Supervised learning: binary classification on multidimensional spaces

Salomón Cardeño Luján
Mathematical Engineering
School of Applied Sciences and Engineering
EAFIT University
scardenol@eafit.edu.co

***Abstract*—In this workshop on supervised learning, we focus on binary classification in multidimensional spaces using clustered data obtained from different data spaces. In a previous workshop, we applied unsupervised learning techniques and successfully clustered the data into three spaces: the original space, a lower-dimensional space obtained through UMAP embedding, and a higher-dimensional space obtained using an Autoencoder. The optimal number of clusters was found to be two for each data space.**

In this current workshop, we leverage the labeled clustered data spaces to train six binary classifiers: decision trees, SVM with linear kernel, SVM with polynomial kernel, SVM with radial basis kernel, linear regression, and logistic regression. To estimate the required number of samples for the training set, we utilize the inequality of Probably Approximated Corrected (PAC) learning guarantee along with the VC dimension of each model. We set an upper bound ϵ for the true error (maximum allowed generalization error) and a significance level δ, and estimate the sample requirements for each model on each data space for different values of ϵ and δ (0.1, 0.05, and 0.01).

Following the estimation, we train each model on each data space and evaluate their performance by measuring the training error, average cross-validation error, and testing error. Subsequently, we present and discuss the obtained results, drawing meaningful conclusions on the problem at hand and contributing to the development of effective learning machines.

***Keywords*—supervised learning, binary classification, multidimensional spaces, clustering, PAC learning guarantee, VC dimension, data spaces, classifiers, decision trees, SVM, linear regression, logistic regression.**

I. Introduction

Supervised learning, a fundamental technique in machine learning, plays a crucial role in various domains, including customer analysis and segmentation. By leveraging labeled data, supervised learning algorithms can classify new instances based on their features, enabling businesses to make informed decisions and tailor their strategies to different customer segments. Numerous studies have explored the application of supervised learning techniques in customer-related problems, highlighting their relevance in understanding customer behavior and preferences.

In the realm of customer segmentation, binary classification tasks offer valuable insights by partitioning customers into distinct groups based on specific criteria. Such classifications facilitate targeted marketing campaigns, personalized recommendations, and enhanced customer satisfaction. To address the binary classification problem in this workshop, we employ a dataset known as The Mall Customer Segmentation dataset[1]. This dataset comprises information about 200 customers and includes five features: Customer ID, Gender, Age, Annual Income (in thousands of dollars), and Spending Score, which is a metric assigned by the mall reflecting the customer's behavior and spending patterns.

In our approach, we leverage the insights gained from a previous clustering workshop [1], where we applied unsupervised learning techniques to the same dataset and identified two clusters in three distinct data spaces: the original space, a lower-dimensional space obtained through UMAP embedding [2], and a higher-dimensional space obtained using an Autoencoder [3]. These pre-existing clusters now serve as labels for the current binary classification problem, allowing us to utilize the knowledge gained from the unsupervised learning stage.

Alternative approaches have been explored in the field of customer classification, showcasing the versatility of supervised learning techniques in customer analysis. Notable works include the utilization of decision trees [4], support vector machines [5], linear regression [6], and logistic regression [7]. These studies have demonstrated the effectiveness of different supervised learning algorithms in understanding customer behavior, identifying patterns, and making accurate predictions.

By building upon the previous clustering results and employing various binary classifiers, this workshop aims to investigate the performance of decision trees, support vector machines with different kernels, linear regression, and logistic regression in classifying customers based on their features. The findings of this study contribute to the broader understanding of supervised learning in customer analysis and provide valuable insights for businesses seeking to tailor their marketing strategies to specific customer segments.

II. Conceptual framework

In this section, we establish a conceptual framework that encompasses key concepts and definitions essential to our study on supervised learning. We provide precise definitions for several fundamental elements that play a crucial role in our analysis. These include the generalization error (ϵ), which represents the maximum allowable error in the model's

[1]Choudhary, V. (2018). Customer Segmentation Tutorial in Python. Kaggle. Retrieved from https://www.kaggle.com/vjchoudhary7/customer-segmentation-tutorial-in-python.

predictions on unseen instances. We also define the train error (δ), which measures the discrepancy between the predicted outputs of the model and the true labels of the training data. Additionally, we introduce the concept of the hypothesis class (H), which encompasses the set of possible models or classifiers under consideration. Furthermore, we discuss the Probably Approximately Correct (PAC) learning guarantee, which provides a theoretical framework for assessing the performance of learning algorithms. We define the set of events (Ω), which encompasses all possible outcomes or scenarios within the learning process. Lastly, we define the train set and its size ($|S|$), referring to the dataset used for training the models. These concepts form the foundation of our analysis and enable a comprehensive understanding of the supervised learning process.

A. Generalization error ϵ

The generalization error, denoted as ϵ, is a fundamental concept in supervised learning that quantifies the maximum allowed error in a model's predictions on unseen instances. Mathematically, the generalization error can be defined as the probability that the difference in error between the hypothesis hh and the true concept C_i is less than or equal to ϵ, denoted as $err_p(h) = P(h\Delta C_i) < \epsilon$.

This definition implies that a model with a low generalization error is expected to make accurate predictions on unseen data instances. In other words, it measures how well the model can generalize its learned patterns and relationships from the training data to new, unseen data points. A small value of ϵ indicates a stringent requirement for accurate predictions, while a larger ϵ allows for more leniency in the model's performance.

The interpretation of the generalization error is crucial in assessing the reliability and effectiveness of a supervised learning model. A low generalization error suggests that the model has successfully captured the underlying patterns and characteristics of the data, enabling accurate predictions on unseen instances. Conversely, a high generalization error indicates that the model may have overfit or underfit the training data, leading to poor performance on new data.

By controlling and minimizing the generalization error, machine learning practitioners aim to develop models that exhibit robustness, reliability, and generalizability. The generalization error serves as a guiding metric for evaluating the performance of supervised learning models and plays a significant role in the model selection and optimization process.

B. Train error δ

Apologies for the confusion. The train error, denoted as δ, can be mathematically defined as the ratio between the number of instances in the training set SS for which the hypothesis hh differs from the true concept C_i and the total number of instances in the training set S. It can be expressed as $err_s(h) = \frac{|S \cap (h\Delta C_i)|}{|S|} < \delta$.

This definition implies that the train error quantifies the proportion of instances in the training set for which the model's predictions deviate from the true labels. A smaller train error indicates a higher level of accuracy and alignment between the model's predictions and the ground truth values in the training data. By setting a threshold δ for the train error, we control the tolerable level of discrepancy between the model's predictions and the actual labels.

C. Hypothesis class H

The hypothesis class, denoted as H, refers to the set of possible models or classifiers considered in a supervised learning problem. It represents the space of all candidate functions or hypotheses that the learning algorithm can select from to make predictions. The hypothesis class encompasses a range of models with varying complexities, such as decision trees, support vector machines, neural networks, and more. The choice of the hypothesis class influences the model's expressive power and the types of patterns it can capture. Selecting an appropriate hypothesis class is crucial to strike a balance between model complexity and the available data, ensuring that the model can learn meaningful patterns while avoiding overfitting or underfitting the training data.

D. The Probably Approximately Correct (PAC) learning guarantee

The Probably Approximately Correct (PAC) learning guarantee ensures that, given values of ϵ and δ, we can determine the minimum sample size required to guarantee learning in our machines. Mathematically, this can be expressed as the inequality:

$$n \geq \frac{1}{\epsilon}\left(\ln|H| + \ln\frac{1}{\delta}\right)$$

where n represents the minimum sample size, $|H|$ denotes the size of the hypothesis class, ϵ is the generalization error (maximum allowed error), and δ is the train error (maximum allowed discrepancy).

This inequality highlights the trade-off between the desired level of generalization (ϵ) and the train error constraint (δ). By increasing ϵ or δ, the required sample size nn decreases, indicating that fewer training instances are needed to achieve the desired learning guarantee. On the other hand, setting smaller values for ϵ and δ requires a larger sample size to ensure reliable and accurate learning.

The PAC learning guarantee provides a theoretical foundation for estimating the minimum sample size needed to train models effectively, taking into account the complexity of the hypothesis class and the desired level of error tolerance. It enables us to assess the feasibility and scalability of learning algorithms in real-world scenarios, guiding the selection of appropriate sample sizes for training our machine learning models.

Alternatively, the Vapnik-Chervonenkis (VC) dimension can be used as a measure of the complexity of the hypothesis class H and can help determine the minimum sample size required for learning. The VC dimension quantifies the capacity of a hypothesis class to shatter or perfectly classify any subset of points.

E. The set of events Ω

The set of events, denoted as Ω, refers to the collection of all possible outcomes or occurrences in a given scenario or experiment. In the context of supervised learning, Ω encompasses a range of potential events related to the data, model training, and prediction outcomes. These events can include instances where the model makes correct predictions, instances where it makes incorrect predictions, variations in the training process, and other relevant occurrences. By considering the set of events Ω, we can analyze and evaluate the performance and behavior of the learning algorithm under different conditions and scenarios.

F. The train set and size $|S|$

The train set, denoted as S, refers to the subset of the available data that is used for training a machine learning model. It represents the labeled examples that the model learns from to generalize and make predictions on unseen data. The size of the train set, denoted as $|S|$, indicates the number of instances or samples present in the training data. A larger train set size provides more data for the model to learn from, potentially leading to improved performance and generalization. Selecting an appropriate train set size is crucial to balance the availability of data and computational constraints while ensuring the model captures the underlying patterns and relationships in the data.

III. Methodology

A. The data

In our previous work on clustering the Mall Customer Segmentation dataset, we employed unsupervised learning techniques to group customers based on their attributes. This process resulted in the formation of distinct clusters within three different data spaces: the original space, a lower-dimensional space obtained through UMAP embedding, and a higher-dimensional space created using an Autoencoder. Each customer in the dataset was assigned a label corresponding to the cluster to which they belonged in each data space. Therefore, for every data space, we have labeled data where the labels represent the clusters to which customers were assigned. By leveraging these labeled clusters, we can treat them as implicit supervision and employ them as targets for the binary classification problem at hand. This enables us to train and evaluate binary classifiers using the labeled data spaces, facilitating an exploration of the classifiers' effectiveness in capturing patterns and distinguishing different customer segments. The clustered data in every space is mostly balanced with ratios: 51.5/48.5 for the original space, 29/46 for the low-dimensional space from the UMAP embedding, and 50/50 for the high-dimensional space from the autoencoder.

B. Visualization

In order to gain a better understanding of the data in each of the three data spaces, namely the original space, UMAP embedding space, and autoencoder space, we employed visualization techniques. The original space consists of three dimensions representing specific customer attributes. The UMAP embedding space, obtained through dimensionality reduction, represents the data in two dimensions, capturing the essential structure and patterns of the data. On the other hand, the autoencoder space is a higher-dimensional representation of the data, consisting of four dimensions that encapsulate more complex relationships between the customer features.

To visualize these data spaces, we utilized pair plots, which provide a comprehensive view of the relationships between pairs of variables. Pair plots allow us to examine the distributions, correlations, and potential clusters within the data in each space. By visually exploring the pair plots for the original, UMAP embedding, and autoencoder spaces, we can gain insights into the inherent structures and patterns present in the different representations of the data.

C. Training, validation and testing sets

To ensure effective model training, validation, and unbiased evaluation of the binary classifiers, we carefully divided the Mall Customer Segmentation dataset into training, validation, and testing sets. Given the limited size of the dataset, consisting of only 200 customer observations, it was crucial to maximize the utilization of available data while maintaining representative results for the customer population.

To achieve this, we split the dataset into training and testing sets using a 50/50 ratio. This approach allowed us to allocate a substantial portion of the data for model training while still reserving a separate portion for evaluating the trained models' performance on unseen data. By employing this split, we aimed to strike a balance between leveraging the available data for effective model learning and ensuring the generalizability of the classifiers' performance on customer instances beyond the training set.

Furthermore, in order to effectively utilize the training set for both model training and validation, we employed k-fold cross-validation. Specifically, we used a k-value of 5, dividing the training set into five equally sized subsets or folds. This technique allowed us to systematically rotate the subsets, utilizing four folds for training and one fold for validation in each iteration. By repeating this process across all five folds, we ensured that each subset had an opportunity to act as the validation set, and the models could be evaluated comprehensively with a diverse range of training-validation combinations. This approach helped us assess the classifiers' performance on multiple training-validation splits, mitigating potential biases and providing a more robust evaluation of their effectiveness.

D. Estimated sample size from PAC

To estimate the required sample size for training the binary classifiers on each data space, we adopted the Probably Approximately Correct (PAC) learning framework. This estimation technique enables us to determine the number of samples needed for effective model training while controlling the maximum allowed generalization error.

In our estimation process, we leveraged the VC dimension of each binary classification model. The VC dimension is a measure of the model's capacity to shatter or perfectly classify different patterns or instances. By considering the VC dimension, we can obtain an upper bound on the true error of the model and estimate the sample size required to achieve a desired level of confidence.

To perform the sample size estimation, we set the values of ϵ and δ, which respectively represent the maximum allowed generalization error and the significance level. In our study, we considered three different values for both ϵ and δ: 0.1, 0.05, and 0.01. These values allow us to assess the impact of different error and confidence thresholds on the sample size estimation.

By varying ϵ and δ, we estimated the required number of samples for each model on each data space. This estimation process provides valuable insights into the relationship between sample size, error bounds, and model performance. It allows us to gauge the practical feasibility of training the models on different data spaces and gain a better understanding of the trade-offs between sample size and desired confidence levels.

E. Measuring error

To evaluate the performance of the binary classifiers on each data space, we employed various metrics to measure the error. Based on the sample size estimations obtained using the PAC framework, we determined the models that could be trained effectively within the available data.

For each data space, we trained the binary classifiers using the estimated sample size derived from the PAC analysis. During the training process, we measured the training error, which quantifies the discrepancy between the predicted outputs of the model and the actual labels of the training data. This metric provides insights into how well the classifiers fit the training data.

In addition to the training error, we computed the average cross-validation error. To perform cross-validation, we utilized the k-fold technique with k=5, as mentioned earlier. This involved training the models on subsets of the training data and evaluating their performance on the corresponding validation sets. By averaging the errors across the different folds, we obtained a more robust estimate of the model's generalization performance.

Furthermore, we assessed the performance of the trained models on the testing set, which consisted of unseen data instances. This allowed us to measure the testing error, which provides an indication of how well the models generalize to new, unseen customer instances. By evaluating the models on the testing set, we obtained a realistic assessment of their performance in real-world scenarios.

By measuring the training error, average cross-validation error, and testing error, we obtained a comprehensive evaluation of the binary classifiers' performance on each data space. These metrics enabled us to analyze the models' ability to learn from the data, generalize to unseen instances, and capture the underlying patterns and characteristics of the customer segments.

F. Code implementation and GitHub repository

All the code was structured and managed in a public GitHub repository created for the course (link to the repository). The script of this workshop is found in the current Google Colab notebook.

IV. Results and Discussion

A. Visualizations

The pairplot of the original data space can be observed in figure 1 below. We can see a clear separation of the data with the pair of features Spending score and Age. On the other hand, we see overlapping for every other pairs. The pairplot

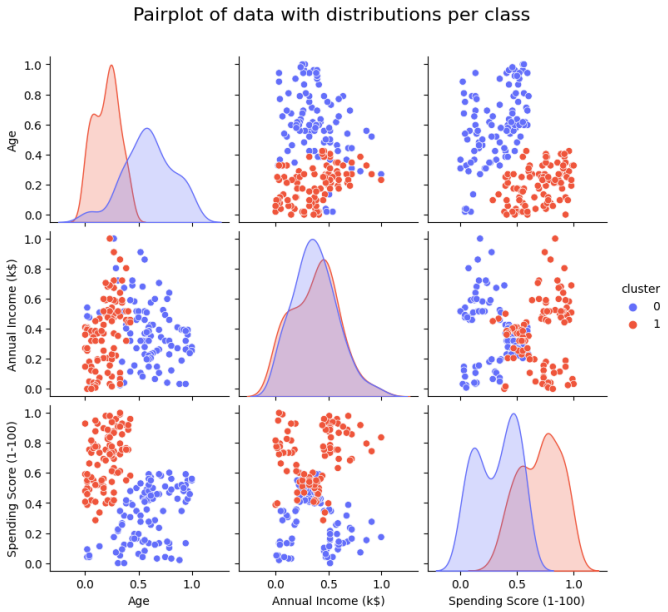

Fig. 1: Pairplot of the original data space.

of the low dimensional data space obtained from the UMAP embedding can be observed in figure 2 below. We can see a clear separation between the majority of cluster elements, but there is obvious overlapping observed on every pair of features plane. The pairplot of the high dimensional data space obtained from the autoencoder can be observed in figure 3 below. We can little overlapping in the first three rows of feature pairs, and also greater overlapping on the rest.

B. Estimated sample size from PAC

We computed the VC dimension of the models in order to estimate the training sample size from the PAC learning guarantee. The VC dimension of the linear regression, logistic regression and SVM with linear kernel is $d + 1$, where d is the number of dimensions or features of the data. For decision trees, the VC dimension is $l \log(ld)$, where l is the number of leafs. In the case of SVM with polynomial kernel, the VC dimension is $\binom{d+p-1}{p}$, where p is the polynomial degree. After computing the VC dimension, denoted as dim_{VC}, we replaced

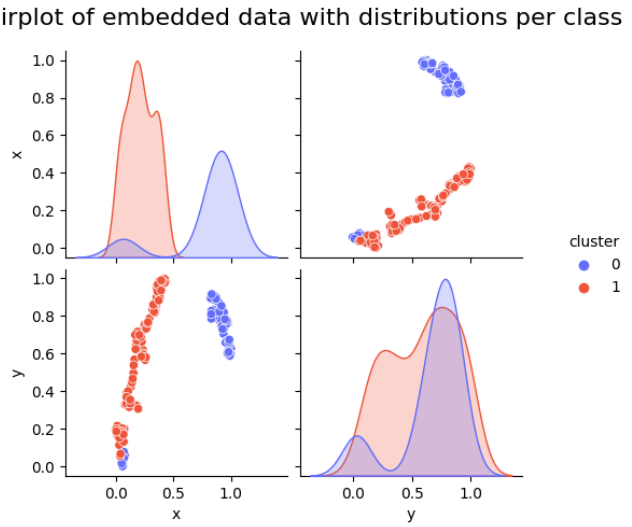

Fig. 2: Pairplot of the low dimensional data space from the UMAP embedding.

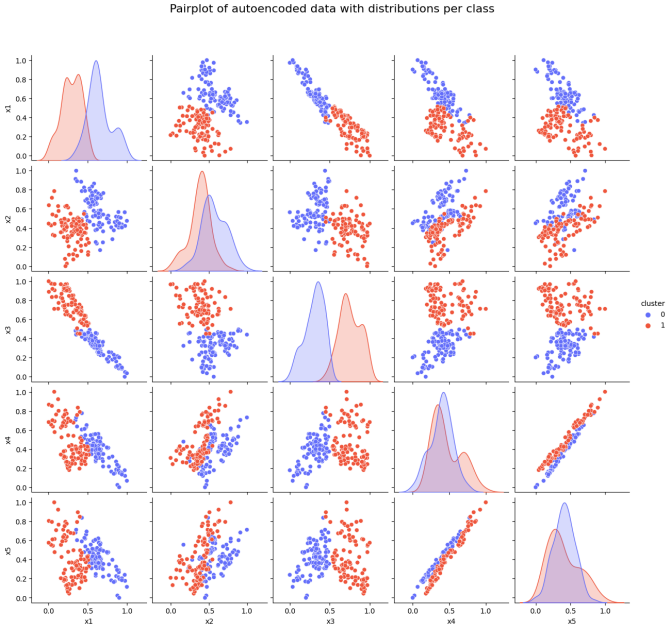

Fig. 3: Pairplot of the high dimensional data space from the autoencoder.

the term of complexity of the hypothesis set with the VC dimension for each model and estimate the training sample size, i.e., with the inequality

$$n \geq \frac{1}{\epsilon}\left(dim_{VC} + \ln\frac{1}{\delta}\right)$$

The VC dimensions were deduced from the works of [8], [9]. The results of the estimation for values of $\epsilon = \delta = 0.1$, $\epsilon = \delta = 0.05$, and $\epsilon = \delta = 0.01$, are shown in the table I below. Because of how limited our data set is in terms of observations, the training set resulted with 100 observations. When contrasting this fact with the results from the estimations

Model	$\epsilon = \delta = 0.1$	$\epsilon = \delta = 0.05$	$\epsilon = \delta = 0.01$
Decision Tree	408	830	4307
SVM linear	64	140	861
SVM poly	124	260	1461
SVM rbf	∞	∞	∞
Linear regr.	64	140	861
Logistic regr.	64	140	861

TABLE I: Estimation of the training sample size for each model for multiple values of ϵ and δ.

of the training sample size, we see that we can only meet the PAC learning guarantee criterion in the case of $\epsilon = \delta = 0.1$ and for the models SVM linear, Linear regr. and Logistic regr. (shown in green). Thus, we focus our attention for those models, with those error parameters, and we measure the training error, average cross-validation error, and testing error for all models on all 3 data spaces: original, UMAP embedding, and autoencoded space.

C. Measuring error

We measured the training error, average cross-validation error, and testing error for all the models that met the estimated training sample size for the PAC learning guarantee. The results for the original data space are shown in table II below. We observe that the SVM linear model overfitted the data as both the training and average cross-validations errors were approximately the same and the testing error was higher. The same happened with both the Linear regr. and Logistic regr. models. The results for the low dimensional data space

Model	**Training error**	**Avg. cv error**	**Testing error**
SVM linear	0.03	0.03	0.07
Linear regr.	0.195	0.205	0.241
Logistic regr.	0.03	0.03	0.05

TABLE II: Training errors, average cross-validation errors, and testing errors for the models that met the estimated training sample size for the original data space.

obtained from the UMAP embedding are shown in table III below. We observe that the SVM linear model managed to generalize the data well, as it behavior improved on newer data. For both the Linear regr. and Logistic regr. models, the lower testing error compared to the training error could indicate some level of generalization, but caution should be exercised as the overfitting behavior observed on the validation set suggests that the model's performance may not be reliable on new, unseen instances. The results for the high dimensional

Model	**Training error**	**Avg. cv error**	**Testing error**
SVM linear	0.08	0.07	0.01
Linear regr.	0.303	0.319	0.222
Logistic regr.	0.11	0.12	0.02

TABLE III: Training errors, average cross-validation errors, and testing errors for the models that met the estimated training sample size for the low dimensional data space obtained from the UMAP embedding.

data space obtained from the autoencoder are shown in table IV below. We observe that the SVM linear model overfitted

the data as the error was higher on newer data. The same happened to both the Linear regr. and Logistic regr. models.

Model	Training error	Avg. cv error	Testing error
SVM linear	0.01	0.02	0.02
Linear regr.	0.159	0.182	0.186
Logistic regr.	0.01	0.01	0.02

TABLE IV: Training errors, average cross-validation errors, and testing errors for the models that met the estimated training sample size for the high dimensional data space obtained from the autoencoder.

V. Conclusions

Based on the results obtained from the estimated sample size for the PAC learning guarantee, it is evident that our limited dataset of only 200 observations poses a challenge in meeting the desired criterion. However, among the models considered, SVM linear, Linear regression, and Logistic regression showed promise in meeting the PAC learning guarantee with error parameters of $\epsilon = \delta = 0.1$. Therefore, our focus was directed towards these models and error parameters for further analysis.

In our investigation of the training error, average cross-validation error, and testing error for the models that met the estimated training sample size, we observed interesting patterns across the different data spaces: the original space, UMAP embedding, and autoencoded space.

For the original data space, it became apparent that the SVM linear model exhibited signs of overfitting, as both the training and average cross-validation errors were similar and higher than the testing error. Similar behavior was observed for the Linear regression and Logistic regression models, indicating a lack of generalization to unseen instances.

Shifting our attention to the low-dimensional data space obtained from the UMAP embedding, we observed that the SVM linear model demonstrated better generalization, as its performance improved on unseen data. However, caution was advised for the Linear regression and Logistic regression models, as the lower testing error compared to the training error hinted at potential generalization but raised concerns due to the overfitting behavior observed during the validation phase.

Lastly, in the high-dimensional data space obtained from the autoencoder, we noticed a similar trend of overfitting for the SVM linear, Linear regression, and Logistic regression models, as the error increased on unseen instances.

In conclusion, our analysis highlights the importance of careful model selection and consideration of the specific data space in supervised learning tasks. The limited dataset posed challenges in meeting the desired PAC learning guarantee, but certain models exhibited promising behavior within the given error parameters. Further investigations and improvements are necessary to enhance the generalization capabilities of the models and overcome overfitting tendencies. By understanding the nuances of the data spaces and appropriately selecting models, we can pave the way for more effective and reliable learning machines in customer classification tasks.

References

[1] S. Cardeño, "Unsupervised learning: clustering methods on multidimensional spaces," *Workshop from AI course*, 2023.

[2] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[3] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[4] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, vol. 2. Springer, 2009.

[5] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, pp. 273–297, 1995.

[6] N. R. Draper and H. Smith, *Applied regression analysis*, vol. 326. John Wiley & Sons, 1998.

[7] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*, vol. 398. John Wiley & Sons, 2013.

[8] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[9] N. Rengifo, "Supervised learning, practical work artificial intelligence," *Workshop from AI course*, 2022.