big data and cognitive computing

Article

Automating Feature Extraction from Entity-Relation Models: Experimental Evaluation of Machine Learning Methods for Relational Learning

Boris Stanoev [1,2,*], **Goran Mitrov** [1,2], **Andrea Kulakov** [1], **Georgina Mirceva** [1], **Petre Lameski** [1,2] and **Eftim Zdravevski** [1,2,*]

[1] Faculty of Computer Science and Engineering, Ss Cyril and Methodius University, 1000 Skopje, North Macedonia; goran.mitrov@finki.ukim.mk (G.M.); andrea.kulakov@finki.ukim.mk (A.K.); georgina.mirceva@finki.ukim.mk (G.M.); petre.lameski@finki.ukim.mk (P.L.)

[2] Magix.AI, 1000 Skopje, North Macedonia

* Correspondence: boris.stanoev@finki.ukim.mk (B.S.); eftim.zdravevski@finki.ukim.mk (E.Z.)

Abstract: With the exponential growth of data, extracting actionable insights becomes resource-intensive. In many organizations, normalized relational databases store a significant portion of this data, where tables are interconnected through some relations. This paper explores relational learning, which involves joining and merging database tables, often normalized in the third normal form. The subsequent processing includes extracting features and utilizing them in machine learning (ML) models. In this paper, we experiment with the propositionalization algorithm (i.e., Wordification) for feature engineering. Next, we compare the algorithms PropDRM and PropStar, which are designed explicitly for multi-relational data mining, to traditional machine learning algorithms. Based on the performed experiments, we concluded that Gradient Boost, compared to PropDRM, achieves similar performance (F1 score, accuracy, and AUC) on multiple datasets. PropStar consistently underperformed on some datasets while being comparable to the other algorithms on others. In summary, the propositionalization algorithm for feature extraction makes it feasible to apply traditional ML algorithms for relational learning directly. In contrast, approaches tailored specifically for relational learning still face challenges in scalability, interpretability, and efficiency. These findings have a practical impact that can help speed up the adoption of machine learning in business contexts where data is stored in relational format without requiring domain-specific feature extraction.

Keywords: data mining; relational learning; propositionalization; machine learning; deep learning

check for updates

Citation: Stanoev, B.; Mitrov, G.; Kulakov, A; Mirceva, G.; Lameski, P.; Zdravevski, E. Automating Feature Extraction from Entity-Relation Models: Experimental Evaluation of Machine Learning Methods for Relational Learning. *Big Data Cogn. Comput.* **2024**, *8*, 39. https://doi.org/10.3390/bdcc8040039

Academic Editor: Domenico Talia

Received: 27 February 2024
Revised: 21 March 2024
Accepted: 28 March 2024
Published: 1 April 2024

1. Introduction

With the enormous expansion of data, the necessity for big data architectures for the efficient, reliable, and prompt processing of this data has become more pronounced [1]. In turn, this entails efficient algorithms for optimizing cluster size and cost [2], and for scalable feature selection and dimensionality reduction [3].

Moreover, the whole knowledge discovery process in databases, as shown in Figure 1 [4,5], requires multiple steps, which are even more complex and time-consuming in big data applications. Preparing relational data (i.e., data that can be represented in an Entity Relation—ER model) in a format suitable for machine learning is a manual effort. A large part of the data today is stored in so-called relational databases, which are stored in several related tables with logical or physical foreign keys. Processing this kind of data requires a good understanding of the data domain, and human experience plays a significant role. To be able to train machine learning (ML) algorithms on data represented in normalized ER models, it is necessary to join and merge many tables, perform manual feature engineering and extraction, and ultimately represent the data in one wide table.

For that reason, in this paper, we focus on the topic of relational learning [6–8], which involves merging relational database tables into one table, extracting features, and later using those features to train machine learning models. Aiming to automate this process of extracting knowledge from relational databases, we perform various experiments with different feature extraction algorithms in relational learning and machine learning models.

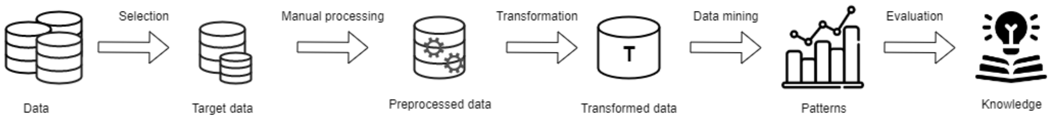

Figure 1. Knowledge discovery in databases consisted of acquiring data, understanding the data, processing, transformation, data mining, and evaluation.

As shown in Figure 2, the first step of the proposed methodology is to transform relational databases into a single table. To do that, we use propositionalization [9] techniques that generate complex features. Specifically, the wordification [10] technique is used as a type of propositionalization algorithm. As presented in Figure 2, wordification is the first step in the automated relational learning process, which, as a result, generates an extensive feature set, denoted as 'propositionalized relational database' in the figure.

In the next step, using this very wide dataset with many features, we evaluate various machine learning algorithms. We evaluate traditional ML algorithms (ones not explicitly adapted for these types of tasks) and propositional learners (ML algorithms designed to be trained on the propositional representation of the source data). Specifically, from the propositional learners, we evaluate PropDRM [11] and PropStar [11]. PropDRM represents a deep relational machine tailored for propositional learning, while PropStar is a feature-based technique specifically developed for classification tasks in relational databases, utilizing advanced feature embeddings. This comparison of propositional learners and traditional ML algorithms aims to highlight the strengths and limitations of each set of methods in various learning scenarios and data types.

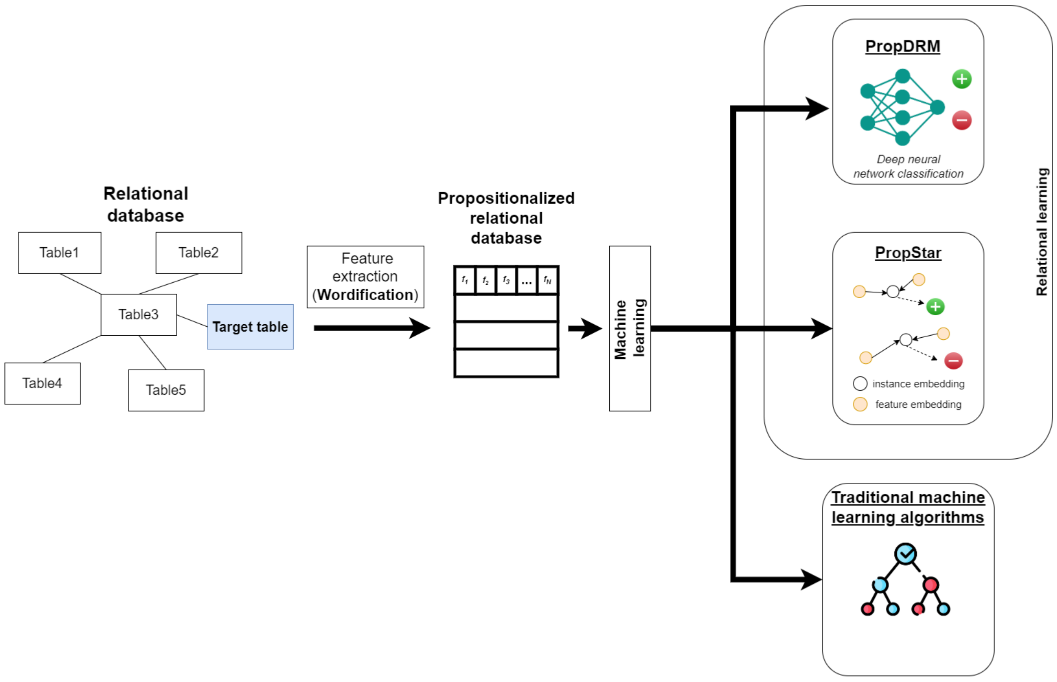

Figure 2. Automated relational learning process.

The main contributions of this work are the following:

- We reproduce the original paper's works studying the wordification, PropDRM, and PropStar algorithms.

- We examine a comparative analysis between the traditional algorithms versus PropDRM and PropStar, based on the performance over various metrics.
- We demonstrate that traditional algorithms achieve similar results compared to the PropDRM and PropStar in the common benchmark datasets.

In Section 2, we mention some existing studies related to relational learning, propositionalization approaches, and datasets used as a baseline in this topic. Then, in Section 3, we delve into the Wordification technique and the combination of this algorithm with various machine learning models. After that, in Section 4 we describe the datasets examined in this paper. Next, in Sections 5 and 6, we present and discuss the results from our experiments, respectively. Finally, in Section 7, we conclude the paper and present some ideas for future work.

2. Related Work

This section reviews the different methods and techniques analyzed in this paper. Namely, in different organizations and enterprises, the majority of the data is stored in relational databases consisting of dozens, hundreds, or even thousands of tables often normalized in the third normal form [12]. Therefore, there is a need to first represent these databases in a denormalized tabular format that ML algorithms can utilize for learning. In the following subsections, we consider different feature extraction algorithms and learning tasks in the pipeline represented in Figure 1.

We first performed a literature analysis to understand the general landscape of methods in the field of relational learning to analyze potentially relevant articles to the following topics:

- Machine learning on relational database;
- Feature engineering for relational database;
- Relational learning;
- Learning from structured data.

Figure 3 presents a visual representation of the annual occurrences of the relevant articles categorized based on the search phrases. An observation that emerges from the figure is the growing trend of articles published between January 2012 and February 2024. This indicates a notable surge in interest surrounding the subject of relational learning, thereby worthy of further exploration and investigation. From the potentially relevant articles, we considered 23 articles with methods related to the main topic of the articles. From those 23, we selected the most promising ones and reproduced the experiments published in the original article while performing additional tests. The following subsections summarize these methods.

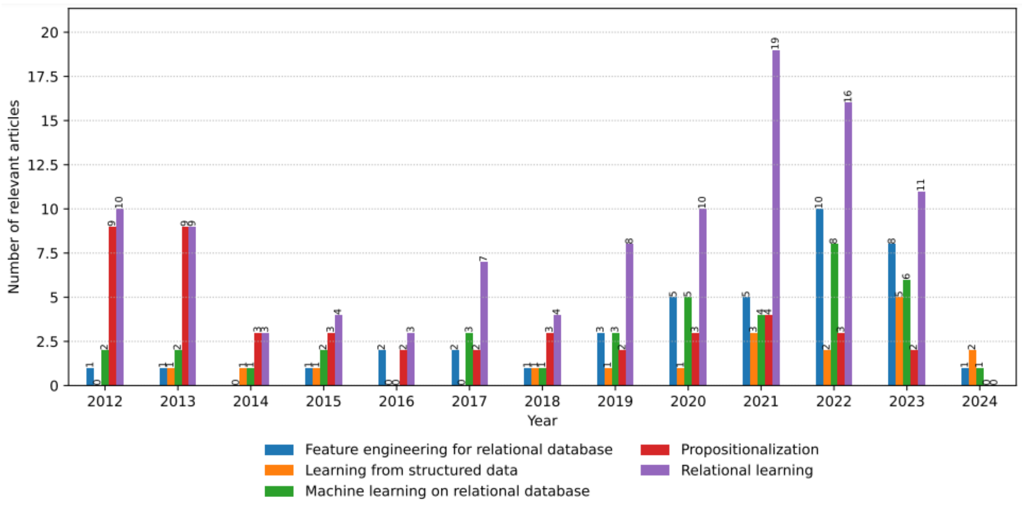

Figure 3. The number of relevant articles per year from January 2012 to February 2024, grouped by the search phrase.

2.1. Methods for Feature Extraction from Relational Databases

LINUS [13], a pioneering propositionalization approach, was used for automated relational feature construction. However, it had limitations, such as not allowing recursion, existential local variables, and join queries. To address these limitations, SINUS [14], a descendant of LINUS, incorporated more advanced feature construction techniques. Relaggs [15] represents a method called relational aggregation. This technique, part of propositionalization, uses the structure of a relational database as a guide for its process. It incorporates optimization methods commonly found in relational databases, like indexing, to aggregate and summarize data from non-target relations about specific entries in the target table.

RollUp [16] is a basic propositionalization method that performs a depth-first search through a data model, summarizing data from various tables onto a target table by "rolling up" information at each level of the search until all data is aggregated at the target.

The Data Science Machine introduces Deep Feature Synthesis [17], akin to RollUp, for automatic feature generation from relational or multi-table datasets using simple aggregations. It also employs an unsupervised feature selection method using Singular Value Decomposition (SVD) and ranks features by their relevance to the target class. This method is part of a broader effort to automate the entire data science process, with relational feature engineering being a key aspect.

The DARA [18] algorithm focuses on condensing information from records in a non-target table, which is linked in a many-to-one fashion to records in a target table. It summarizes these records and then integrates this summarized information into the target table.

Previously mentioned methods use simple aggregation functions sum, min, max, mean, and count to summarise information from one to many related tables and have shown to be effective on multi-table classification problems. A limiting problem with those methods is that runtime and memory performance degrades when there are multiple one-to-many relationships. These relationships necessitate the recursive creation of aggregate features, leading to a rapid increase in the number of features. This increase, known as a combinatorial explosion, caused by the sequential one-to-many joins or 'depth', results in a large number of redundant features. This redundancy can adversely affect both accuracy and runtime.

Wordification [10] is a propositionalization method that transforms a relational database into a corpus of text documents. It is efficient for large datasets and allows for the use of text-mining techniques on the transformed data.

2.2. Approaches for Propositional Learning

Deep Relational Machine (DRM) [19], a deep learning neural network that excels in capturing structural and relational information in data. The DRM learns representations using first-order Horn clauses in the initial layer and restricted Boltzmann machines in successive layers. However, deep relational machines were used on propositionalized data for the first time by Srinivasan [20].

Similarly, the PropStar algorithm [11], based on the StarSpace model [21], is used for transforming relational databases into propositionalization features. StarSpace is a versatile neural embedding model capable of various tasks, including text classification and multi-relational graph embedding. PropStar leverages StarSpace to create embedding vectors that can be easily compared in latent space.

2.3. Graph Learning Methods

Knowledge Embedding with Numbers (KEN) [22] is a novel method for feature extraction from relational databases by leveraging graph embedding techniques, to effectively encode numerical attributes. This advancement facilitates the enrichment of machine learning models with comprehensive background information, demonstrating notable improvements in predictive tasks across diverse datasets.

Lifted Relational Neural Networks (LRNN) [23,24] is a framework that dynamically generates differentiable computation graphs from relational data. This approach allows for a unified representation of various neural models, facilitating the extension of graph neural networks (GNNs) to achieve greater expressiveness and efficiency in relational learning tasks.

Relational Deep Learning (RDL) [25], represents a novel approach that allows the execution of machine learning tasks directly on relational databases, eliminating the need for manual feature engineering. By conceptualizing databases as temporal, heterogeneous graphs, RDL enables Graph Neural Networks to effectively learn from the intricate relationships and dynamic changes inherent in relational data.

In a recent study [26], a deep learning framework is introduced for direct end-to-end learning from relational databases through a neural message-passing architecture that aligns with the database's relational structure. This framework is shown to provide competitive performance against existing models, marking a significant step towards the application of deep learning techniques on relational data.

While the methods described previously represent significant advancements in leveraging graph-based approaches for relational data, our study opts for a different direction. The primary reason for this deviation lies in our focus on maintaining the inherent structure of relational databases rather than transforming or interpreting this data through the lens of graph or knowledge graph methodologies. Our decision stems from a desire to work within the original relational database framework, as it allows for direct interaction with the data in its native form, preserving the original semantics and relationships as defined by the database schema.

3. Methods

Based on the review of the related work described in the previous section, in this section, we describe the most promising approaches: Wordification [10] for feature extraction, and the unified approaches PropDRM [11], and PropStar [11], followed by the traditional algorithms used for learning over the propositionalized data. Therefore, the remainder of the paper focuses on a detailed comparison of these approaches and achieved metrics.

3.1. Feature Extraction

Wordification

In this subsection, we describe in detail the Wordification method, where we demonstrate how the algorithm works over the East–West Trains dataset [27].

Wordification [10] is a process that involves transforming a relational database into a collection of feature vectors. Each original instance is converted into a "document" represented as a Bag-Of-Words (BOW) vector, where simple features are assigned weights resembling "words" in the transformed BOW space. These "words" correspond to attribute-value pairs from the target and related tables, weighted using Term Frequency-Inverse Document Frequency (TF-IDF) or simpler schemes like term frequency (TF) or binary presence/absence indicators. While wordification may result in some loss of information compared to propositionalization methods, it offers advantages such as interpretability, scalability for large databases, parallel processing, and using text mining techniques like document clustering and word cloud visualization for multi-relational data mining. Wordification is classified as the first step in the automated relational learning process, presented in Figure 2.

This method uses the relational database, represented as an ER model, as an input into the wordification algorithm. To represent a table as a BOW, we treat each entry in the table as a text document. We create synthetic words called *witems* by combining the table name, column name, and its discrete value (using Equation (1)). Continuous values are discretized before this process. Each entry's document is formed by generating *witems* for the table and related rows based on the relational database schema. These *witems* are then joined together to create the entry's document.

$$witem = table\_name\_attribute\_name\_value \tag{1}$$

Once we have documents for all entries, we obtain a corpus of text documents. Each *witem* in the corpus is considered an attribute to create the BOW representation. The value associated with each *witem* for a given entry is determined by its *TF-IDF* (term frequency-inverse document frequency) weight, where *TF-IDFf*(w_i, *d*) calculates the importance of the *witem* in the document associated with the entry. For a given *witem*, w, in document d from corpus D, the measure *TF-IDF* $(\cdot)$ is defined as follows:

$$tfidf(w,d) = tf(w,d) \times \log \frac{|D|}{|\{d \in D : w \in d|\}} \tag{2}$$

where *TF* $(\cdot)$ represents the TF, that is the number of times that *witem* w appears in document d, and *IDF* is the logarithm of the inverse of the number of documents the *witem* w appears in. Additionally, this algorithm has two more weight implementations, TF, and binary weights.

Because of simplicity, we present a sample of the Trains dataset [27] to demonstrate how the algorithm transforms the relational database into BOW representation (Tables 1 and 2). In Figure 4, we can see the Entity Relationship (ER) diagram from the Trains dataset, which contains two tables: *trains*, which is the target table; and *cars*.

Table 1. Trains sample data.

Trains	
Id	**Direction**
1	East
2	West

Table 2. Cars sample data.

Cars		
Id	**Train_Id**	**LoadShape**
1	1	Triangle
2	1	Circle
3	2	Circle
4	2	Diamond

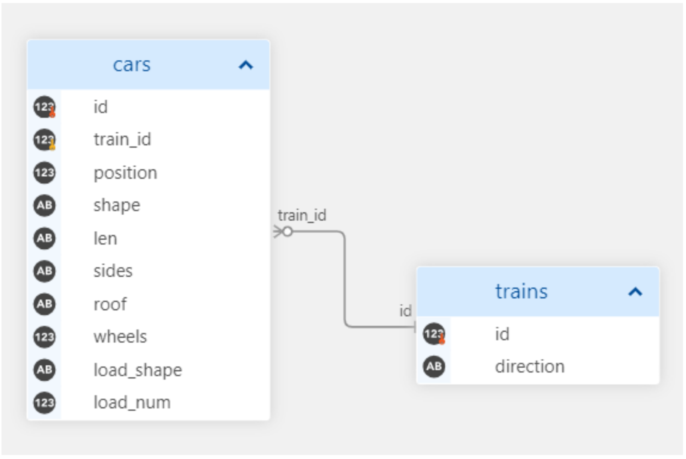

Figure 4. Entity relationship diagram from Trains dataset.

From our sample data, the documents that are generated from wordification are displayed in Table 3 corresponding generated documents. After this, the documents are transformed into the BOW representation by calculating the TF-IDF values for each word of each document (using Equation (2)) with the class attribute column appended to the transformed BOW table, as shown in Table 4.

Table 3. Documents generated by Wordification algorithm for each train.

Train 1:	[Cars_LoadShape_Triangle, Cars_LoadShape_Circle], East
Train 2:	[Cars_LoadShape_Circle, Cars_LoadShape_Diamond], West

Table 4. Calculated TF-IDF values from the sample data.

Train_Id	Cars_LoadShape_Triangle	Cars_LoadShape_Circle	Cars_LoadShape_Diamond	Direction
1	1.386	0	0	East
1	0	0.693	0	East
2	0	0.693	0	West
2	0	0	1.386	West

3.2. Relational Learning

3.2.1. PropDRM

PropDRM [11] utilizes a modified version of the DRM approach. It can directly learn from large, sparse matrices produced by the wordification algorithm for propositionalizing relational databases. In wordification, each instance is represented by a bag of features in the format TableName_AttributeName_Value, treated as words in the transformed BOW representation. In PropDRM, these words represent individual relational items using the (table.name, column.name, value) notation.

PropDRM generates relational representations for each instance. These batches of instances serve as input to a neural network for downstream tasks such as classification or regression, as shown in Figure 2. Although propositionalization and learning are conceptually separate, they can be integrated efficiently in practice. The advantage of PropDRM is its ability to handle sparse matrices generated by the wordification algorithm.

$$\omega = \sigma(W_1^T * (ELU(Drop(W_0^T * P + b_0))) + b_1) \tag{3}$$

PropDRM employs dense feed-forward neural networks (Equation (3)) with dropout regularization and ELU activation function on intermediary weights. The sigmoid function is used for output weights to obtain binary predictions. Additionally, W_1 and W_0 are weight matrices and the superscript T denotes the transpose of a matrix, b_1 and b_0 are bias vectors, and P is the input feature matrix. The training procedure utilizes binary cross-entropy loss, referred to as Loss, which is defined in Equation (4). The loss function returns the probability p_{ij} of an instance i belonging to a class j, where y_{ij} is a binary value (0 or 1) indicating whether class j is the correct class label assigned to instance i, and C is a set of all the target classes. Each output neuron, for the total number of classes $|C|$, produces a single probability p_{ij} for the associated class j within the complete set of classes C. During the training of the neural networks using small batches, the results of the loss function are averaged to compute the overall loss of a given batch of instances.

$$Loss^{CE}(i) = \sum_{j \in C} y_{ij} * \log p_{ij} \tag{4}$$

Overall, PropDRM combines Wordification, neural networks, and binary cross-entropy loss to learn from relational databases and make predictions effectively.

3.2.2. PropStar

In this subsection, we review the PropStar [11] approach that is based on the StarSpace algorithm [21]. PropStar is a feature-based approach for classification in relational databases using feature embeddings. This approach uses embedding vectors to represent the features of the dataset. In comparison with the PropDRM approach, this algorithm is different in the sense that representations are not learned for individual instances (as is the case of DRMs). Instead, they are learned for every single relational feature that is the output of the selected propositionalization algorithm, for example in our case that is Wordification.

The PropStar involves two steps: transforming the relational database into sets of features using the Wordification method and utilizing these sets of relational items as input for the StarSpace entity embedding algorithm. Embeddings are computed by efficient C++ implementation, and the resulting representations are learned for individual relational items. It is worth mentioning that the embeddings are computed for each distinct relational feature, including the label. In an intuitive sense, embedding construction can be interpreted as identifying the positions of relational items in latent space by considering their patterns of co-occurrence with other items across all training instances, as shown in Figure 2. Since the relational features and labels are present in the same latent space, predicting the label for a given relational bag can be done by direct comparison. Let M represent a novel instance to be classified, where M is represented as a multiset of relational items. StarSpace averages the representations of relational features, present in a given input instance (a bag). The representation is normalized and compared to label embeddings in the common space. The representation of a relational bag e_M (defined in Equation (5)) is calculated as

$$e_M = \frac{\bigoplus\limits_{f_i \in M} e_{f_i}}{\sqrt{|M_{unique}|}} \tag{5}$$

where e_{f_i} denotes the embedding of the i-th relational feature within instance M, and e_M is the resulting d-dimensional, real-valued vector that represents the bag. The $\oplus$ operator indicates an element-wise summation over all unique relational feature embeddings considered in instance M. The set M_{unique} contains all the unique relational features present in M. The computed vector e_M is then compared to label embeddings in the same space (displayed in Equation (6)). Therefore, e_c represents the embedding of class label c, and the similarity between e_M and each e_c is calculated using the cosine similarity function $sim(e_1, e_2)$, a measure that is very efficient when it comes to comparing high-dimensional data. The label whose embedding is most similar to e_M, as determined by this similarity metric, is selected as the label for the instance.

$$label(e_M) = \arg\max_{c \in C}[sim(e_M, e_c)] \tag{6}$$

The spatial complexity for this algorithm is linear. The complexity of predicting a single relational bag is O($|C|$), where C is the number of unique labels.

Overall, PropStar represents features as embedding vectors, allowing for the direct classification of new instances based on the proximity of their feature representations to class label embeddings in a latent space.

3.2.3. Traditional Machine Learning Algorithms

In addition to those specialized methods for realtional learning mentioned earlier in this subsection, we also consider traditional machine learning algorithms. Specifically, we consider the following algorithms because of their versatility and proven performance of diverse machine learning tasks. What is common about them is that they rely on a training dataset presented in a tabular format where one of the columns is the target label.

While numeric data is commonly used in machine learning, many real-world issues involve dealing with discrete forms of data such as graphs, relationships, texts, or electronic health records. To apply numeric-based deep learning methods to these types of data, it is imperative to transform the nominal data into a numeric format that is compatible with these learning algorithms. Numerical representations are beneficial even in discrete domains, as they allow symbolic learners to make generalizations based on the similarity of objects.

To utilize the full potential of contemporary machine learning algorithms like AdaBoost, ExtraTrees, GradientBoost, and RandomForest, alongside deep neural networks, it is essential to convert discrete data into numeric vectors. These vectors should effectively preserve and convey object similarities as distances within the numeric space. The propositional learners are the second step in the automated relational learning process, presented in Figure 2.

AdaBoost [28], short for Adaptive Boosting, is a powerful ensemble technique. AdaBoost combines multiple weak classifiers to form a strong classifier by adjusting the weights of incorrectly classified instances so that subsequent classifiers focus more on them.

RandomForest [29] is a popular ensemble learning method that operates by constructing a set of decision trees during training. For classification tasks, it outputs the class which is the mode of the classes of the individual trees.

ExtraTrees [30], or Extremely Randomized Trees, is an ensemble learning method that is similar to the RandomForest algorithm. It differs by the way it splits nodes, using random thresholds for each feature rather than searching for the most optimal thresholds.

GradientBoost [31,32] is an ensemble method that constructs the model progressively, stage by stage. It is a form of boosting that optimizes a loss function, and each new model incrementally reduces the errors made by the previous models.

4. Datasets

In this section, we describe the datasets, displayed in Table 5, that were used for evaluating the performance of the algorithms:

- The Trains dataset [27] is commonly used in the well-known ILP East–West trains challenge problem. This challenge involves predicting a train's direction (eastbound or westbound) based on the cars' characteristics in each direction. The trains can have various cars, each with different shapes and carrying different loads.
- The Carcinogenesis task [33] focuses on predicting whether a diverse range of chemical compounds is carcinogenic. The dataset used for this task involved conducting experiments on rodents, which spanned several years and involved hundreds of animals. The dataset comprises 329 compounds, with 182 of them identified as carcinogens.
- The Mutagenesis task [34] focuses on predicting the mutagenicity of aromatic and heteroaromatic nitro compounds. This task is important as it is closely related to predicting carcinogenesis. The dataset consists of 230 compounds, with 138 labeled as 'active' indicating positive mutagenicity and the remaining compounds labeled as 'inactive' serving as negative examples. Data present in the original paper were split into two subsets: 188 compound dataset and a smaller dataset with 42 compounds.
- IMDB database http://www.webstepbook.com/supplements/databases/imdb.sql (accessed on 27 February 2024) contains tables of movies, actors, movie genres, directors, and director genres. The dataset used in our experiments encompasses only movies whose titles and years of production appear in the IMDB's top-250 and bottom-100 charts (Snapshot taken on 2 July 2012). The snapshot contains 166 movies and all their actors, genres, and directors. Movies present in the IMDB top-250 chart were labeled as positive examples, and those in the bottom-100 as negatives.
- The MovieLens dataset https://relational-data.org/dataset/MovieLens (accessed on 27 February 2024), available in the UC Irvine machine learning repository, is similar to the IMDB dataset mentioned earlier but significantly larger. It contains over 1.2 million instances in total. This task aims to predict the gender of users in the movie database.

Table 5. Original datasets summary information.

Database	Table	No. Rows	No. Attributes
Trains	Trains	20	2
	Cars	63	10
Carcinogenesis	atom	9064	5
	canc	329	2
	sbond_1	13,562	4
	sbond_2	926	4
	sbond_3	12	4
	sbond_7	4134	4
Mutagenesis 42	atoms	1001	5
	bonds	1066	5
	drugs	42	7
	ring_atom	1785	3
	ring_strucs	279	3
	rings	259	2
Mutagenesis 188	atoms	4893	5
	bonds	5243	5
	drugs	188	7
	ring_atom	9330	3
	ring_strucs	1433	3
	rings	1317	2
IMDB	actors	7118	4
	directors	130	3
	directors_genres	1123	4
	movies	166	4
	movies_directors	180	3
	movies_genres	408	3
	roles	7738	4
MovieLens	actors	99,129	3
	directors	2201	3
	movies	3832	5
	movies2actor	152,532	3
	movies2directors	4141	3
	u2base	946,828	3
	users	6039	4

5. Results

5.1. Classification Performance Evaluation

This section presents the results from the examined methodologies over the proposed datasets. In our comprehensive experimental setup, we tried to reproduce the experiments from the original paper [11] and embarked on a systematic examination of a suite of machine learning algorithms across multiple datasets to assess their performance and generalizability. The algorithms under scrutiny included traditional models like AdaBoost, ExtraTrees, RandomForest, and GradientBoost, as well as two propositionalization algorithms, PropDRM and PropStar. To ensure a thorough evaluation, each algorithm was subjected to an extensive range of hyper-parameters, thereby facilitating a robust exploration of their respective predictive capabilities. We used a 10-fold cross-validation approach consistently across all datasets to ensure our evaluation was reliable and uniform.

This cross-validation not only enhanced the validity of our results by reducing the potential for over-fitting but also allowed us to extrapolate our findings with greater confidence.

In evaluating the comparative effectiveness of traditional machine learning algorithms against proprietary models across multiple datasets, our analysis reveals noteworthy trends, which are represented in Table 6, where the highlighted cells show the best performing algorithm per metric and dataset. A more visual representation of the same results is presented in Figure 5. The traditional algorithms AdaBoost, ExtraTrees, RandomForest, and GradientBoost demonstrated consistent performance, with GradientBoost frequently achieving superior accuracy and F1 scores. This trend was particularly evident within the 'Carcinogenesis' and 'IMDB' datasets.

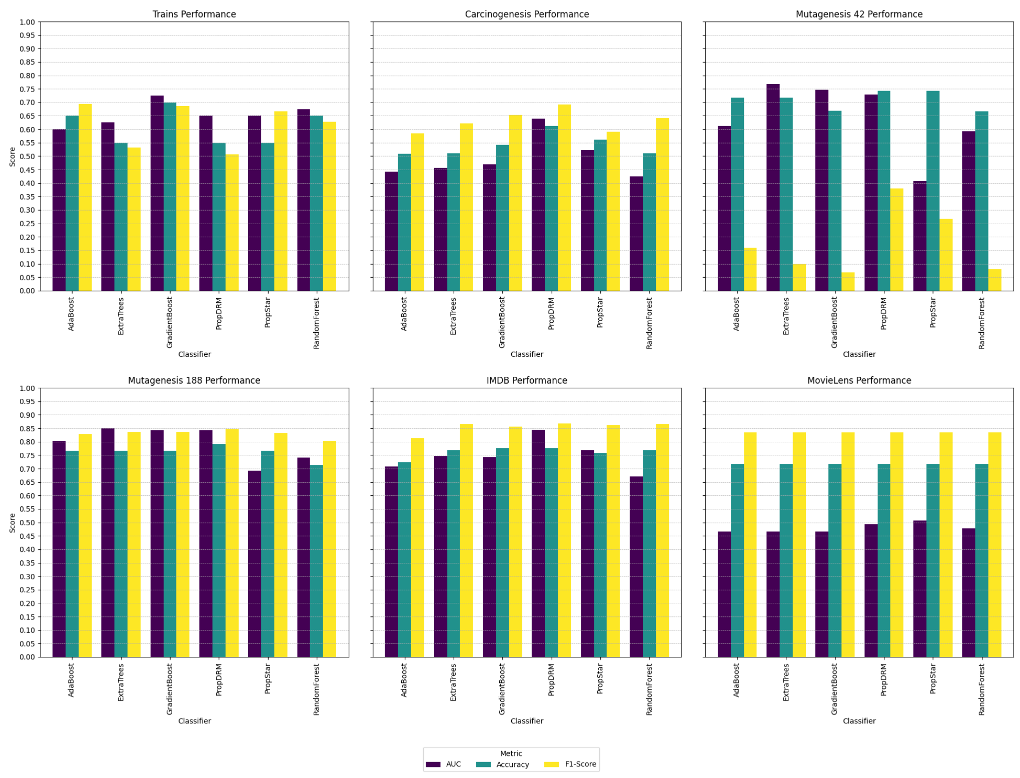

Figure 5. Wordification performance of different datasets and algorithms.

Table 6. Wordification performance of different datasets and algorithms.

Dataset	Classifier	Accuracy	F1-Score	AUC
Trains	AdaBoost	0.650	**0.693**	0.600
	ExtraTrees	0.550	0.533	0.625
	GradientBoost	**0.700**	0.687	**0.725**
	RandomForest	0.650	0.627	0.675
	PropDRM	0.550	0.507	0.650
	PropStar	0.550	0.667	0.650
Carcinogenesis	AdaBoost	0.508	0.584	0.442
	ExtraTrees	0.511	0.622	0.456
	GradientBoost	0.541	0.652	0.469
	RandomForest	0.511	0.642	0.424
	PropDRM	**0.611**	**0.692**	**0.640**
	PropStar	0.562	0.591	0.523
Mutagenesis 42	AdaBoost	0.717	0.160	0.611
	ExtraTrees	0.717	0.100	**0.767**
	GradientBoost	0.669	0.067	0.746
	RandomForest	0.667	0.080	0.593
	PropDRM	**0.742**	**0.380**	0.729
	PropStar	**0.742**	0.267	0.407

Table 6. *Cont.*

Dataset	Classifier	Accuracy	F1-Score	AUC
Mutagenesis 188	AdaBoost	0.766	0.828	0.804
	ExtraTrees	0.767	0.836	**0.851**
	GradientBoost	0.766	0.837	0.843
	RandomForest	0.713	0.804	0.741
	PropDRM	**0.792**	**0.846**	0.843
	PropStar	0.766	0.833	0.692
IMDB	AdaBoost	0.724	0.814	0.707
	ExtraTrees	0.768	0.866	0.747
	GradientBoost	**0.777**	0.856	0.742
	RandomForest	0.768	0.866	0.670
	PropDRM	0.776	**0.868**	**0.845**
	PropStar	0.759	0.862	0.768
MovieLens	AdaBoost	**0.717**	**0.835**	0.466
	ExtraTrees	**0.717**	**0.835**	0.466
	GradientBoost	**0.717**	**0.835**	0.466
	RandomForest	**0.717**	**0.835**	0.478
	PropDRM	**0.717**	**0.835**	0.494
	PropStar	**0.717**	**0.835**	**0.507**

Notes: The best score for each dataset and metric is in **bold**.

5.2. Statistical Evaluation

To assess whether some classifiers are consistently better than others across the datasets for each metric separately, we used statistical tests. Given that we are comparing multiple groups (classifiers) across multiple independent samples (datasets), the most suitable approach is to use the Analysis of Variance (ANOVA) test for each metric. Before applying ANOVA, it is important to note that it assumes the following:

- The residuals are normally distributed (or approximately normally distributed).
- Homogeneity of variances (equal variances across groups).
- Independent observations.

The results of the ANOVA analysis for each metric are as follows:

- **Accuracy**: F-statistic = 34.95, p-value < 0.00001.
- **F1-Score**: F-statistic = 99.86, p-value < 0.00001.
- **AUC**: F-statistic = 17.20, p-value < 0.00001.

All metrics show highly significant p-values in their respective ANOVA tests, indicating that there are statistically significant differences in performance among the classifiers for each metric. Since the p-values are much lower than the typical alpha level of 0.05, we reject the null hypothesis of equal means and conclude that not all classifiers perform equally across the datasets for each metric.

6. Discussion

Conversely, the proprietary algorithms, PropDRM and PropStar, showed a divergent pattern of results. PropDRM exhibited exceptional F1-scores, most notably in the 'Mutagenesis 188' dataset, suggesting a robustness in performance that occasionally surpassed the traditional models. PropStar, however, did not consistently manifest a competitive advantage in any of the measured metrics. It is worth mentioning that the 'MovieLens' dataset demonstrated clear signs of overfitting despite various attempts to rectify it; therefore, its performance data were not considered in our conclusions to maintain the accuracy of our overall analysis.

Compared to the initial study [11] that focused solely on accuracy, this research incorporates additional metrics such as the F1 score and ROC-AUC to provide a more comprehensive evaluation of the performance of the proposed methodologies, PropDRM and PropStar. This is because relying solely on accuracy can be deceptive, especially in imbalanced datasets where it can disproportionately reflect the majority class's influence, failing to account for the model's ability to correctly predict the minority class. In contrast to the initial study, which employed a range of propositionalization techniques (Aleph, RSD, RelF, and Wordification) and machine learning algorithms (J48 and Support Vector Machine), our effort to replicate the original paper's experiments concentrates exclusively on the Wordification algorithm. This approach is paired with a variety of traditional machine learning algorithms (AdaBoost, RandomForest, ExtraTrees, Gradient Boost) and propositional learners (PropDRM and PropStar). Despite our efforts to reproduce the initial study's results, our findings generally fell short of the original benchmarks regarding the Accuracy metric.

Given the comprehensive statistical analysis across three key performance metrics (Accuracy, F1-Score, and AUC) and considering the findings from the statistical tests, no single classifier consistently outperforms others across all metrics. The choice of the "best" classifier depends on the specific needs of the task at hand and the importance of each metric. However, we can make some general observations.

GradientBoost shows significant improvements in Accuracy compared to several other classifiers, indicating strong overall performance in correctly predicting outcomes. However, it has a notable disadvantage in F1-Score, suggesting it may not balance precision and recall as effectively as some other models, particularly in imbalanced datasets.

RandomForest exhibits strong performances across multiple metrics, with particularly significant improvements in Accuracy and AUC compared to several classifiers. This suggests it is not only good at making correct predictions but also effective in ranking predictions with confidence across diverse scenarios. Its performance in F1-Score is also competitive, making it a robust choice for various applications.

PropDRM stands out in F1-Score, showing it can effectively balance precision and recall, which is crucial in scenarios where class imbalance is a concern. Its performance in Accuracy and AUC is also competitive, though it is not always the top performer. PropDRM could be preferred in scenarios where the F1-Score is prioritized, indicating its strength in handling imbalanced data effectively.

7. Conclusions and Future Work

This paper, explored the current state of relational learning, a topic that involves learning models or patterns from relational data. We discussed terms that are specific to this topic such as propositionalization techniques which are applicable for transforming relational databases into a single table format. One of the techniques that are examined in this paper is Wordification, which transforms the relational database into BOW features. Furthermore, we proceed with propositional learners such as PropDRM and PropStar, along with traditional machine learning algorithms—approaches that are highly compatible and perform effectively when integrated with the Wordification method.

In this paper, we reproduced the implementation from [11] and experimented with the previously mentioned techniques and algorithms. As discussed in Section 5, we can conclude that the traditional machine learning algorithms hold their ground with reliable performance compared to the propositionalization algorithms PropDRM and PropStar. It is worth mentioning that PropDRM demonstrated exceptional performance as an overall result; however, the PropStar algorithm showed a divergent pattern of results that cannot be taken for granted. Finally, the results indicate that the performance of these specific methods is not prominent difference when measured against traditional algorithms.

The topic of relational learning is experiencing increasing momentum with promising advancements achieved thus far. However, there is an opportunity to enhance existing concepts and consider alternative approaches that offer fresh perspectives.

In future work, we plan to explore adjusting weights in the Wordification algorithm. We believe that focusing on features directly related to the predicted label may improve outcomes. Since the success of DRMs largely depends on feature quality, refining this could significantly enhance the PropDRM method. Additionally, our studies indicate that dataset selection is crucial to algorithm performance. Hence, we intend to experiment with a wider variety of datasets, including more complex and challenging ones, to better understand the effectiveness of different approaches in real-world data situations.

Author Contributions: Conceptualization, B.S. and E.Z.; Data curation, B.S. and E.Z.; Formal analysis, P.L. and E.Z.; Funding acquisition, E.Z.; Investigation, B.S.; Methodology, B.S. and E.Z.; Project administration, G.M. (Georgina Mirceva), P.L. and E.Z.; Resources, B.S., A.K., G.M. (Georgina Mirceva), P.L. and E.Z.; Software, B.S. and G.M. (Goran Mitrov); Supervision, P.L.; Validation, G.M. (Goran Mitrov), A.K., P.L. and E.Z.; Visualization, B.S. and G.M. (Goran Mitrov); Writing—original draft, B.S. and G.M. (Goran Mitrov); Writing—review and editing, B.S., G.M. (Goran Mitrov), A.K., G.M. (Georgina Mirceva), P.L. and E.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially financed by the Faculty of Computer Science and Engineering at the Ss. Cyril and Methodius University in Skojpe, Macedonia.

Data Availability Statement: The data presented in this study are openly available at their respective publishers. The Trains dataset [27] is available at https://relational-data.org/dataset/Trains (accessed on 27 February 2024). The Carcinogenesis dataset [33] is available at https://relational-data.org/dataset/Carcinogenesis (accessed on 27 February 2024). The Mutagenesis dataset [34] is available as separate two datasets at https://relational-data.org/dataset/Mutagenesis (accessed on 27 February 2024). The IMDB dataset is available at http://www.webstepbook.com/supplements/databases/imdb.sql (accessed on 27 February 2024). The MovieLens dataset is available at https://relational-data.org/dataset/MovieLens (accessed on 27 February 2024).

Acknowledgments: This article is based upon work from COST Action CA19121 (Good Brother: Network on Privacy-Aware Audio- and Video-Based Applications for Active and Assisted Living), supported by COST (European Cooperation in Science and Technology). COST is a funding agency for research and innovation networks. COST Actions help connect research initiatives across Europe and enable scientists to grow their ideas by sharing them with their peers. This boosts their research, career, and innovation. More information at https://www.cost.eu (accessed on 27 February 2024).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Grzegorowski, M.; Zdravevski, E.; Janusz, A.; Lameski, P.; Apanowicz, C.; Slezak, D. Cost Optimization for Big Data Workloads Based on Dynamic Scheduling and Cluster-Size Tuning. *Big Data Res.* **2021**, *25*, 100203. [CrossRef]
2. Zdravevski, E.; Lameski, P.; Dimitrievski, A.; Grzegorowski, M.; Apanowicz, C. Cluster-size optimization within a cloud-based ETL framework for Big Data. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 3754–3763. [CrossRef]
3. Zdravevski, E.; Lameski, P.; Kulakov, A.; Jakimovski, B.; Filiposka, S.; Trajanov, D. Feature Ranking Based on Information Gain for Large Classification Problems with MapReduce. In Proceedings of the 2015 IEEE Trustcom/BigDataSE/ISPA, Helsinki, Finland, 20–22 August 2015; Volume 2, pp. 186–191. [CrossRef]
4. Piatetsky-Shapiro, G. An Overview of Knowledge Discovery in Databases: Recent Progress and Challenges. In *Proceedings of the Rough Sets, Fuzzy Sets and Knowledge Discovery*; Ziarko, W.P., Ed.; Springer: London, UK, 1994; pp. 1–10.
5. Fayyad, U., Knowledge Discovery in Databases: An Overview. In *Relational Data Mining*; Džeroski, S., Lavrač, N., Eds.; Springer: Berlin/Heidelberg, Germany, 2001; pp. 28–47. [CrossRef]
6. Muggleton, S.; de Raedt, L. Inductive Logic Programming: Theory and methods. *J. Log. Program.* **1994**, *19–20*, 629–679; Special Issue: Ten Years of Logic Programming. [CrossRef]
7. Lavrač, N.; Džeroski, S. *Inductive Logic Programming: Techniques and Applications*; Prentice-Hall: Hoboken, NJ, USA, 1994.
8. Gärtner, T. *Kernels for Structured Data*; World Scientific: Singapore, 2008. [CrossRef]
9. Lavrač, N.; Podpečan, V.; Robnik-Šikonja, M. *Representation Learning: Propositionalization and Embeddings*; Springer: Berlin/Heidelberg, Germany, 2021.
10. Perovšek, M.; Vavpetič, A.; Cestnik, B.; Lavrač, N. A Wordification Approach to Relational Data Mining. In Proceedings of the Discovery Science, Singapore, 6–9 October 2013; Fürnkranz, J., Hüllermeier, E., Higuchi, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 141–154.
11. Lavrač, N.; Škrlj, B.; Robnik-Sikonja, M. Propositionalization and Embeddings: Two Sides of the Same Coin. *Mach. Learn.* **2020**, *109*, 1465–1507. [CrossRef]
12. Codd, E.F. Further Normalization of the Data Base Relational Model. *Data Base Syst.* **1972**, *6*, 33–64.

13. Lavrač, N.; Džeroski, S.; Grobelnik, M. *Learning Nonrecursive Definitions of Relations with Linus*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 265–281. [CrossRef]
14. Lavrač, N.; Flach, P. An extended transformation approach to Inductive Logic Programming. *ACM Trans. Comput. Log.* **2000**, *2*, 458–494. [CrossRef]
15. Krogel, M.A.; Wrobel, S. Transformation-Based Learning Using Multirelational Aggregation. In Proceedings of the International Conference on Inductive Logic Programming, Strasbourg, France, 9–11 September 2001.
16. Knobbe, A.J.; de Haas, M.; Siebes, A. Propositionalisation and Aggregates. In Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery, Freiburg, Germany, 3–5 September 2001.
17. Kanter, J.M.; Veeramachaneni, K. Deep feature synthesis: Towards automating data science endeavors. In Proceedings of the 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Paris, France, 19–21 October 2015; pp. 1–10.
18. Alfred, R. The Study of Dynamic Aggregation of Relational Attributes on Relational Data Mining. In Proceedings of the International Conference on Advanced Data Mining and Applications, Harbin, China, 6–8 August 2007.
19. Lodhi, H. Deep Relational Machines. In Proceedings of the International Conference on Neural Information Processing, Daegu, Republic of Korea, 3–7 November 2013.
20. Srinivasan, A.; Vig, L.; Bain, M. Logical Explanations for Deep Relational Machines Using Relevance Information. *J. Mach. Learn. Res.* **2018**, *20*, 130:1–130:47.
21. Wu, L.Y.; Fisch, A.; Chopra, S.; Adams, K.; Bordes, A.; Weston, J. StarSpace: Embed All The Things! *arXiv* **2017**, arXiv:abs/1709.03856.
22. Cvetkov-Iliev, A.; Allauzen, A.; Varoquaux, G. Relational data embeddings for feature enrichment with background information. *Mach. Learn.* **2023**, *112*, 687–720. [CrossRef]
23. Sourek, G. Deep Learning with Relational Logic Representations. In Proceedings of the International Joint Conference on Artificial Intelligence, Macao, China, 10–16 August 2019.
24. Sourek, G.; Železný, F.; Kuželka, O. Beyond graph neural networks with lifted relational neural networks. *Mach. Learn.* **2020**, *110*, 1695–1738. [CrossRef]
25. Fey, M.; Hu, W.; Huang, K.; Lenssen, J.E.; Ranjan, R.; Robinson, J.; Ying, R.; You, J.; Leskovec, J. Relational Deep Learning: Graph Representation Learning on Relational Databases. *arXiv* **2023**, arXiv:abs/2312.04615.
26. Zahradník, L.; Neumann, J.; Šír, G. A Deep Learning Blueprint for Relational Databases. In Proceedings of the NeurIPS 2023 Second Table Representation Learning Workshop, New Orleans, LA, USA, 10–16 December 2023.
27. Michie, D.; Muggleton, S.H.; Page, D.L.; Srinivasan, A. *To the International Computing Community: A New East-West Challenge*; Oxford University Computing Laboratory: Oxford, UK, 1994.
28. Freund, Y.; Schapire, R.E. A decision-theoretic generalization of online learning and an application to boosting. In Proceedings of the European Conference on Computational Learning Theory, Barcelona, Spain, 13–15 March 1995. [CrossRef]
29. Liaw, A.; Wiener, M.C. Classification and Regression by randomForest. *R News* **2022**, *2*, 18–22.
30. Geurts, P.; Ernst, D.; Wehenkel, L. Extremely randomized trees. *Mach. Learn.* **2006**, *63*, 3–42. [CrossRef]
31. Friedman, J.H. Greedy function approximation: A gradient boosting machine. *Ann. Stat.* **2001**, *29*, 1189–1232. [CrossRef]
32. Friedman, J.H. Stochastic gradient boosting. *Comput. Stat. Data Anal.* **2002**, *38*, 367–378. [CrossRef]
33. Srinivasan, A.; King, R.D.; Muggleton, S.H.; Sternberg, M.J.E. Carcinogenesis Predictions Using ILP. In Proceedings of the ILP, Prague, Czech Republic, 17–20 September 1997.
34. Debnath, A.K.; de Compadre, R.L.L.; Debnath, G.; Shusterman, A.J.; Hansch, C. Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. Correlation with molecular orbital energies and hydrophobicity. *J. Med. Chem.* **1991**, *34*, 786–97. [CrossRef]