UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Tecnura

https://revistas.udistrital.edu.co/ojs/index.php/Tecnura/issue/view/1214

DOI: https://doi.org/10.14483/22487638.22094

INVESTIGACIÓN

Application of machine learning for predictions of consecutive dependent data of type $\{[(a, b) \rightarrow c] \rightarrow d\}$

Aplicación de machine learning para predicciones de datos dependientes consecutivos de tipo $\{[(a, b) \rightarrow c] \rightarrow d\}$

Diego Alexander Quevedo Piratova [1], Jhon Uberney Londoño Villalba [2], Arnaldo Andres Gonzalez Gomez [3]

Fecha de Recepción: 30 de enero de 2023 **Fecha de Aceptación**: 23 de abril de 2024

Cómo citar: Quevedo-Piratova., D.A. , Villalba-Londoño., J.U. y Gonzalez-Gomez., A.A (2022). Application of machine learning for predictions of consecutive dependent data of type {[(a, b) → c] → d} *Tecnura*, **28**(79), 66-86. https://doi.org/10.14483/22487638.22094

ABSTRACT

Objective: Machine learning techniques have emerged in response to the desire for automatic pattern detection within datasets in fields such as statistics, mathematics, and data analytics. They allow for the extraction of relevant information from datasets of significantly large volumes, providing the possibility of making predictions. This paper presents an application focused on decision trees, linear regression, and random forest regression algorithms to predict final data from consecutive dependent data of type $\{[(a, b) \rightarrow c] \rightarrow D\}$.

Methodology: The study adopts a quantitative research design, which takes as input datasets based on interval data. It utilizes a correlational research model by implementing Python and its Scikit-Learn library, which includes various algorithms for prediction. Specifically, we compare the application of decision trees, linear regression, and random forest regression on the same set of datasets, but with a characteristic of dependency between them.

Results: Upon application of the proposed model, it yields an estimated prediction score, which indicates the accuracy of the model concerning the data provided.

[1]Magister in Educational Technology, Magister in Innovative Media for Education. Graduated in Technological Design, systems engineering student. Research professor of the AXON research group attached to the systems engineering program of the engineering school of the National Unified Corporation for Higher Education CUN. Bogotá Colombia.
Email: diego_quevedo@cun.edu.co

[2]Master in Educational Technology Management, Specialist in Virtual Learning Environments. Graduated in Technological Design, systems engineering student. Research professor of the AXON research group attached to the systems engineering program of the engineering school of the National Unified Corporation for Higher Education CUN. Bogotá Colombia.
Email: jhon.londono@cun.edu.co

[3]Electronic Engineer graduated from the Francisco José de Caldas District University, specializing in Data Analytics. Research professor of the AXON research group assigned to the systems engineering program of the engineering school of the National Unified Corporation for Higher Education CUN . Bogotá Colombia.
Email: arnaldo_gonzalez@cun.edu.co

Conclusions: The application of a complex algorithm does not inherently guarantee a higher rate of accuracy. Conversely, configuring the model correctly, training multiple trees, or adjusting parameter values can significantly enhance the obtained results
Financing: Unified National Corporation for Higher Education (CUN).

Keywords: algorithms, datasets, decision trees, Python, prediction, Scikit-Learn, linear regression

RESUMEN

Objetivo: Las técnicas de Machine Learning surgen como una respuesta al deseo de detectar automáticamente patrones en un conjunto de datos (*datasets*) en campos como la estadística, la matemática y la analítica de datos, permitiendo extraer información relevante de *datasets* de volúmenes significativamente grandes y realizar predicciones. Éste artículo presenta una aplicación enfocada en los algoritmos de árboles de decisión, regresión lineal y regresión aleatoria de tipo bosque para predecir un dato final a partir de datos dependientes consecutivos de tipo $\{[(a,b) \rightarrow c] \rightarrow D\}$.
Metodología: Se parte de un diseño de investigación cuantitativo, que toma como insumo unos *datasets* basados en datos de intervalo, establecidos en un modelo de investigación correlacional al aplicar Python y su librería Scikit-learn. Esta biblioteca incluye diferentes algoritmos que pueden ser utilizados para realizar predicciones. En este caso, se compara la aplicación de árboles de decisión, regresión lineal y regresión aleatoria de tipo bosque sobre un mismo grupo de *datasets*, pero que tienen una característica de dependencia entre ellos.
Resultados: Cuando se aplica el modelo propuesto, este genera un puntaje estimado de la predicción, el cual indica la precisión del modelo respecto a los datos entregados.
Conclusiones: La aplicación de un algoritmo complejo no garantiza un mayor índice de precisión; por el contrario, configurar de manera correcta el modelo, entrenando múltiples árboles o cambiando los valores de los parámetros mejora en gran medida los resultados obtenidos.
Financiamiento: Corporación Unificada Nacional de Educación Superior (CUN).
Palabras clave: Python, Scikit-Learn, algoritmos, predicción, árboles de decisión, regresión lineal, *datasets*

INTRODUCTION

The desire for automatic pattern detection in datasets has long beenpervasive in fields such as statistics, mathematics, and data analytics. Machine Learning emerges as a solution to fulfill this desire, as it allows for the extraction of relevant information in datasets of considerable size while providing prediction possibilities that range from simple classification to clairvoyance or other mystical art. In this context, Python and its Scikit-Learn library employ various machine-learning algorithms for predictive data analysis. This paper focuses on decision trees, linear regression, and Random Forest Regression algorithms to predict final data from consecutive dependent data of type $\{[(a,b) \rightarrow c] \rightarrow D\}$. The predictive performance of each model is assessed based on the dataset and the strategy presented to the algorithms.

Definition

Depending on the field and the data's origin, datasets can be subjected to various analyses using different tools to tackle a wide array of problems. In this study, supervised learning is applied, wherein "the machine is trained with labeled data as an example, thus providing a learning guide" (Maisueche Cuadrado, 2019, p. 13). Within the domain of supervised learning, there are multiple regression and classification algorithms. This article specifically delves into decision tree, linear regression, and Random Forest regression algorithms for comparative forecasting. Decision trees are widely utilized for their simplicity and stability, despite their extreme rigidity, being recognized as "models of supervised inductive learning, non-parametric, most used as a form of representation of knowledge" (Segura Cardona, 2012, p. 99). Linear regression algorithms, on the other hand, present a more intricate understanding and application due to its flexibility, which can yield inconsistent responses. Finally, Random Forest regression algorithms often demand higher resource consumption, due to their robust calculations and variants. This study aims to derive insights from consecutive dependent datasets of the type $\{[(a,b) \rightarrow c] \rightarrow D\}$, where the initial prediction C is integrated with the initial dataset ab to arrive at a final prediction D. This can be extrapolated to larger datasets with similar dependent relationships, altering not the complexity of the problem, but the way the data is modeled.

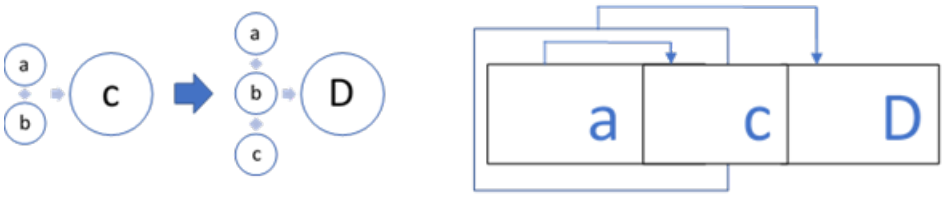


Figura 1. Consecutive dependent dataset of type $\{[(a,b) \rightarrow c] \rightarrow D\}$.

Source: Own work.

Justification

The traditional way to approach a final prediction problem involves associating an item, product, or category with a series of values from a database, which are then used to obtain the final forecast data. Although efficient, this approach requires having all the data before attempting any predictions. Consider a scenario where the independent arrival of two components affects the fi-

nal delivery time of a product, which in turn causes the final selling price to increase or decrease. $\{[(\text{Time 1}, \text{Time 2}) \rightarrow \text{Time 3}] \rightarrow \text{Price}\}$. Another scenario could involve the interconnection of various tasks within a subject whose contents are prerequisites for the following activities. While the tasks themselves are not analyzed, their collective outcomes can be validated, noted, and used to forecast results$\{[(\text{Note 1}, \text{Note 2}) \rightarrow \text{Note 3}] \rightarrow \text{Final Note}\}$. Both scenarios can be generalized to multiple datasets concatenated in analogous ways.

In this context, multiple linear regression " tries to fit linear or linearizable models between a dependent variable and more than a few independent variables" (Montero Granados, 2016, p. 1). Meanwhile, decision trees are described as a "non-parametric technique [that] classifies a population in a model of branch-type segments that build an inverted tree; this model is then used to predict an objective variable" (Song & Ying, 2015, p. X). Lastly, Forest-type Random Regression algorithms apply multiple decision trees with slight variations, as they are considered "a combination of tree predictors where each tree depends on values from a random vector sampled independently and with the same distribution across all trees in the forest " (Breiman, 2001, p. X). Any of these Machine Learning algorithms could be applied to address the issues discussed in this article.

Applicability and structure

Linear regression aims to determine a 'D' value that fits a previously calculated 'straight line' based on data provided to the algorithm. It involves predicting a dependent variable using one or more independent variables 'a' by drawing the straight line that best fits the dataset (Maisueche Cuadrado, 2019). Taking into account that the prediction is not $100\,\%$ accurate, as the " sample is an estimate of the model parameters" (Cardona *et al.*, 2014, p. X), adjustments 'a' must be made in each new cycle to reduce prediction errors, a process known as minimizing the cost function. Equation (1) represents this model:

$$D(x) = a_0 + a_1x_1 + a_2x_2 + \ldots + a_nx_n \tag{1}$$

A decision tree learns decision rules derived from features in the dataset, where " the algorithm performs successive binary partitions in the space of the explanatory variables. In each partition, the algorithm selects the variable that provides the most information, based on a measure of entropy or amount of information" (Díaz Martínez *et al.*, 2004).

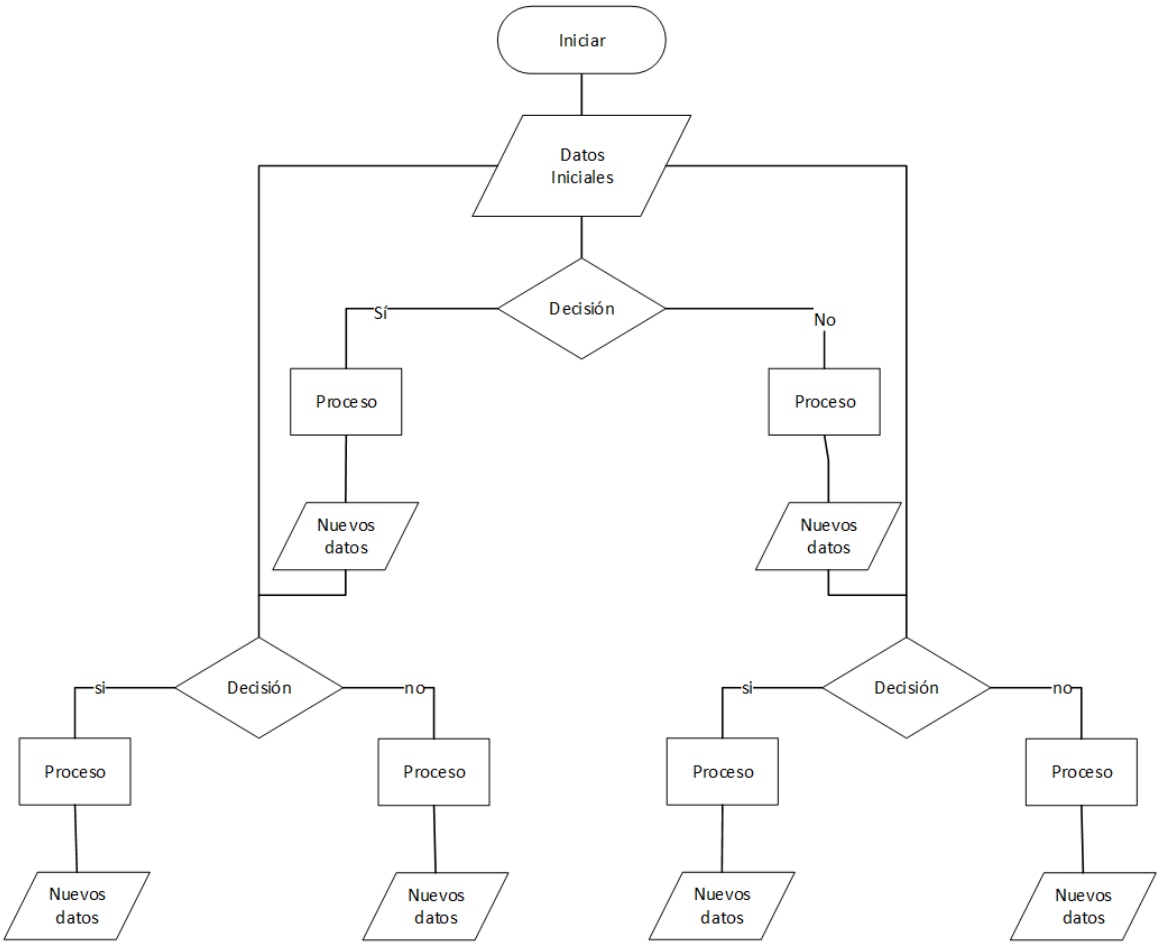


Figura 2. Decision tree structure with dependent data.

Source: own work.

A Forest-type Random regression algorithm, or simply Random Forest Regressor, extends the usage of decision trees by aggregating many of them. This ensemble approach allows for more precise predictions but also consumes significantly more resources, since it is formed "by an algorithm that introduces randomness to reduce the correlation between the trees. Once the forest is built, it can be used to make predictions" (García Ruíz de León *et al.*, 2018, p. 23).

METHODOLOGY

This study adopts a quantitative research design that utilizes datasets based on interval data, accessible through the following link: https://acortar.link/HUZ7WN. The research design is

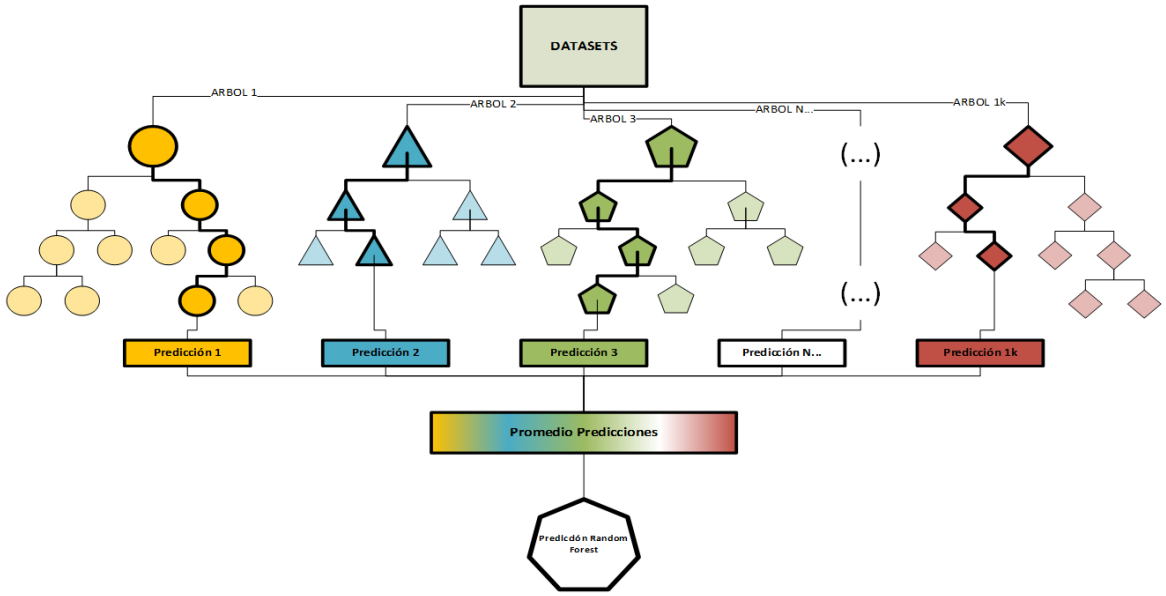


Figura 3. Forest type random regression structure - Random Forest.

Source: own work.

based on a correlational research model, implementing the Python Scikit-Learn library, which offers various algorithms for predictive modeling. Specifically, this study compares the implementation of decision trees, linear regression, and Random Forest Regression on the same group of datasets, which exhibit interdependencies. The evaluation focuses on each model's predictive capacity and the strategies employed by the algorithms. Identifying patterns within datasets and making informed decisions to address problems are fundamental goals of Machine Learning (ML). According to Bell (2015) a ML can be formally defined as a branch of artificial intelligence where "systems can learn and improve with experience and, over time, refine a model that can be used to predict question outcomes based on previous learning."

ML algorithms fall into two possible categories based on how the data is or is not catalogued:

- Supervised learning involves incorporating labels in the dataset. They allow " patterns to be detected and used to label new sets of information " (Hinestroza Ramírez & Cárdenas, 2018).
- Unsupervised learning, in contrast, does not rely on labels in the dataset, which complicates the search for patterns. As a result, "there is no right or wrong answer; it is just about running the machine learning algorithm and seeing what patterns and results show up " (Bell, 2015).

The workflow in ML has been evolving, going from a basic structure comprising four levels (dataset conditioning, representation, learning, and evaluation of the model) to proposals tailored to specific industries of eight or more ranges. Academics projects differ significantly from those in industry. Therefore, merely selecting and training a model is insufficient. Similarly to any project, a well-defined series of stages or steps must be followed to enhance the likelihood of success (Maisueche Cuadrado & Sanz Angulo, 2019).

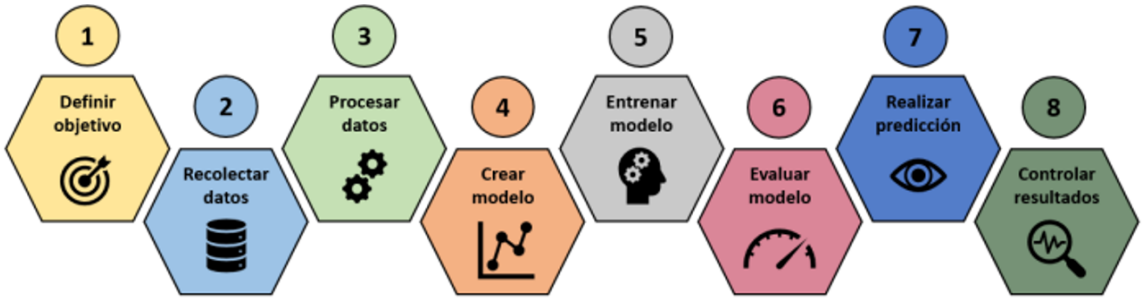


Figure 4. Generic stages to carry out a Machine Learning project.
Source: Maisueche Cuadrado and Sanz Angulo (2019)

The project uses two datasets in Excel, examples of which are presented in Tables 1 and 2. These datasets contained consecutive dependent labeled data of type $\{[(a,b) \rightarrow c] \rightarrow D\}$, where "c" and "D" are calculated differently from "a" and "b" which are the same in both data sets. ML has been worked on by multiple companies and institutions dedicated to data analytics. This has allowed for the development of various libraries and frameworks focused on ML. Among the most popular are Tensor Flow (Google Brain Team), Pythorch (Facebook's AI Research Lab – FAIR), Keras (François Chollet – Google) and Scikit-Learn (David Cournapeau).

In this study, Scikit-Learn library is employed to analyze the dataset, mainly because it is the easiest and simplest to implement. It consistently achieves excellent results, at the same time while demanding fewer computational resources compared to other alternatives. As noted by Feurer and Hutter (2019, p. 3), " each machine learning system has hyperparameters, and the most basic task in automated machine learning (AutoML) is to automatically configure these hyperparameters to optimize performance" Any such analysis requires importing the necessary libraries.

Although there are other libraries that you must include such as pandas, numpy , math , xgboost or seaborn , the previous ones are the ones specific to the models used and their corresponding

Cuadro 1. Extract Dataset 1

a	b	c	D
80	51	57	65
80	51	57	65
9	85	12	68
1	69	69	2
15	3. 4	77	71
17	6	77	37
43	82	47	21
68	74	26	24
42	53	95	88
35	2	35	100

Source: own work.

Cuadro 2. Extract Dataset 2

a	b	c	D
80	51	65	5
9	85	47	26
1	69	35	98
15	3. 4	24	12
17	6	11	22
43	82	62	8
68	74	71	1
42	53	47	3
35	2	18	62

Source: own work.

Cuadro 3. Models and specific libraries needed

Model	Bookshop
Linear regression	**from** sklearn. linear _model **matter** LinearRegression
Decision tree	**from** sklearn. tree **matter** DecisionTreeClassifier
Forest-like random regression	**from** sklearn. outfit **matter** RandomForestRegressor

Source: own work.

graphing . The Dataset must be read immediately and a Dataframe created , in this case we will do it from an Excel file.

```
date = P.S. read _excel ( r './Dataset.xlsx' )
```

The train test split procedure is applied to validate the dataset and simulate how it would behave with new data. Two important parameters in this procedure are "test_size", which specifies the size of the test set, and random_state"which controls the randomness of data selection for training or testing. Adjusting the values of these parameters can impact the accuracy of the model and its subsequent

Cuadro 4. Libraries necessary to make the graphs of the data

Name	Graphic Library
dtreeviz	**from** dtreeviz. trees **import** *
IPython	**from** IPython. display **import** Image , display_svg , SVG
graphviz	**from** graphviz **matter** Digraph
matplot	**matter** matplotlib. pyplot **ace** plt

Source: own work.

predictions. The data used in the study, shown in the figures and results, indicate that increasing these values entails greater resource consuption, potentially obscuring the clarity of the displayed images . This is because by increasing these values, there is also greater resource consumption and the images would not be able to be displayed clearly. Note that this notation expresses the analysis only for the first part of [(a, b)->c]->D; the process must be repeated to analyze "D".

```
test = 0.25
random = twenty-one
X = data [ [ "to" , "b" ]]
y = data [ "c" ]
X_train , X_test , y_train , y_test = train_test_ split ( X , y
, test_size = test , random_state = random )
```

The max_depth parameter must also be specified for the decision tree model since it determines the maximum number of leaves or nodes to be computed. If a value is not placed, the analysis can run indefinitely, consuming high levels of resources.

```
depth = 4
```

The application of the model is the natural step after the previous configurations, each of which requires specific syntax

Cuadro 5. Implementation of models

Model	Code
Linear regression	regressor1 = LinearRegression () regressor1. fit (X_train , y_train)
Decision tree	decision1 = DecisionTreeRegressor (max_depth = depth) decision1. fit (X_train , y_train)
Forest-type random regression	regressor1 = RandomForestRegressor (n_estimators = 200 , random _state = 0) regressor1. fit (X_train , y_train)

Source: own work.

RESULTS

Applying the model returns an estimated prediction score that indicates the model's accuracy relative to the given data, with the maximum score being 1. In this case, the data taken to calculate are the training data, the process can also be carried out with the test data. The scores obtained after training the models for "c" and for "D" are shown in Table 6.

Cuadro 6. Implementation of models

Model	Code	C-score	D-score
Linear regression	score = regressor1. score(X_train , y_train)	0.99985200937484	0.18247241602165787
Decision tree	score = decision1. score(X_train , y_train)	0.9446927198245703	-2.640986255989985
Forest-type random regression	score = regressor1. score(X_train , y_train)	0.9997305877339204	0.9561890331775832

Source: own work.

Among the best tools available for analysis are graphs generated in different ways. They enable decision-making regarding adjustments to models, corrections to datasets, and even the selection of appropriate models.

Graphs obtained for linear regression

Code used to generate the graphs: The dispersion analysis shows a correlation between the "a" and "b" categories, which is further reinforced in the subsequent analysis. This reduces dispersion by incorporating the "b" category in the analysis.

```
pandas_ plot.scatter _matrix ( X_train , c= y_train , figsize =(10,10))
```

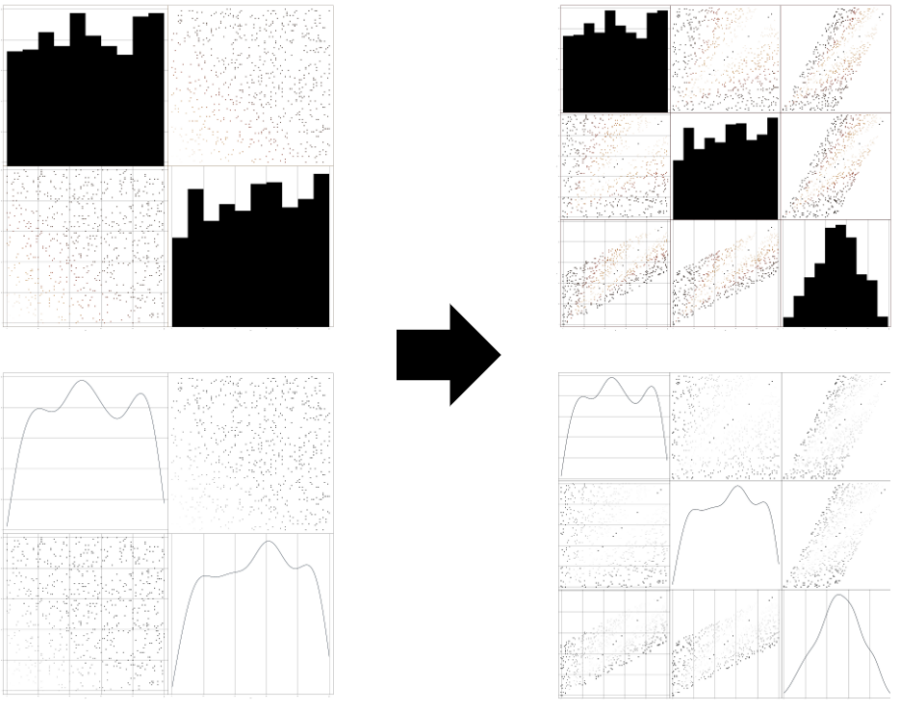


Figure 5. Scatter matrix of type scatter_matrix for "c" and "D" respectively.
Source: own work.

The prediction becomes more uncertain as the number of variables increases. Code used to generate the graphs:

```
visualizer = PredictionError (regressor1)
visualizer.fit ( X, y)
visualizer.score (X, y)
visualizer.show ()
```

Code used to generate the graphs:

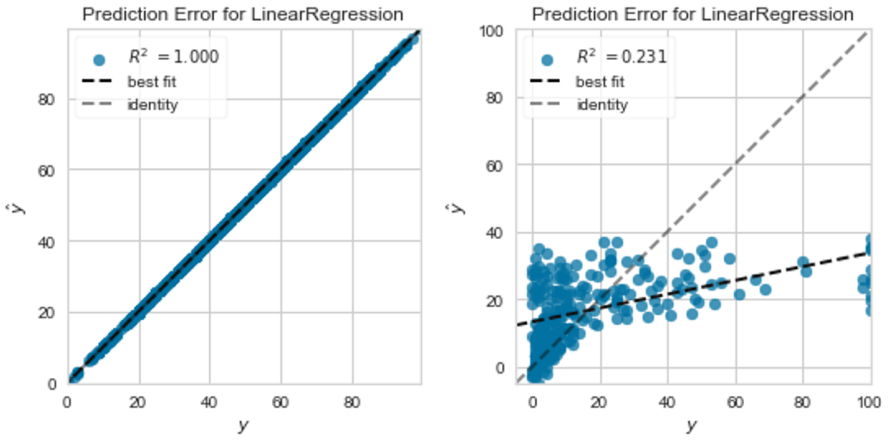


Figure 6. Plot using Prediction Error Plot for "c" on the left and for "D" on the right.

Source:own work.

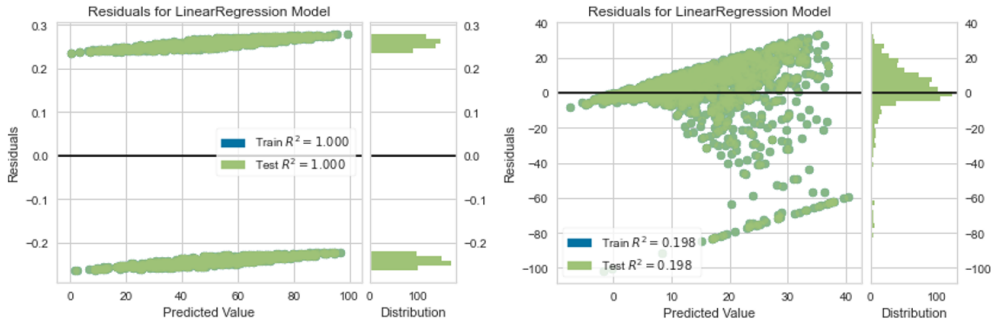


Figure 7. Plot using Residuals Plot for "c"on the left and for "D" on the right.

Source: own work.

```
visualizer = ResidualsPlot (regressor1)
visualizer.fit ( X, y)
visualizer.score (X, y)
visualizer.show ()
```

Decision Tree Charts

Code used to generate the graphs in Figure **9**:

```
fig = plt.figure ( figsize =(15,10))
_ = tree.plot _tree (decision2,
```

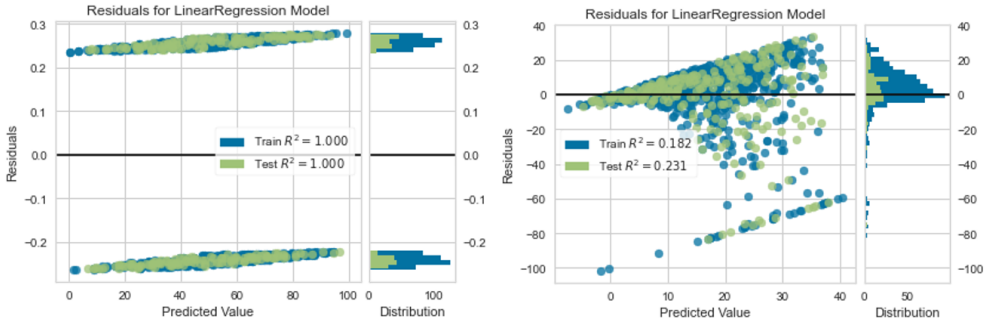


Figure 8. Plot using Residuals Plot for "ca" on the left and for "D" on the right - training data vs test data.

Source: own work.

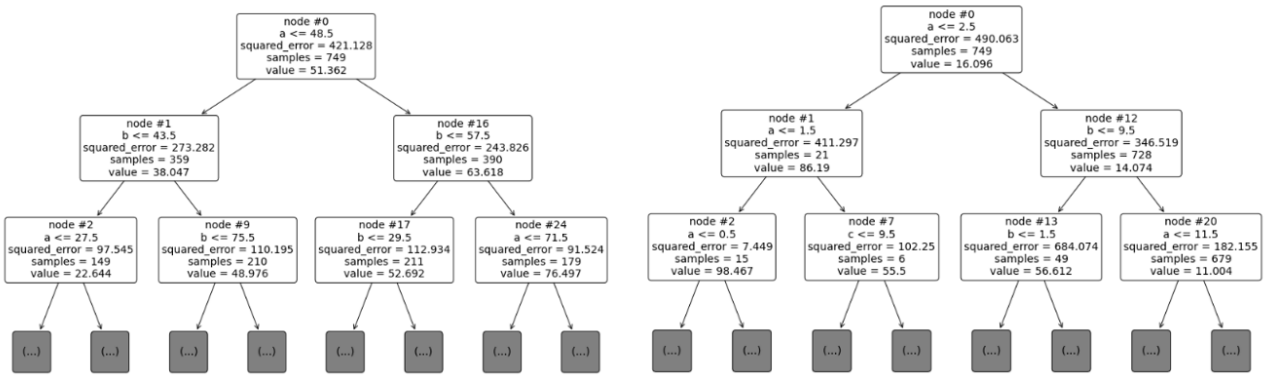


Figure 9. Plot using Pandas Plot for "c" on the left and for "D" on the right.

Source: own work.

```
                    feature_names = ["a", " b","c "],
                    class_names = "D",
                    max_depth = 2,
                    node_ids = True,
rounded=True,
                    fontsize =14,
                    filled =False)
```

Code used to generate the graphs in Figures **10** and **11**:

```
viz = dtreeviz ( decision1,X,Y,
                target_name = 'c',
```

```
                feature_names =[ "a", "b"],
X = X.iloc [51],
                show_node_labels = False,
fancy=True)
viz
```

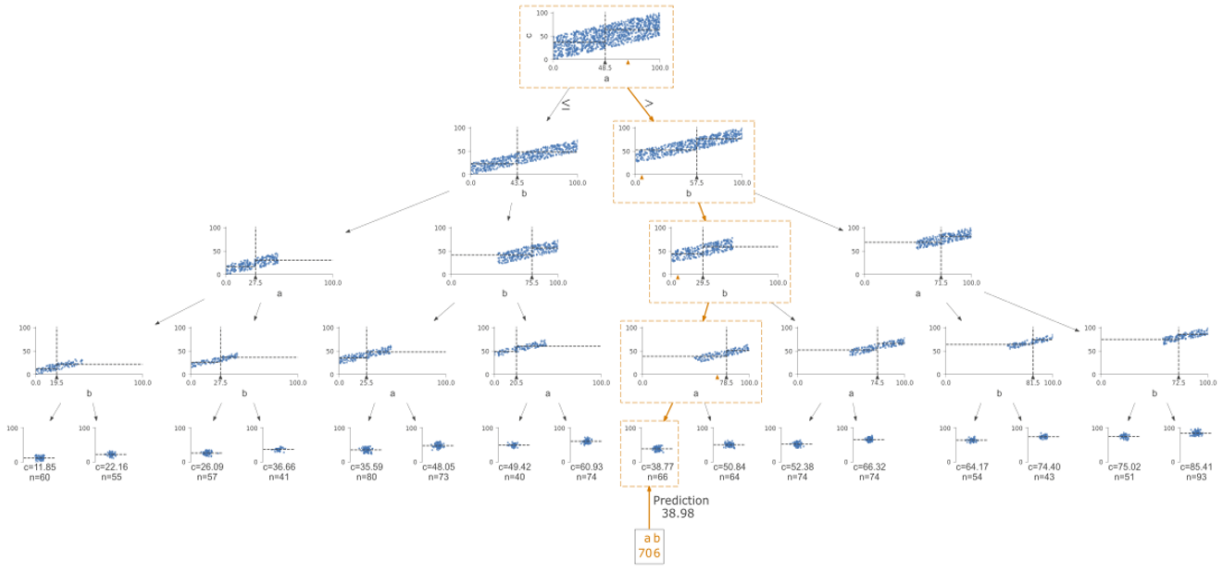


Figure 10. Plot using dtreeviz Plot for "c".

Source: own work.

Code used to generate Figure **12**:

```
figsize = (6.5)
fig = plt.figure ( figsize = figsize )
ax = fig.add_ subplot ( 111, projection='3d')
t = rtreeviz_bivar_3 D( decision1,x,y,
                        feature_names =[ "a", "b"],
                        target_name = 'c',
                        fontsize =14,
                        raise =20,
                        azim =25,
```

```
                        dist =8.2,
show={' splits','title '},ax = ax )
plt.show ()
```

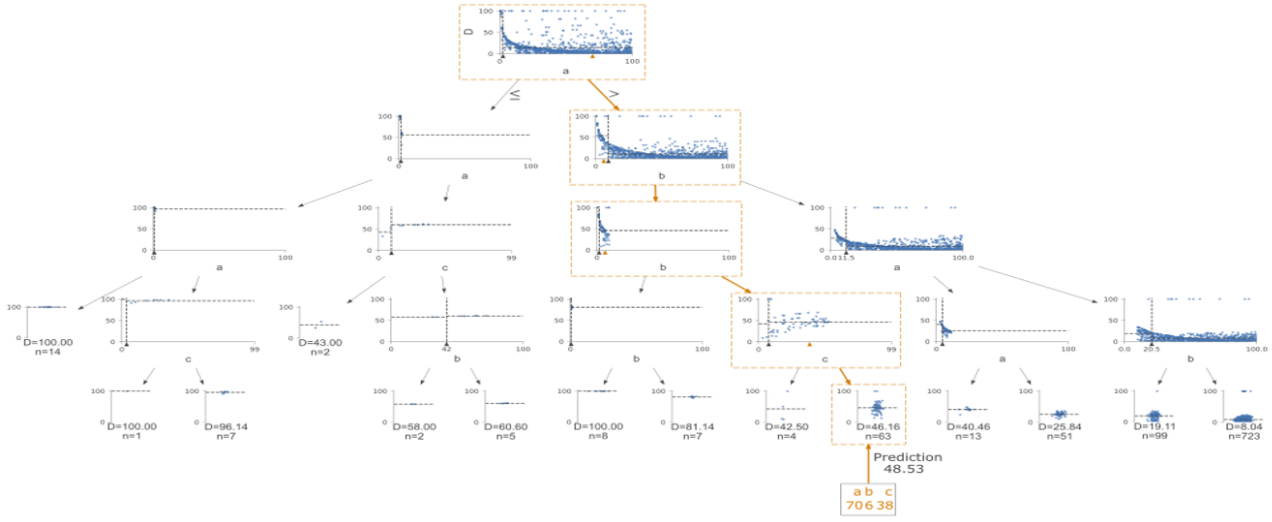


Figure 11. Plot using dtreeviz Plot for "D"

Source: own work

Code used to generate Figure **13**

```
t = rtreeviz_bivar_ heatmap ( decision1,x,y,
                              feature_names =[ "a", "b"],
                              fontsize =14)
plt.show ()
```

Plots for forest-like random regression only for tree 0 of N generated

Code used to generate the graphs **14** and **15**:

```
viz = dtreeviz (regressor1[0],X,Y,
                target_name = 'c',
                feature_names =[ "a", "b"],
X = X.iloc [51],
```

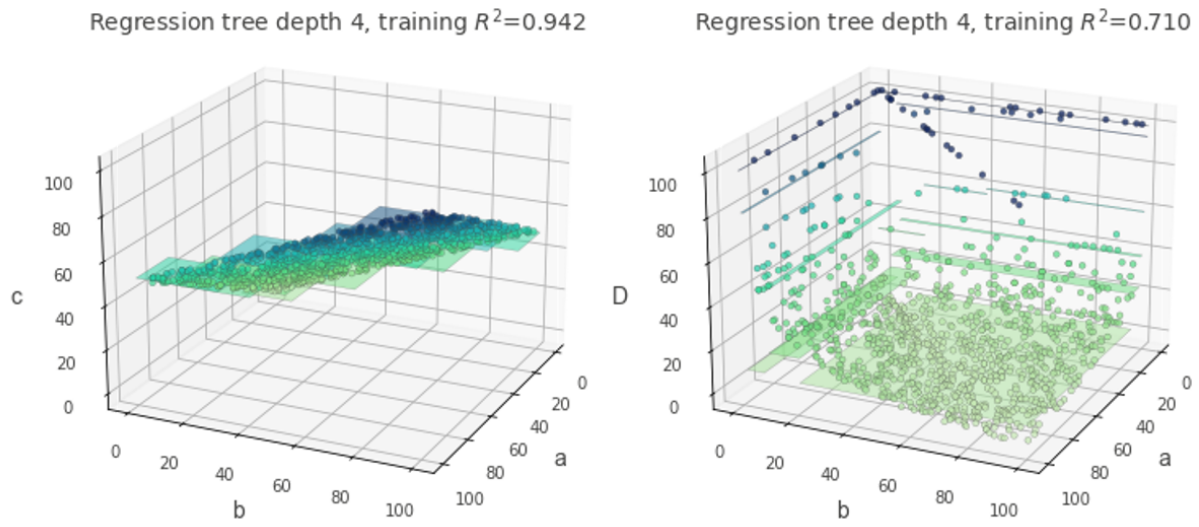


Figure 12. Plot using dtreeviz rtreeviz_bivar_3D Plot for c

Source: own work.

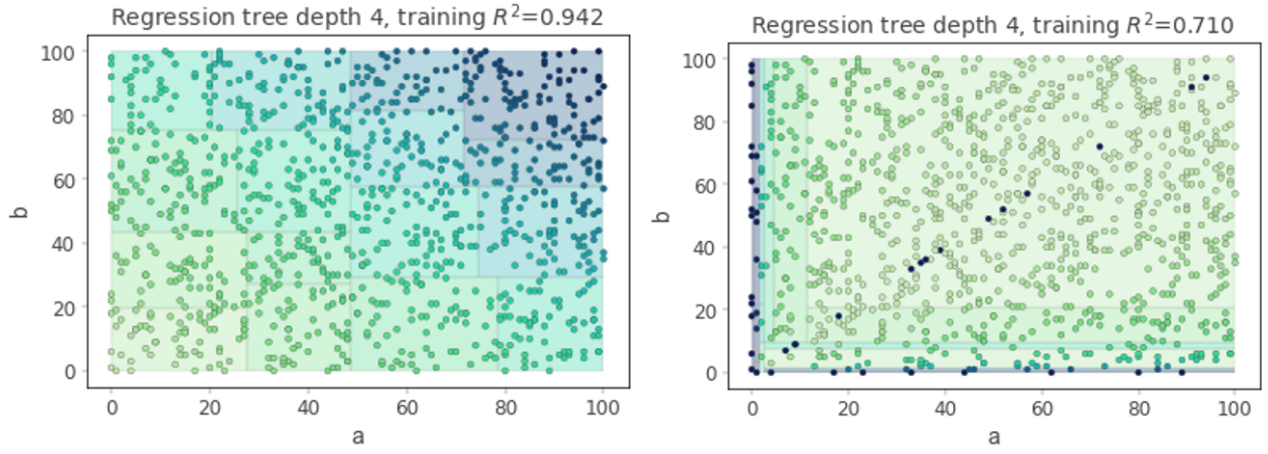


Figure 13. Plot using dtreeviz rtreeviz_bivar_heatmap Plot for D

Source:own work.

```
                show_node_labels = False,
fancy=True)
viz
```

Code used to generate Figure **16**:

```
figsize = (6.5)
```

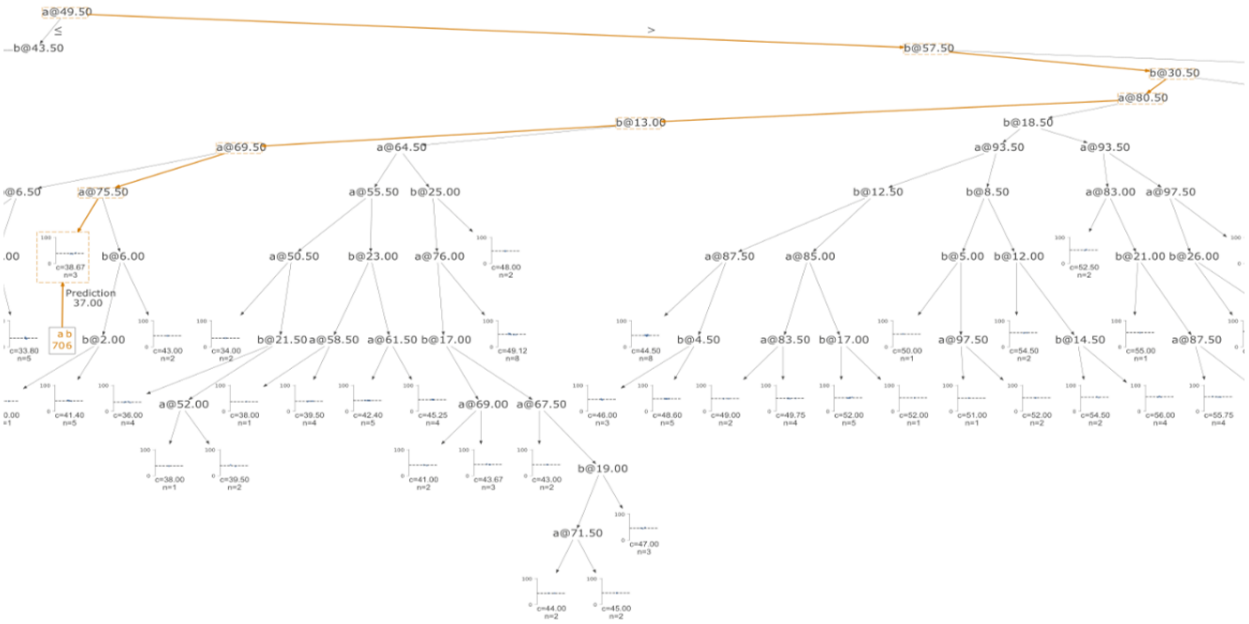


Figure 14. Partial plot using dtreeviz for "c"

Source: own work.

```
fig = plt.figure ( figsize = figsize )
ax = fig.add_ subplot ( 111, projection='3d')
t = rtreeviz_bivar_3D(regressor1[0],x,y,
                      feature_names =[ "a", "b"],
                      target_name = 'c',
                      fontsize =14,
                      raise =20,
                      azim =25,
                      dist =8.2,
show={' splits','title '},ax=ax)
```

Code used to generate Figure **17**:

```
t = rtreeviz_bivar_heatmap (regressor1[0],x,y,
                            feature_names =[ "a", "b"],
                            fontsize =14)
```

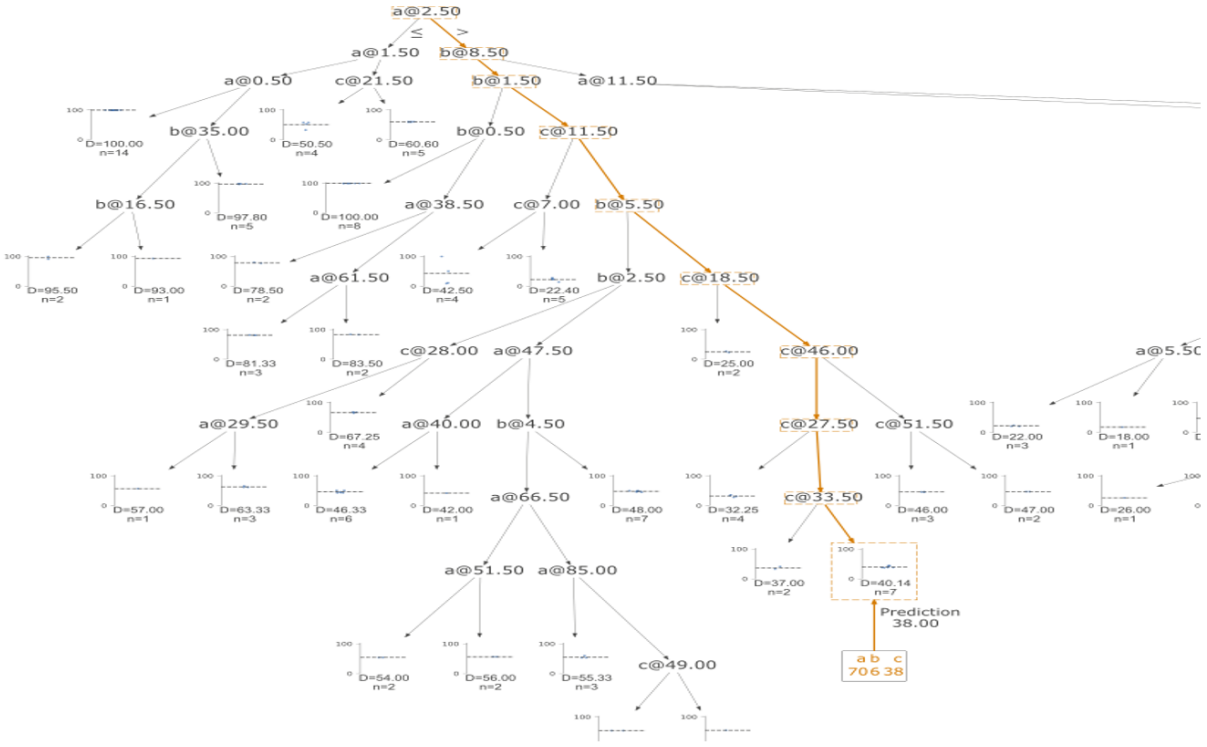


Figure 15. Partial plot using dtreeviz for "D"

Source: own work.

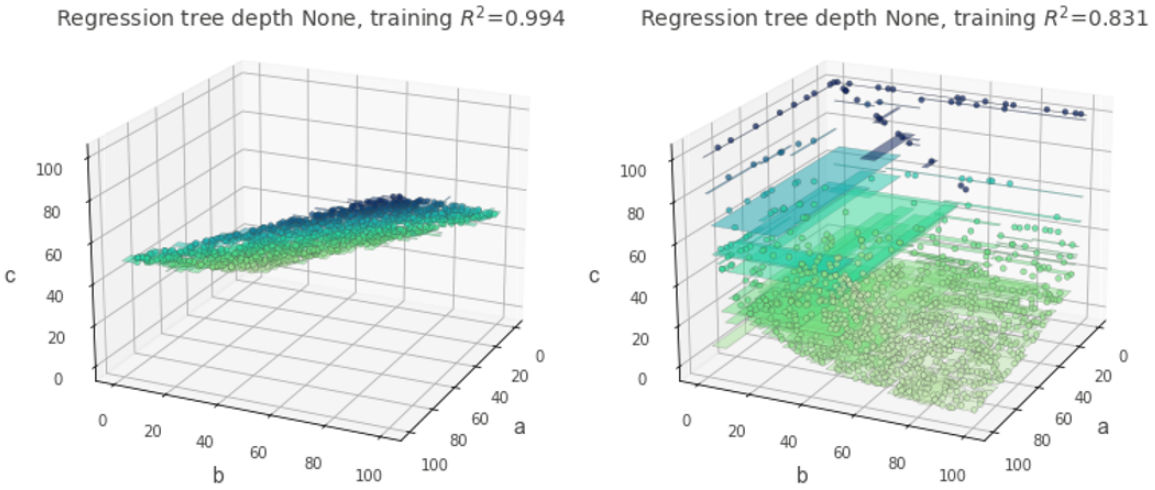


Figure 16. Plot using dtreeviz rtreeviz_bivar_3D Plot for "c" on the left

Source: own work.

```
plt.show ()
```

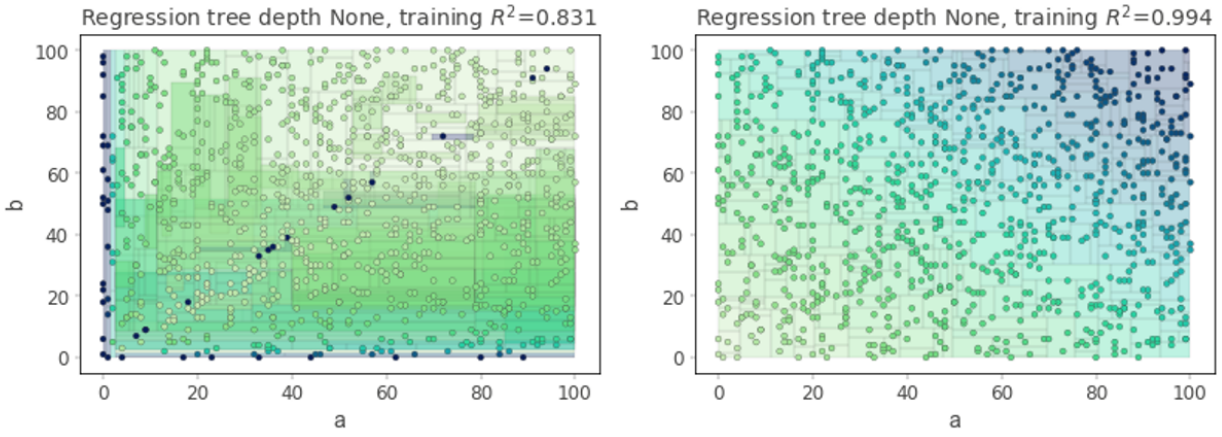


Figure 17. Plot using dtreeviz rtreeviz_bivar_3D Plot for "D"

Source: own work.

CONCLUSIONS

Predictive models are used across many fields for their split approach, facilitating the identification of solutions based on different conditions using the classification or regression methods. Predictive models enable rapid evaluation of multiple options, especially when visual representations of results can be generated, allowing algorithmic decisions to be visually monitored. This process defines critical paths from node to node that influences the final decision.

Decision trees implicitly perform diagnostic variable testing or feature selection. Classification or regression methods require minimal effort from from users to prepare the data.

The consumption of computing resources increases as the model's precision improves. However, a complex algorithm does not guarantee a higher rate of precision effectiveness. Instead, optimal model configuration, training multiple trees, or adjusting parameter values significantly enhances results.

Balancing the dataset before use is essential to avoid applying models with biased data.

FINANCING

The article derives from the research project: " Application of machine learning for predictions of consecutive dependent data of type $\{[(a,b) \rightarrow c] \rightarrow d\}$ " endorsed and financed by the Unified National Corporation for Higher Education (CUN).

REFERENCES

Bell, J. (2015). Machine learning Hands-On for Developers and Technical Professionals. *Indiana: Wiley.*

Breiman, L. (2001). Random Forest. California. *University of California.* https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf

Cardona, D., Rivera, M., González, J., & Cárdenas, E. (2014). Estimación y predicción con el modelo de regresión cúbica aplicado a un problema de salud. *Ingenieria Solidaria, 10(17).* https://doi.org/10.16925/in.v9i17.828

Díaz Martínez, Z., Fernández Menéndez, J., & Segovia Vargas, J. (2004). Sistemas de inducción de reglas y árboles de decisión aplicados a la predicción de insolvencias en empresas aseguradoras. *Departamento de Economía Financiera y Contabilidad / Departamento de Organización de Empresas. Madrid: Universidad Complutense de Madrid.* https://eprints.ucm.es/id/eprint/6833/

Feurer, M., & Hutter, F. (2019). Hyperparameter Optimization. *In Automated Machine Learning (pp. 3-33). Springer.* https://doi.org/10.1007/978-3-030-05318-5

García ruiz de León, M., Mira McWilliams, J. M., & Ahrazem Dfuf, I. (2018). Análisis de sensibilidad mediante Random Forest. *Madrid: Universidad Politécnica de Madrid.*

Hinestroza Ramírez, D., & Cárdenas, J. M. (2018). El Machine Learning a través de los tiempos, y los aportes a la humanidad. *Pereira: Universidad Libre.*

Maisueche Cuadrado, A. (2019). Utilización del Machine Learning en la industria 4.0. *Valladolid: Universidad de Valladolid. Escuela de Ingenierías Industriales.* http://uvadoc.uva.es/handle/10324/37908

Maisueche Cuadrado, A. (2019). Montero Granados, R. (2016). Modelos de regresión lineal múltiple. *Granada, España: Universidad de Granada.* http://www.ugr.es/~montero/matematicas/regresion_lineal.pdf

Segura Cardona, A. M. (2012). Aplicación de árboles de decisión en la salud pública (Implementation of decision trees in public health) (Aplicação de árvores de decisão em saúde pública). *Revista CES salud pública, 3(1), 94-103.* http://dx.doi.org/10.21615/2140

Song, Y.-Y.,& Ying, L. (2015). Decision tree methods: applications for classification and prediction. *Shanghai Arch Psychiatry, 27(2), 130-135.* https://doi.org/10.11919/j.issn.1002-0829.215044