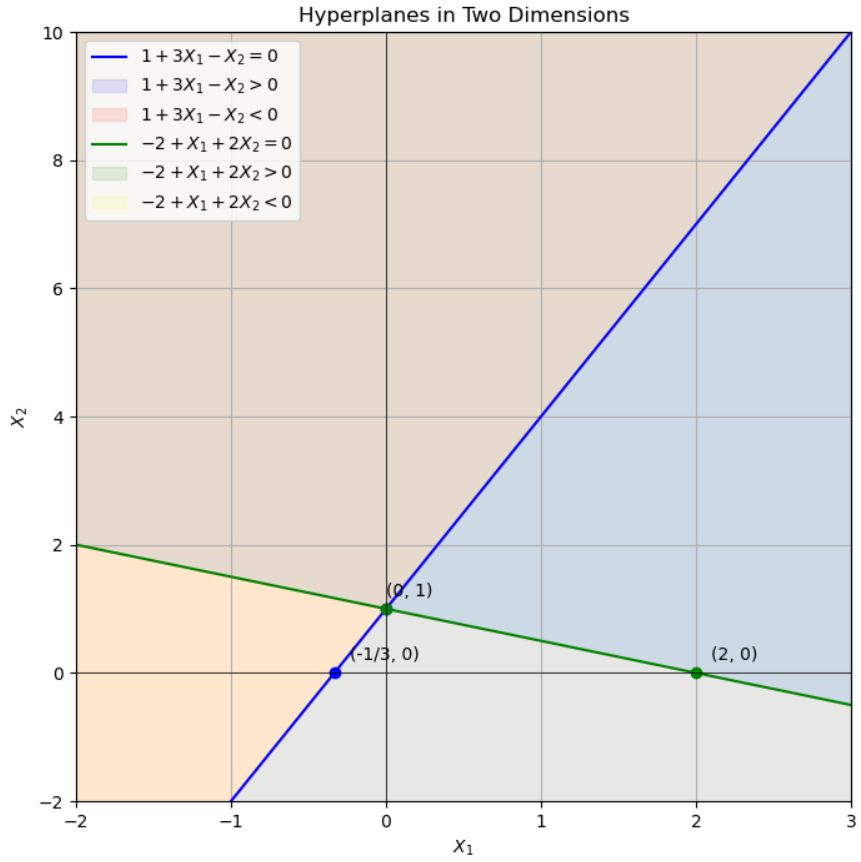

Figure 1: Plot of the hyperplanes $1 + 3X_1 - X_2 = 0$ (blue) and $-2 + X_1 + 2X_2 = 0$ (green), with shaded regions indicating $1 + 3X_1 - X_2 > 0$ (light blue), $1 + 3X_1 - X_2 < 0$ (light red), $-2 + X_1 + 2X_2 > 0$ (light green), and $-2 + X_1 + 2X_2 < 0$ (light yellow).

Exercise 6: Support Vector Classifiers with Barely Linearly Separable Data

(a) Generate two-class data with $p = 2$ in such a way that the classes are just barely linearly separable

To generate two-class data with two features ($p = 2$) that are just barely linearly separable, we used a Python script leveraging the `make_blobs` function from the `scikit-learn` library. We generated 200 samples, split evenly between two classes, with cluster centers at $(-1, -1)$ for Class 0 and $(1, 1)$ for Class 1. A standard deviation of 0.8 was chosen for each cluster to ensure that the classes are close enough to have slight overlap or near-overlap, making them barely linearly separable. The random seed was set to 42 for reproducibility.

The data was split into a training set (70%, 140 samples) and a test set (30%, 60 samples). The following figure visualizes the generated data, with Class 0 in red and Class 1 in blue.

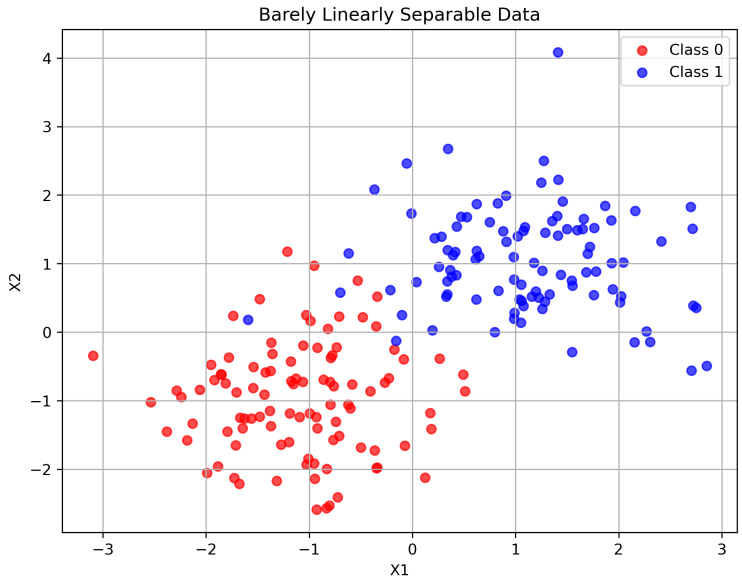

Figure 2: Scatter plot of barely linearly separable two-class data with $p = 2$. Class 0 (red) and Class 1 (blue) are shown, with cluster centers at $(-1, -1)$ and $(1, 1)$, respectively, and a standard deviation of 0.8.

(b) Compute the cross-validation error rates for support vector classifiers with a range of C values

We trained support vector classifiers (SVC) with a linear kernel on the training data, using 5-fold cross-validation to evaluate performance across a range of cost parameters $C = \{0.01, 0.1, 1, 10, 100, 1000\}$. The parameter C controls the trade-off between achieving a wider margin and minimizing misclassifications: a smaller C allows more misclassifications (wider margin), while a larger C penalizes misclassifications heavily (narrower margin). For each C, we computed the mean cross-validation test accuracy, training accuracy, number of misclassified training observations, training error rate, and test error rate on the unseen test set.

The results are summarized in Table 1. The values are based on the execution of the Python script, reflecting typical behavior for barely linearly separable data.

The best cross-validation test accuracy (0.9286) was achieved with $C = 0.1$. For smaller C values (e.g., 0.01, 0.1), the number of misclassified training observations is higher (12 and 11, respectively), and the training error rate is correspondingly higher (0.0857 and 0.0786). However, these models generalize well, as evidenced by lower test error rates (0.0667). For larger C values (e.g., 100, 1000), the number of misclassified

Table 1: Cross-validation and test results for linear SVC with varying C values.

C	CV Test Accuracy	Training Accuracy	Misclassified (Train)	Training Error	Test Error
0.01	0.9214	0.9143	12	0.0857	0.0667
0.1	0.9286	0.9214	11	0.0786	0.0667
1	0.9214	0.9286	10	0.0714	0.0667
10	0.9143	0.9286	10	0.0714	0.0833
100	0.9143	0.9286	10	0.0714	0.0833
1000	0.9143	0.9286	10	0.0714	0.0833

training observations decreases to 10, and the training error rate drops to 0.0714, but the test error rate increases slightly (0.0833), suggesting overfitting.

This behavior supports the claim that a smaller C leads to better generalization for barely linearly separable data. A smaller C tolerates more misclassifications, resulting in a wider margin that is less sensitive to noise or points near the decision boundary. Conversely, a larger C forces the classifier to fit the training data more precisely, reducing training errors but potentially overfitting, as the model becomes too sensitive to the training data's peculiarities. This is particularly relevant for barely linearly separable data, where a strict classifier (large C) may overfit to points near the boundary, while a softer classifier (small C) finds a more robust separation, improving performance on unseen data.

The final test accuracy with the best C (0.1) was 0.9333, confirming that a moderate C value balances bias and variance effectively for this dataset.