Towards Improving Human Arithmetic Learning using Machine Learning

Tessa Hall and Herman Kamper
Department of Electrical and Electronic Engineering
Stellenbosch University, South Africa
tessah17@gmail.com, kamperh@sun.ac.za

***Abstract*—Basic arithmetic is an essential skill that is used in almost all career paths in some way. Ensuring that young children have a solid foundation in simple mathematical concepts is a worldwide goal and new methods to improve arithmetic learning are constantly being developed. Our aim is to utilise machine learning to assist learners with developing their basic mathematics skills by identifying the types of problems a user struggles with and presenting them with targeted questions to improve in these areas. In this paper we focus only on the prediction component: given a set of arithmetic questions and corresponding answers, can we predict which future questions a user will answer incorrectly? The accuracy and suitability of four machine learning models are evaluated using data from computer-generated agents as well as human users. On simulated agents, our models achieve accuracies of around 79% to 96% with decision trees performing the best. On human data, our models achieve accuracies in the range of 63% to 69%, with the decision tree once again outperforming other approaches. We hope that these error predictions models could be incorporated into future E-learning systems targeted at human arithmetic learning.**

***Index Terms*—E-learning, education, human error prediction.**

I. Introduction

Electronic learning (or E-learning) systems are gradually being utilised more and more in modern society at different levels of education [1]. This is largely due to the many benefits associated with these systems. They allow for flexibility in both the location of the learning environment and the pace at which learners can work [2]. Additionally, studies show that both teachers and learners display an interest in using these systems in parallel with the conventional classroom learning set-up, as they believe it will allow for more effective learning [3], [4]. If E-learning is used effectively, it can help address prevalent issues with regards to inequality in access to education and generally improve the quality of education worldwide.

Basic arithmetic is an essential skill that is critical in all walks of life, but many people do not have a strong foundation in this area. In South Africa in particular, the basic numeracy ability of young school children is found to be substandard. The poor performance of matric students in mathematics can partly be attributed to this weak foundation in basic mathematics [5]. Developing arithmetic-focused E-learning systems could help to improve the basic mathematical abilities of both those with and without access to other conventional teaching systems [6].

Machine learning techniques for clustering, regression and classification can be utilised to great effect to develop and improve E-learning systems in a number of ways [7]. For instance, it can be used to implement evaluation and assessment systems, facilitate effective group learning, personalise learning systems to users, and analyse system load [8]. Since learners tend to have differing levels of knowledge on various topics and different preferred learning styles, an adaptive system that can be personalised to them would help to facilitate more effective learning [9]. This is widely recognised, and as a result, the main focus of machine learning in E-learning is in the area of personalisation. One study investigated the personalisation of curriculum sequencing that takes into account the difficulty level of the proposed course relative to the apparent ability of the user [9]. A similar study used machine learning to assess the knowledge levels of learners in order to dynamically select the best autonomous evaluator [10]. Joseph [11] used machine learning in a system for the automatic monitoring and feedback of E-learning impact in a similar way to how teacher supervision in a class works. These studies all focused on the broader aspects of large E-learning systems and how to personalise them to the different learning styles of the users.

Here we focus on a specific area of study: basic arithmetic. Similar to [10], we investigate different classification models to assess human performance in some task. In this case, the task is to predict which basic arithmetic questions a user is likely to get wrong based on questions they have already answered. Four different classification models are investigated to determine which system can most accurately predict the correctness of a user's future answers. We evaluate the different models on two types of users: simulated computer agents, who makes consistent errors on specific types of questions, and actual humans. We show that the models perform better on simulated than on real data; our best model on human data still achieves an accuracy of 69%, giving a strong enough signal to make our approach relevant for practical E-learning applications.

II. System Design

Our final system is envisioned as follows: a user will be asked a number of randomly generated arithmetic questions with the system keeping track of what they get right and wrong. After a predetermined number of questions, the system will start to train a classifier which attempts to predict the type of questions that the user consistently answers incorrectly. The end aim is to present the user with targeted questions in the areas where they performed badly, thereby improving

978-1-7281-4162-6/20/$31.00 © 2020 IEEE

their skills in the areas where they fall short. In this paper, we only consider the component for predicting whether 'n question will be answered correctly or incorrectly; we present a focused investigation with the aim of evaluating different error classifiers on both simulated and actual human data. In this section the machine learning models used to make these predictions are discussed, followed by the input feature representations and structures used to implement the models.

A. Classifier Models

Four machine learning classification models are considered: decision trees, k-nearest neighbours, logistic regression and neural networks [12]. These models are all implemented using the scikit-learn (sklearn) Python machine learning library [13].

a) Decision Trees: A decision tree is a classification algorithm that is easy to understand and visualise. They are similar to flow charts consisting of a "root" node, several internal "test" nodes and a number of terminal nodes also known as "leaves". At these terminal nodes, data is assigned a class. Each internal node evaluates a feature in the test data according to a condition learned from the training data [14]. To learn these conditions, the Gini Index is used to calculate the impurity of the data at each node:

$$\text{Gini} = 1 - p_0^2 - p_1^2 \tag{1}$$

In this equation, p_0 is the fraction of data in class 0 and p_1 is the fraction of data in class 1, with a lower number indicating a purer node (Gini $= 0$ if all items comes from a single class) [15]. A tree is "grown" from the training data by first pooling all the data into a single node. Nodes are then repeatedly split using the condition which results in the best purity improvement according to the Gini Index. Splits are performed greedily until the improvement falls below a threshold. We use the default sklearn parameters.

b) k-Nearest Neighbours: The k-nearest neighbours algorithm classifies data simply based on the similarity of a test point to items in the training data. Our model implementation uses the Euclidean distance between the $k = 3$ nearest data points to the data point under consideration to predict its class. This k-value produced the best results in initial tests performed using agent data. We also use the sklearn weighting method, which places a larger weight on data points closer to the test point [13].

c) Logistic Regression: Logistic regression is a widely-used algorithm that can be utilised to classify data into two or more categories. The hypothesis used for logistic regression is known as the sigmoid function:

$$h(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathrm{T}} \cdot \boldsymbol{x}}} \tag{2}$$

with $\boldsymbol{x}$ representing the feature vector and $\boldsymbol{\theta}$ the weight vector, which is learned from the training data. The result of this hypothesis always lies between 0 to 1 and is interpreted as the probability that the test data will be classified as a class 1 [16]. Again the default sklearn parameters are used.

d) Neural Networks: A multi-layer perceptron classifier is an artificial neural network consisting of an input layer, an output layer and at least one hidden layer. The model is trained using backpropagation [16]. Our model uses one hidden layer with 100 neurons, Adam optimisation, and the rectified linear unit activation function. The maximum number of iterations allowable if convergence is not achieved is set to 1000 to ensure that the model can be trained in a time period suitable for use in real-time E-learning systems [13].

B. Feature Design

We investigate several different feature representations as inputs to the different machine learning models.

a) Base Features: The base feature vector consists of six features as shown in Table I. The first four features are used to represent the operator of the arithmetic question in binary. One of these features is set to 1 to represent the relevant operator, whereas the other three are set to 0. The last two features contain the numerical values of the first and second numbers in the calculation.

TABLE I
THE BASE FEATURE SYSTEM.

Features	**Description**
Feature 1	Binary representation for addition
Feature 2	Binary representation for subtraction
Feature 3	Binary representation for multiplication
Feature 4	Binary representation for division
Feature 5	The first number of the calculation
Feature 6	The second number of the calculation

b) High-Level Features: The high-level feature vector consists of the six base features from Table I extended with the twelve additional features shown in Table II. These additional features provide information about five potential prime factors of the question's numerical components and also identify whether these components are themselves prime numbers. This feature system is specifically aimed at improving the accuracy of the decision tree models by providing special features that can be considered in the trees' test nodes to capture specific systematic user errors.

c) Polynomial Features: The polynomial feature vector consists of the six base features from Table I extended with the seven additional features shown in Table III. The polynomial

TABLE II
THE HIGH-LEVEL FEATURE SYSTEM.

Features	**Description**
Features 7/8	Binary status of the first/second number as primes
Features 9/10	Binary status of the first/second number as divisible by 2
Features 11/12	Binary status of the first/second number as divisible by 3
Features 13/14	Binary status of the first/second number as divisible by 5
Features 15/16	Binary status of the first/second number as divisible by 7
Features 17/18	Binary status of the first/second number as divisible by 11

TABLE III
THE POLYNOMIAL FEATURE SYSTEM. Num_1 AND Num_2 ARE THE FIRST AND SECOND NUMBER OF THE ARITHMETIC CALCULATION, RESPECTIVELY.

Features	Description
Feature 7	Num_1 x Num_2
Feature 8	Num_1^2
Feature 9	Num_2^2
Feature 10	Num_1^3
Feature 11	Num_2^3
Feature 12	Num_1^2 x Num_2
Feature 13	Num_2^2 x Num_1

features are obtained by multiplying the two numerical components of the arithmetic questions by themselves and each other up to the third degree. This system was specifically aimed at improving the accuracy of the logistic regression models by allowing for the creation of non-linear decision boundaries.

C. Model Structures

Classifying human mistakes for addition and subtraction questions could conceivably be very different from classifying multiplication or division questions. It is therefore possible that some of the models and feature systems above could be better matched to arithmetic questions of a specific type. To investigate this, we consider two model structures.

a) Non-Hierarchical: The non-hierarchical model implementation utilises only one model that is trained on the feature vectors from each of the three feature systems. This model can then be used to predict the expected results for any arithmetic question type.

b) Hierarchical: In the hierarchical implementation, we first predict the probability of a user getting a question of a specific operator wrong (addition, subtraction, multiplication or division). This is done using a logistic regression model trained only on the binary features used to represent the operator of the arithmetic questions. Based on the question type, prediction is then performed using separate models each trained on different subdivisions of the training data according to operator. The models types are varied between the four under investigation. Figure 1 shows a breakdown of this hierarchical structure. The operator features in Table I are removed for the purpose of this implementation, but otherwise the feature vectors are the same as detailed in Tables I, II and III.

III. Experimental Method

Our overall system repeatedly asks a user to answer randomly generated arithmetic questions; we then attempt to classify which types of questions the user gets wrong. In this section, we describe the full methodology used to evaluate the error prediction ability of the different models for both simulated and actual human users.

A. Data

One hundred random basic arithmetic questions are generated with addition and subtraction question components ranging between 1 and 100 and multiplication and division components ranging between 1 and 12. For the input feature vectors, these numbers are normalised to lie within the range of 0 to 1. As binary targets, all questions answered correctly are assigned a value of 1 and questions answered incorrectly are assigned a 0. We use the default sklearn parameters for all our models, and therefore do not require any tuning data; for each user, the first fifty questions are used as training data and the remaining fifty as evaluation data. We consider two types of users.

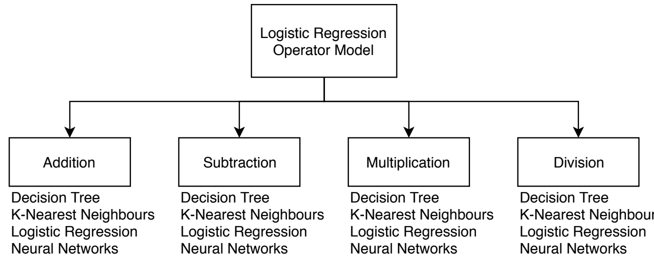

Fig. 1. The hierarchical implementation of our approach. Separate models classify different question types, depending on the operator used in the question.

a) Computer-Generated Agents: A number of computer agents are created to simulate users. These agents are designed to consistently answer specific types of arithmetic questions incorrectly, allowing us to perform experiments under idealised conditions. Six agents are used, as shown in Table IV. Four different sets of one hundred arithmetic questions are generated and answered by the agents. Evaluation metrics (see below) are aggregated over these four sets and over the agents.

b) Human Users: In order to evaluate the performance of the different models in a real-world environment, we collected data from eight human users. Users were provided with a simple computer application that asked 100 randomly generated arithmetic questions allowing for six seconds to answer each question. The users' answers to the questions are compared with the correct answers in order to generate the required binary target vectors.

B. Evaluation

Each of the four classification models (Section II-A) are trained using the training sets for all three feature systems

TABLE IV
DESCRIPTIONS OF THE COMPUTER-GENERATED AGENTS.

Agents	Type of Incorrect Answer
Agent 1	All addition questions.
Agent 2	All multiplication questions where either the first or second number is equal to 7.
Agent 3	All division questions were the second number is equal to 3.
Agent 4	All subtraction questions where the second number is even and the first number is larger than 50.
Agent 5	All multiplication questions where the second number is smaller than 6 and all addition questions where the first number is larger than 40.
Agent 6	All division questions where the second number is uneven and some random number (between 0 and 1) is larger than 0.4.

	Actual Positive (0)	Actual Negative (1)
Predicted Positive (0)	True Positive (TP)	False Positive (FP)
Predicted Negative (1)	False Negative (FN)	True Negative (TN)

Fig. 2. The modified confusion matrix used for model evaluation, with the values in parenthesis indicating whether a question was answered incorrectly (0) or correctly (1).

(Section II-B), for both the hierarchical and non-hierarchical structures (Section II-C). Thereafter, the evaluation sets are used to compare model predictions with the true labels.

Since we are interested mainly in when a user answers a question incorrectly (labelled as class 0), we use a variation on a typical confusion matrix. Figure 2 shows the confusion matrix used with the positive state corresponding to class 0 (incorrect answer) whereas the negative state corresponds to class 1 (correct answer). Our main interest is in the number of TP predictions, indicating that a model positively identifies the questions answered incorrectly in the evaluation data.

We also use the balanced prediction accuracy of the various models as an additional metric:

$$\text{Accuracy} = 0.5 \times \frac{\text{TP}}{\text{TP}+\text{FN}} + 0.5 \times \frac{\text{TN}}{\text{TN}+\text{FP}} \tag{3}$$

We use this balanced metric instead of standard accuracy, since users will often answer questions correctly most of the time, causing a large class imbalance.

IV. Results

A. Computer-Generated Agents

We first consider model performance on the computer-generated agents. Table V and Figure 3 provide a breakdown of the confusion matrix and evaluation accuracies obtained for both hierarchical and non-hierarchical model implementations aggregated across all feature systems. Although the differences between the different models are small, they are consistent. The results indicate that the hierarchically structured models are more accurate than the non-hierarchical models. Decision trees perform the best with the highest accuracy and the most true positive predictions. The decision tree models, however, produce the second-most false positive results after the k-nearest neighbours models. This may indicate that they detect additional unintentional patterns in the agent data. Logistic regression models are the least effective, with the non-hierarchical models predicting 25.5% less true positive results than the best hierarchical decision trees.

Tables VI and VII provide a breakdown of the confusion matrix and evaluation accuracies obtained for the different feature systems of the non-hierarchical and hierarchical model implementations, respectively. Figure 4 is derived from these

TABLE V
Results for the hierarchical (H) and non-hierarchical decision tree (DT), k-nearest neighbours (KNN), logistic regression (LR) and neural network (NN) models on agent data.

Model: (H) for Hierarchy	**True Positive**	**False Positive**	**True Negative**	**False Negative**	**Accuracy**
DT (H)	432	104	3022	42	93.91%
DT	415	108	3018	59	92.05%
KNN (H)	403	126	3000	71	90.50%
KNN	390	138	2988	84	88.93%
LR (H)	369	96	3030	105	87.39%
LR	322	81	3045	152	82.67%
NN (H)	405	102	3024	69	91.09%
NN	366	71	3055	108	87.47%

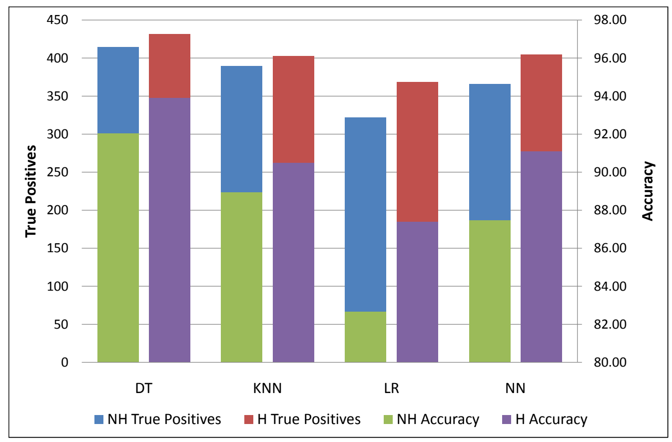

Fig. 3. A comparison of hierarchical (H) and non-hierarchical (NH) decision tree (DT), k-nearest neighbours (KNN), logistic regression (LR) and neural network (NN) models across all three feature systems on agent data.

tables and shows which feature systems are the most effective for each model on the computer-generated data.

On the simulated computer-agent data, the most effective model overall is the hierarchical decision tree using high-level features. This implementation achieves the highest accuracy and produces the most true positive predictions of all 24 models. The non-hierarchical implementation of the logistic regression models using the high-level feature system performs the worst.

B. Human Users

Table VIII and Figure 5 provide a breakdown of the confusion matrix and evaluation accuracy results obtained for both the hierarchical and non-hierarchical models aggregated across all feature systems on human user data. Compared to the performance on the simulated agent data (Table V and Figure 3), the results on real humans are worse; this indicates (not surprisingly) that the types of arithmetic mistakes that humans make are much less systematic than those of simulated agents.

Here, the decision trees produce the most true positive results, and the hierarchical implementation has the second highest evaluation accuracy of all of the models. The logistic regression models have similarly high evaluation accuracies, but predict

TABLE VI
RESULTS FOR THE NON-HIERARCHICAL DECISION TREE (DT), k-NEAREST NEIGHBOURS (KNN), LOGISTIC REGRESSION (LR) AND NEURAL NETWORK (NN) MODELS FOR EACH FEATURE SYSTEM ON AGENT DATA.

Model: (B) Base (H) High (P) Poly	True Positive	False Positive	True Negative	False Negative	Accuracy
DT (B)	136	41	1001	22	91.07%
DT (H)	146	32	1010	12	94.67%
DT (P)	133	35	1007	25	90.41%
KNN (B)	138	48	994	20	91.37%
KNN (H)	117	39	1003	41	85.15%
KNN (P)	135	51	991	23	90.27%
LR (B)	113	36	1006	45	84.03%
LR (H)	95	17	1025	63	79.25%
LR (P)	114	28	1014	44	84.73%
NN (B)	121	25	1017	37	87.09%
NN (H)	118	24	1018	40	86.19%
NN (P)	127	22	1020	31	89.13%

TABLE VII
RESULTS FOR THE HIERARCHICAL DECISION TREE (DT), k-NEAREST NEIGHBOURS (KNN), LOGISTIC REGRESSION (LR) AND NEURAL NETWORK (NN) MODELS FOR EACH FEATURE SYSTEM ON AGENT DATA.

Model: (B) Base (H) High (P) Poly	True Positive	False Positive	True Negative	False Negative	Accuracy
DT (B)	141	41	1001	17	92.65%
DT (H)	152	26	1016	6	96.85%
DT (P)	139	37	1005	19	92.21%
KNN (B)	136	47	995	22	90.78%
KNN (H)	132	35	1007	26	90.09%
KNN (P)	135	44	998	23	90.61%
LR (B)	119	40	1002	39	85.74%
LR (H)	127	32	1010	31	88.65%
LR (P)	123	24	1018	35	87.77%
NN (B)	136	27	1015	22	91.74%
NN (H)	130	36	1006	28	89.41%
NN (P)	139	39	1003	19	92.12%

the least true positive results. Except for the decision trees, the hierarchical implementations result in lower accuracies and less true positive predictions.

Tables IX and X provide a breakdown of the confusion matrix and evaluation accuracies obtained for the different feature systems of the non-hierarchical and hierarchical model implementations, respectively. Figure 6 is derived from these tables and shows which feature systems are the most effective for each model implementation on the human user data. The non-hierarchical decision trees using the base feature system predicts the most true positive results, but does not perform as well on accuracy. The non-hierarchical logistic regression model using the polynomial feature system has the highest accuracy, but does not predict as many true positive results as some of the other models. The hierarchical decision trees using the base feature system may provide a good compromise between accuracy and the number of true positive predictions, but this would depend on how we plan to generate targeted questions in our end application. This requires further investigation.

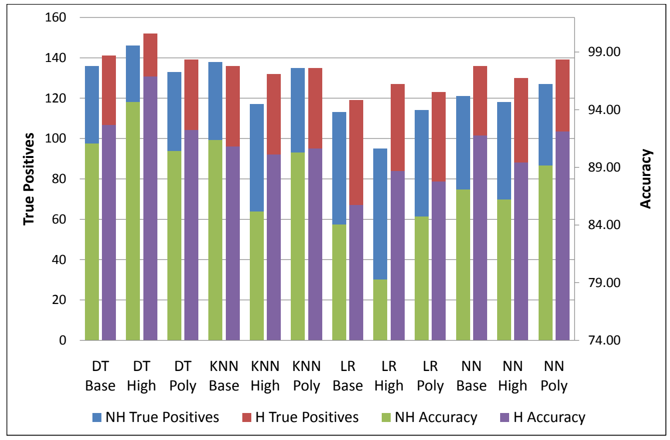

Fig. 4. A comparison of different feature systems for hierarchical (H) and non-hierarchical (NH) decision tree (DT), k-nearest neighbours (KNN), logistic regression (LR) and neural network (NN) models.

TABLE VIII
RESULTS FOR THE HIERARCHICAL (H) AND NON-HIERARCHICAL DECISION TREE (DT), k-NEAREST NEIGHBOURS (KNN), LOGISTIC REGRESSION (LR) AND NEURAL NETWORK (NN) MODELS ON HUMAN USER DATA.

Model: (H) for Hierarchy	True Positive	False Positive	True Negative	False Negative	Accuracy
DT (H)	138	196	755	111	67.41%
DT	138	224	727	111	65.93%
KNN (H)	120	170	781	129	65.16%
KNN	129	168	783	120	67.07%
LR (H)	115	130	821	134	66.26%
LR	116	113	838	133	67.35%
NN (H)	120	176	775	129	64.84%
NN	133	164	787	116	68.08%

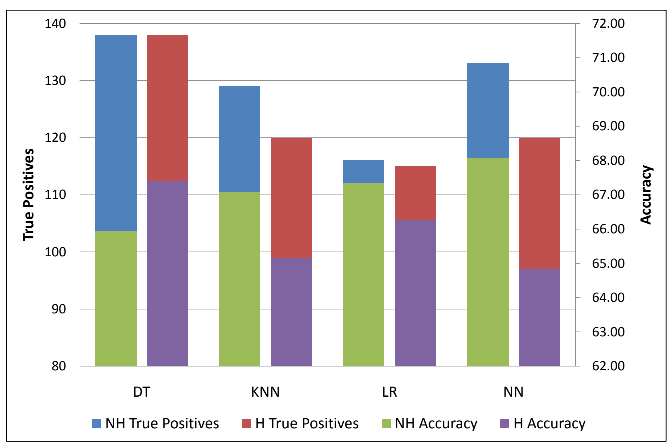

Fig. 5. A comparison of hierarchical (H) and non-hierarchical (NH) decision tree (DT), k-nearest neighbours (KNN), logistic regression (LR) and neural network (NN) models across all three feature systems on human user data.

V. Conclusion

We considered four types of machine learning classifiers, trained to predict the correctness of a user's answers to basic arithmetic questions. We evaluated models on simulated agents as well as real human participants. Our best overall model on the human data is a hierarchical decision tree, achieving an accuracy of 69%. Although this leaves room for improvement, it indicates that systematic human errors can be predicted, and that we that we can incorporate this error prediction approach within a larger E-learning system. Future work will consider the best model and approach for generating targeted questions to improve a user's arithmetic skills.

TABLE IX
Results for the non-hierarchical decision tree (DT), k-nearest neighbours (KNN), logistic regression (LR) and neural network (NN) models for each feature system on human user data.

Model: (B) Base (H) High (P) Poly	**True Positive**	**False Positive**	**True Negative**	**False Negative**	**Accuracy**
DT (B)	50	80	237	33	67.50%
DT (H)	47	75	242	36	66.48%
DT (P)	41	69	248	42	63.82%
KNN (B)	44	57	260	39	67.52%
KNN (H)	44	54	263	39	67.99%
KNN (P)	41	57	260	42	65.71%
LR (B)	36	34	283	47	66.32%
LR (H)	39	45	272	44	66.40%
LR (P)	41	34	283	42	69.34%
NN (B)	40	41	276	43	67.63%
NN (H)	49	75	242	34	67.69%
NN (P)	44	48	269	39	68.94%

TABLE X
Results for the hierarchical decision tree (DT), k-nearest neighbours (KNN), logistic regression (LR) and neural network (NN) models for each feature system on human user data.

Model: (B) Base (H) High (P) Poly	**True Positive**	**False Positive**	**True Negative**	**False Negative**	**Accuracy**
DT (B)	47	57	260	36	69.32%
DT (H)	48	72	245	35	67.56%
DT (P)	43	67	250	40	65.34%
KNN (B)	37	53	264	46	63.93%
KNN (H)	47	64	253	36	68.22%
KNN (P)	36	53	264	47	63.33%
LR (B)	33	38	279	50	63.89%
LR (H)	41	49	268	42	66.97%
LR (P)	41	43	274	42	67.92%
NN (B)	41	54	263	42	66.18%
NN (H)	38	60	257	45	63.43%
NN (P)	41	62	255	42	64.92%

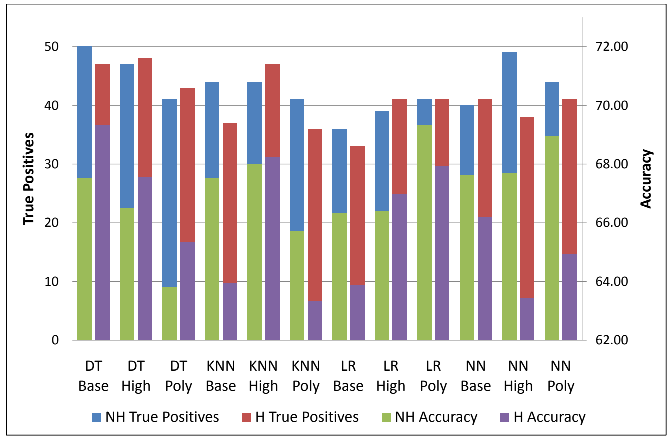

Fig. 6. A comparison of different feature systems for hierarchical (H) and non-hierarchical (NH) decision tree (DT), k-nearest neighbours (KNN), logistic regression (LR) and neural network (NN) models with human user data.

References

[1] S. Aljawarneh, Z. Muhsin, A. Nsour, F. Alkhateeb, and E. AlMaghayreh, "E-learning tools and technologies in education: A perspective," in *MIT LINC 2010*, 2010.

[2] S. Siddiq, "Machine learning in e-learning," Ph.D. dissertation, Dept. of Computer Science, Western University, 2017.

[3] E. Wakelam, A. Jeffries, N. Davey, and Y. Sun, "The potential for using artificial intelligence techniques to improve e-learning systems," in *European Conference on e-Learning (ECEL)*, 2015.

[4] A. Aregbesola, S. Tunmibi, P. Adejobi, and O. Ibrahim, "Impact of e-learning and digitalization in primary and secondary schools," *Journal of Education and Practice*, vol. 6, no. 17, pp. 53–58, 2015.

[5] P. Aunio, R. Mononen, L. Ragpot, and M. Törmänen, "Early numeracy performance of south african school beginners," *South African Journal of Childhood Education*, vol. 6, no. 1, 2016.

[6] M. Amer and F. A. Alnaja, "E-learning application to teaching mathematics," *International Journal of Management and Applied Science*, vol. 3, no. 9, pp. 81–83, 2017.

[7] M. Krendzelak, "Machine learning and its applications in e-learning systems," in *IEEE 12th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, 2014.

[8] A. Moubayed, M. Injadat, A. B. Nassif, H. Lutfiyya, and A. Shami, "E-learning: Challenges and research opportunities using machine learning —& data analytics," *IEEE Access*, vol. 6, pp. 39 117–39 138, 2018.

[9] C.-M. Chen, "Intelligent web-based learning system with personalized learning path guidance," *Computers —& Education*, vol. 51, no. 2, pp. 787–814, 2008.

[10] G. Nazeeh, "Knowledge level assessment in e-learning systems using machine learning and user activity analysis," *International Journal of Advanced Computer Science and Applications*, vol. 6, no. 4, pp. 107–113, 2015.

[11] H. R. Joseph, "Promoting education: A state of the art machine learning framework for feedback and monitoring e-learning impact," in *IEEE Global Humanitarian Technology Conference*, 2014.

[12] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer-Verlag, 2008.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[14] L. Rokach and O. Maimon, *Data Mining and Knowledge Discovery Handbook*. Springer, 2005.

[15] M. Steinbach, P.-N. Tan, V. Kumar, and A. Karpatne, *Introduction to Data Mining*. Pearson, 2005.

[16] A. Ng, "Machine learning." [Online]. Available: https://www.coursera.org/learn/machine-learning