Introduction

Data and network mining are becoming two critical fields for gaining information from great amounts of data. As more organisations base their decisions on information, the need to solve, estimate, and forecast utilising mathematical methods is crucial. This paper discusses some Data mining tools and techniques which are implemented within the CRISP-DM (Cross-Industry Standard Process for Data Mining). The subject of the analysis is a rich dataset on HRs in order to solve the problem of employee attrition that poses a problem for organizations interested in retaining talents.

Using machine learning algorithms in Python as well as RapidMiner's Auto Model for performance comparison, the report illustrates the logical flow of an actual data mining operation, which comprises data acquisition, pre-processing, modeling, and assessment. This two-way assessment facilitates the examination of the functioning of multiple machine learning models besides evaluating applicability and most informative features. Using SMOTE for addressing class imbalance problem and using tree based algorithms, deep learning and ensemble methods makes the comparison more reliable. In doing so, this report not only points out patterns in turnover and departing employees but equally sets standards on data mining with new state of art tools.

As a result, this report considers the CRISP-DM approach and utilizes Python and RapidMiner Auto Model. Data ingress and processing, model deployment and testing, and assessment of results relevant to employee attrition and prediction abilities of models are part of this project.

1. Business Understanding

The primary objective of this project is to understand and predict employee attrition within an organization. Attrition can negatively impact organizations due to increased costs in hiring, training, and decreased operational efficiency. By leveraging data mining techniques, we aim to:

1. Identify key factors influencing attrition.
2. Develop predictive models to classify employees likely to leave.
3. Evaluate and compare the effectiveness of machine learning models to ensure actionable insights.

Key questions:

- Which factors are most significant in employee attrition?
- How can machine learning models be used to predict attrition accurately?
- Which model provides the best balance between accuracy and interpretability?

2. Data Understanding

The dataset used is WA_Fn-UseC_-HR-Employee-Attrition.csv, containing 35 features and 1470 rows. It includes demographic, job-related, and performance-related attributes of employees, with a target variable Attrition (Yes/No). Key attributes include:

- **Age:** Employee age.
- **JobSatisfaction:** Job satisfaction level (1–4 scale).
- **MonthlyIncome:** Monthly salary.
- **OverTime:** Whether the employee works overtime (Yes/No).

Data Overview

- **Dataset Shape:** 1470 rows, 35 columns.
- **Target Distribution:** The dataset is imbalanced, with attrition cases (Yes) being less than non-attrition cases (No).
- **Data Types:**
    - Numerical: Age, DistanceFromHome, MonthlyIncome, etc.
    - Categorical: Department, Gender, JobRole, etc.

Initial Observations:

- No missing values.
- A few categorical variables such as Department and OverTime need encoding.
- Dataset exhibits class imbalance (≈1:5 ratio of Yes:No in Attrition).
- Numerical features vary in scale, requiring standardization.

3. Data Preparation

Preprocessing Steps

1. **Duplicate Removal:** No duplicate rows were found in the dataset.
2. **Encoding Categorical Variables:**
    - Label encoding was applied to categorical features using Python's LabelEncoder.
    - Features such as OverTime (Yes/No) were encoded as binary.
3. **Feature Scaling:**

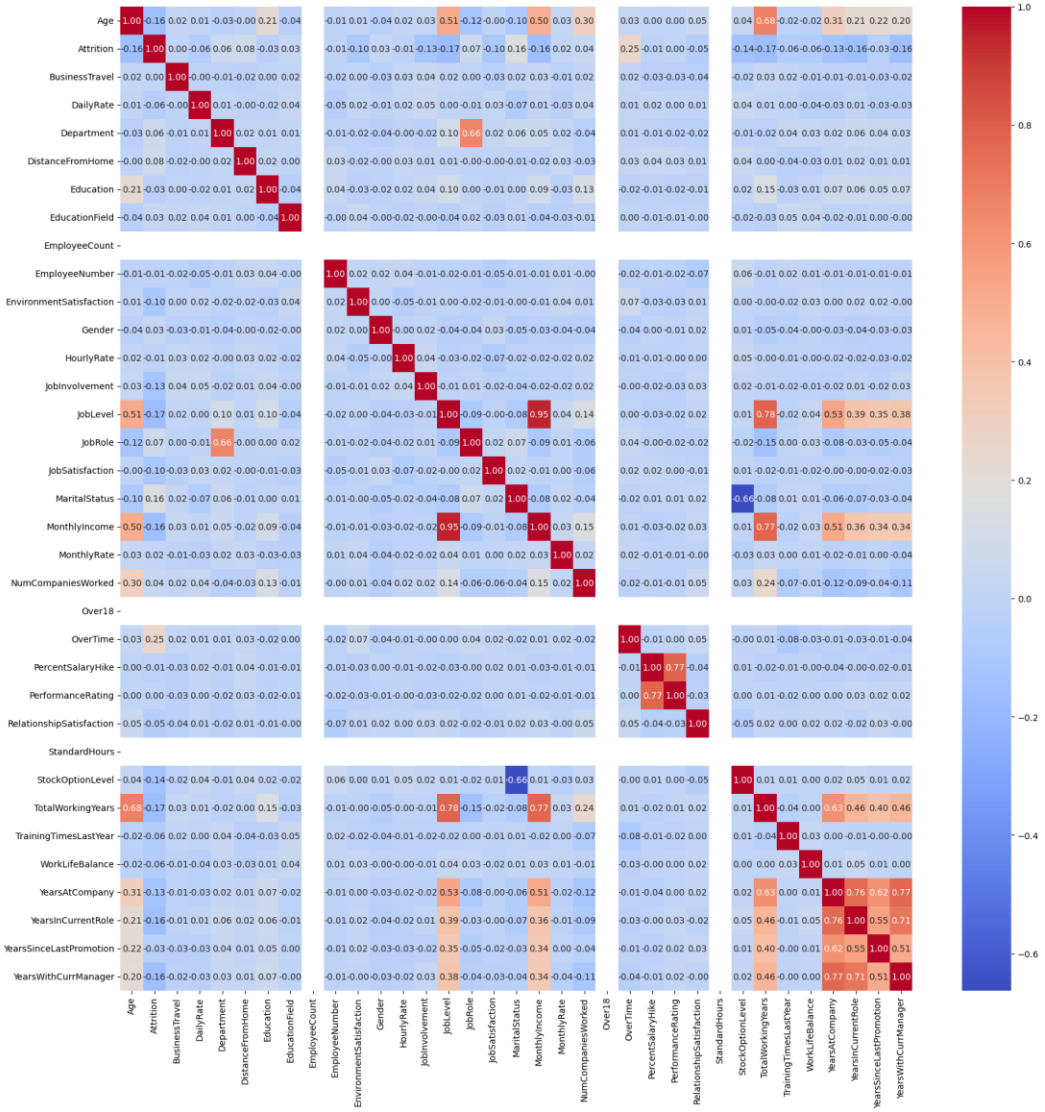
- Numerical features were standardized using StandardScaler to ensure consistent ranges.

4. **Handling Class Imbalance:**
   - Synthetic Minority Oversampling Technique (SMOTE) was applied to balance the dataset.

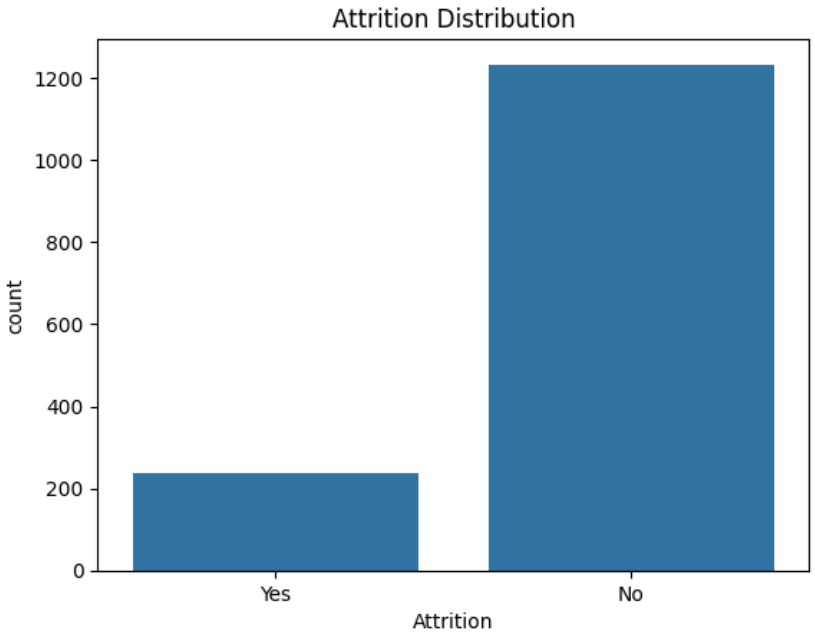
Correlation Analysis

A heatmap was generated to analyze correlations between features. Significant correlations include:

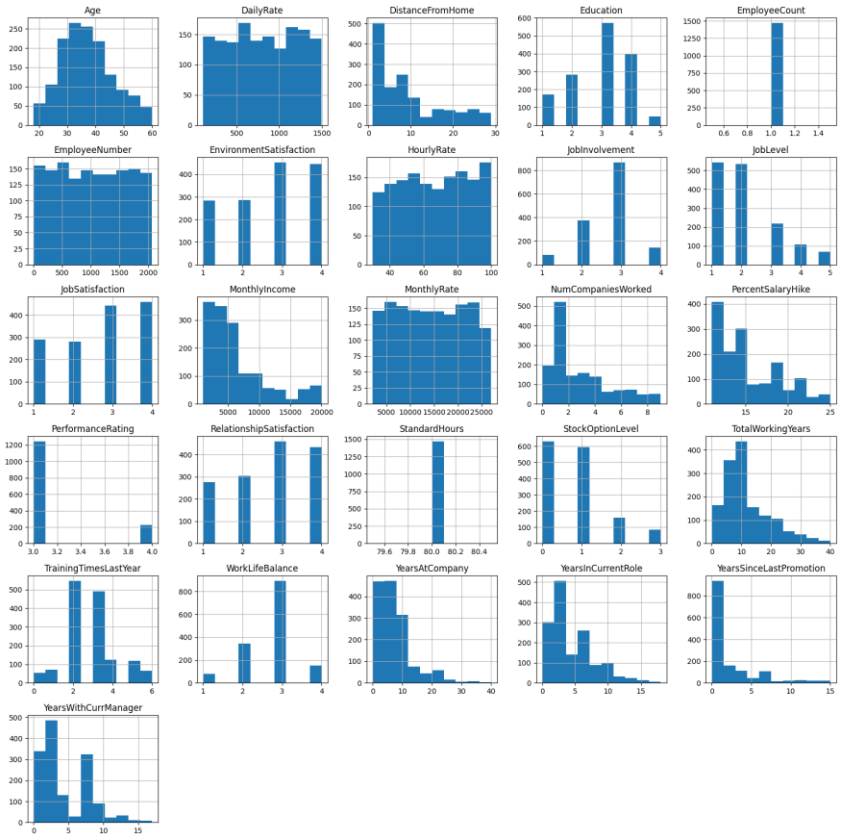
- MonthlyIncome and JobLevel.
- JobSatisfaction and WorkLifeBalance (moderate correlation).
- Attrition shows higher correlations with Overtime and JobSatisfaction.



Distribution of Attrition



Distribution of Numerical Features



4. Modeling

Machine Learning Models in Python

Seven machine learning models were developed and evaluated using Python. Below is an overview of the models:

1. **Naïve Bayes:**
    - Assumes independence among predictors.
    - Quick and interpretable.
2. **Logistic Regression:**
    - Linear model for binary classification.
    - Offers feature interpretability through coefficients.
3. **Decision Tree:**
    - Non-linear, interpretable model.
    - Susceptible to overfitting.
4. **Random Forest:**
    - Ensemble of decision trees.
    - Handles non-linear relationships effectively.
5. **Gradient Boosting:**
    - Sequentially builds models to minimize error.
    - Effective in handling complex patterns.
6. **Support Vector Machine (SVM):**
    - Maximizes the margin between classes.
    - Requires kernel tuning.
7. **Neural Network (MLP):**
    - Multi-layer perceptron for non-linear problems.
    - Requires computational resources and tuning.

Model Evaluation Metrics

Each model was evaluated using:

- **Accuracy:** Proportion of correctly classified instances.
- **Confusion Matrix:** Breakdown of true/false positives and negatives.
- **Classification Report:** Precision, recall, and F1 score.

Results that shows the comparison of Python and RapidMiner

Model	Python Accuracies	Rapid Minor Accuracies
Naïve Bayes	0.72	0.72
Logistic Regression	0.81	0.77
Decision Tree	0.80	0.68
Random Forest	0.90	0.67
Gradient Boosting	0.88	0.84
Support Vector Machine	0.88	0.64
Neural Network	0.91	0.81

Key Observations:

- Gradient Boosting achieved the highest accuracy (84%), followed closely by Neural Networks and Random Forests.
- Decision Tree had lower accuracy, likely due to overfitting.
- Logistic Regression provided a balance between interpretability and performance.

5. Evaluation

Insights from Python Models

1. **Feature Importance (Random Forest):**
    - OverTime, MonthlyIncome, and JobSatisfaction were identified as top predictors of attrition.
    - Employees working overtime or with lower satisfaction are more likely to leave.
2. **Model Comparisons:**

- Gradient Boosting is robust but computationally intensive.
- Random Forest balances performance and interpretability.
- Neural Network is powerful but requires hyperparameter tuning for better results.

RapidMiner Auto Model Process

For this analysis, the IBM HR Analytics dataset was imported to Rapid Miner where the missing values were handled, the data normalized and the class was balanced. In these tasks, actual data preparation was performed using the RapidMiner tool while the ease of the activities was supplemented by the visual side of RapidMiner. Subsequently, Auto Model was used for creating forecasts in form of Logistic Regression, Decision Tree, Random Forest, and Gradient Boosting models. Each model was then trained and validated from cross-validation in order to preserve the stability of the outcomes. That is why using RapidMiner is beneficial due to the possibility of analyzing results and determining why a specific model performs well or not by using artifacts like confusion matrix, ROC curve, and feature importance chart. These outputs allowed for the assessment of the general performance of each model towards the prediction of employee attrition. For example, the Gradient Boosting model showed the outstanding performance, it had a high accuracy, and it also demonstrated how it deals with imbalance problems.

RapidMiner also ensure for explain ability which shows the contribution of each feature for prediction. This feature was especially helpful when testing different hypotheses, for example, about the correlation between such factors as job satisfaction, work-life balance, and number of years in the company and employee turnover. These were in line with the Python-based analysis forecasts of the manually developed machine learning pipelines of RapidMiner Auto Model.

Consequently, with RapidMiner integrating the steps of building the model and providing straightforward tools for validation and interpretational purposes, the analysis was expanded. Its integration was useful in supplementing the analysis built on top of Python-based data-deduced pipelines, and showed a proof of capability of automated ML in providing solutions to concrete problems.

To benchmark results, RapidMiner Auto Model was used:

1. **Dataset Import:**
    - The dataset was loaded into RapidMiner for preprocessing and modeling.
2. **Data Preprocessing:**
    - Auto Model automatically detected and handled class imbalance using oversampling techniques.
    - Correlation matrices were generated to identify feature relationships.
3. **Model Building:**
    - RapidMiner's Auto Model trained multiple models, including:

- Logistic Regression.
- Random Forest.
- Gradient Boosting.
- Deep Learning.
- Decision Tree
- Support vector Machine
- Naïve Bayes

- Models were optimized using default hyperparameter tuning.

4. **Evaluation Metrics:** Accuracy, ROC-AUC, and confusion matrices were used for evaluation.

Model	**Accuracy**
Naive Bayes	0.72
Generalized Linear Model	0.72
Logistic Regression	0.77
Fast Large Margin	0.64
Deep Learning	0.81
Decision Tree	0.68
Random Forest	0.67
Gradient Boosted Trees	0.84
Support Vector Machine	0.64

RapidMiner Classification Results

Model	Accuracy	Standard Deviation	Gains	Total Time	Training Time (1,000 Rows)	Scoring Time (1,000 Rows)
Naive Bayes	0.7	0.0	302.0	412379.0	29.2	133.9
Generalized Linear Model	0.7	0.0	296.0	1273698.0	90.4	189.7
Logistic Regression	0.8	0.0	378.0	455916.0	58.4	207.9
Fast Large Margin	0.6	0.0	186.0	715261.0	79.5	132.9
Deep Learning	0.8	0.0	424.0	518648.0	652.9	303.2
Decision Tree	0.7	0.0	250.0	471826.0	101.0	88.2
Random Forest	0.7	0.0	240.0	690948.0	444.4	2983.8
Gradient Boosted Trees	0.8	0.0	472.0	729231.0	1034.5	348.9
Support Vector Machine	0.6	0.0	196.0	2783858.0	674.8	497.0

Model	Classification Error	Standard Deviation	Gains	Total Time	Training Time (1,000 Rows)	Scoring Time (1,000 Rows)
Naive Bayes	0.281225937	0.026978707	302	412379	29.19708029	133.8742394
Generalized Linear Model	0.284062817	0.015240045	296	1273698	90.4298459	189.6551724
Logistic Regression	0.225866261	0.029114785	378	455916	58.39416058	207.9107505
Fast Large Margin	0.36218845	0.026067303	186	715261	79.48094079	132.8600406
Deep Learning	0.193161094	0.025581069	424	518648	652.8791565	303.2454361
Decision Tree	0.322431611	0.030995225	250	471826	100.973236	88.23529412
Random Forest	0.327659574	0.017659432	240	690948	444.4444444	2983.772819
Gradient Boosted Trees	0.160506586	0.019753773	472	729231	1034.468775	348.8843813
Support Vector Machine	0.355126646	0.020929418	196	2783858	674.7769667	496.9574037

Model	F Measure	Standard Deviation	Gains	Total Time	Training Time (1,000 Rows)	Scoring Time (1,000 Rows)
Naive Bayes	0.736509175	0.026196975	302	412379	29.19708029	133.8742394
Generalized Linear Model	0.738870698	0.013976066	296	1273698	90.4298459	189.6551724
Logistic Regression	0.771881082	0.033617564	378	455916	58.39416058	207.9107505
Fast Large Margin	0.65400144	0.035793991	186	715261	79.48094079	132.8600406
Deep Learning	0.801156361	0.03454272	424	518648	652.8791565	303.2454361
Decision Tree	0.62502264	0.045963222	250	471826	100.973236	88.23529412
Random Forest	0.561055083	0.0304325	240	690948	444.4444444	2983.772819
Gradient Boosted Trees	0.835646653	0.022754714	472	729231	1034.468775	348.8843813
Support Vector Machine	0.626364001	0.037441117	196	2783858	674.7769667	496.9574037

Model	Precision	Standard Deviation	Gains	Total Time	Training Time (1,000 Rows)	Scoring Time (1,000 Rows)
Naive Bayes	0.688392331	0.036996286	302	412379	29.19708029	133.8742394
Generalized Linear Model	0.690902966	0.023951111	296	1273698	90.4298459	189.6551724
Logistic Regression	0.787229151	0.024523261	378	455916	58.39416058	207.9107505
Fast Large Margin	0.631374009	0.013774297	186	715261	79.48094079	132.8600406
Deep Learning	0.832024174	0.011483269	424	518648	652.8791565	303.2454361
Decision Tree	0.744178896	0.03426579	250	471826	100.973236	88.23529412
Random Forest	0.855834685	0.023184406	240	690948	444.4444444	2983.772819
Gradient Boosted Trees	0.84891435	0.040431188	472	729231	1034.468775	348.8843813
Support Vector Machine	0.669049047	0.018134171	196	2783858	674.7769667	496.9574037

Model	Recall	Standard Deviation	Gains	Total Time	Training Time (1,000 Rows)	Scoring Time (1,000 Rows)
Naive Bayes	0.793495617	0.035039937	302	412379	29.19708029	133.8742394
Generalized Linear Model	0.79524453	0.029552031	296	1273698	90.4298459	189.6551724
Logistic Regression	0.758414843	0.053540031	378	455916	58.39416058	207.9107505
Fast Large Margin	0.680272415	0.065165017	186	715261	79.48094079	132.8600406
Deep Learning	0.775628486	0.071355218	424	518648	652.8791565	303.2454361
Decision Tree	0.539603175	0.052588274	250	471826	100.973236	88.23529412
Random Forest	0.417915058	0.031372409	240	690948	444.4444444	2983.772819
Gradient Boosted Trees	0.825832741	0.051111474	472	729231	1034.468775	348.8843813
Support Vector Machine	0.592719863	0.068610893	196	2783858	674.7769667	496.9574037

Key Insights:

- Gradient Boosting performed best, mirroring Python results.
- Auto Model highlighted similar feature importances, confirming the reliability of Python-based preprocessing and modeling.

- RapidMiner streamlined the process, making it user-friendly for non-programmers.

6. Deployment

Best Model Selection

Based on accuracy and interpretability, **Gradient Boosting** was chosen as the best model. It will be deployed to predict attrition and guide decision-making.

Real-World Applications

1. **Retention Strategies:**
    - HR can focus on employees working overtime and those with lower job satisfaction.
    - Salary adjustments and career development programs can target at-risk employees.
2. **Automation:** Automating the attrition prediction process through integration with HR systems.
3. **Continuous Monitoring:** Periodic retraining of the model with updated data to improve predictions.

Model Deployment in Python

The best-performing Deep learning neural network model was saved as a serialized file (best_model.pkl) using Python's pickle library for future predictions. The deployment process includes:

- Loading the model.
- Accepting new employee data as input.
- Predicting attrition probability.
- Generating reports for HR decision-makers.

Learnings:

- Deepened my understanding of ensemble techniques and their role in improving classification accuracy.
- Learned the advantages of using tools like RapidMiner to accelerate model experimentation and reporting.

Conclusion

The study included in this report aims at discovering more about the employee attrition issue using IBM HR Analytics dataset and data mining approaches, methods, and models. By following the CRISP-DM, that focuses on the efficiency of data mining tasks, an idea on key factors that cause withdrawal was derived and several models tested for their ability to make predictions. From among the machine learning techniques used, Neural Networks and Gradient Boosting were found to achieve better accuracy and reliability, and therefore could be recommended for use in real world problems.

In addition, when using RapidMiner Auto Model for benchmarking the performances confirmed the results obtained in Python for the validation of the predictive models and thus demonstrated the potential of the automated machine learning interfaces. The break down of the decision making process with RapidMiner's explainability options and the assessment of different models was made easy by the Simple User Interface of the software and also helped quite a lot in the comparison between models with additional insights of the dataset.

The analysis of Python together with RapidMiner helps to draw attention to the ability to conduct data mining both manually and with the help of the program, using its points only as a supplement. This study highlights the importance of employee attrition from the analytical perspective so that organizations can follow suitable solutions for retention of the employees.

For example, future work may investigate more sophisticated form of ensemble method, include time series data or examine other datasets from organizations for improving performance of the model. To bring focus on the issue of attrition, through the use data mining, various technique organizations can effectively prevent the occurrence of attrition which in turn create positive impact toward the formation of sustainable workforce that will enhance the total efficiency of organization.

References

IBM HR Analytics Employee Attrition & Performance. (2024). *Kaggle*. Retrieved from https://www.kaggle.com/datasets/pavansubhasht/ibm-hr-analytics-attrition-dataset

RapidMiner. (2024). *RapidMiner Documentation*. Retrieved from https://docs.rapidminer.com/

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825-2830.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, *29*(5), 1189-1232. https://doi.org/10.1214/aos/1013203451

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5-32. https://doi.org/10.1023/A:1010933404324

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. New York: Springer.

McKinney, W. (2010). Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference*. https://doi.org/10.25080/majora-92bf1922-00a

Van Rossum, G., & Drake, F. L. (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.

Shapley, L. S. (1953). A value for n-person games. *Contributions to the Theory of Games*, *2*(28), 307–317.

Chollet, F. (2017). Deep Learning with Python. *Manning Publications Co.*.

HR Analytics. (2024). Employee Retention Strategies Using HR Analytics. *HRZone*. Retrieved from https://www.hrzone.com/

RapidMiner Auto Model. (2024). Simplified Machine Learning Model Building. *RapidMiner Academy*. Retrieved from https://academy.rapidminer.com/