УДК 004.8 DOi 10.30837/ bi.2021.1(96).08

M.O. Butsenko[1], I.V. Afanasieva[2], N.V. Golian[3], N. Kameniuk[4]

[1]backelor's student of the department of Software Engineering,
Kharkiv National University of Radio Electronics, Ukraine, mykyta.butsenko@nure.ua

[2]associate professor of the department of Software Engineering,
Kharkiv National University of Radio Electronics, Ukraine, iryna.afanasieva@nure.ua

[3]associate professor of the department of Software Engineering,
Kharkiv National University of Radio Electronics, Ukraine, nataliia.golian@nure.ua

[4]master's student of the department of Software Engineering,
Riga Technical University, Latvia, nataliia.kameniuk@gmail.com

A NEURAL NETWORK APPROACH FOR THE AUTOMATIC SELECTION OF A COMPLEX OF REHABILITATION EXERCISES

This article is devoted to solving the problem of automatic selection of a set of rehabilitation exercises during injuries, considering the state of the human cardiovascular system through the use of neural networks. To solve this problem, it was necessary to choose one of two classical approaches – multiclass classification or multilabel classification, each of which solves the problem of data classification through its own algorithm, and use the selected neural network architecture to create a software system. While working on this system, it was also necessary to solve certain problems related to each of these approaches (the need for a large sample due to the large number of exercises that the system should recommend) or a specific approach (inability to select multiple exercises at once – for Multiclass Classification, lower productivity and the number of supported programming languages – for Multilabel Classification).

Samples of different sizes (from 1 million records and more) were used to train the neural network, which were generated through a self-written program that generated a given number of records and wrote them to a .CSV (comma-separated values) file.

ANALYSIS, EXERCISE, MULTICLASS CLASSIFICATION, MULTILABEL CLASSIFICATION, NEURAL NETWORK, SOFTWARE SYSTEM, RECOMMENDATION, CARDIOVASCULAR SYSTEM, INJURY

Буценко М.О., Афанасьева И.В., Голян Н.В., Каменюк Н. Нейросетевой подход автоматического подбора комплекса реабилитационных упражнений. Эта статья посвящена решению проблемы автоматического подбора комплекса восстановительных упражнений при травмах с учетом состояния сердечно-сосудистой системы человека с помощью использования нейронных сетей. Для решения данной задачи необходимо было выбрать один из двух классических подходов – Multiclass Classification или Multilabel Classification, каждый из которых решает проблему классификации данных через собственный алгоритм, и использовать выбранную архитектуру нейронной сети для написания программной системы. Во время работы над этой системой также необходимо решить определенные проблемы, касающиеся каждого из этих подходов (необходимость в большой выборке из-за большого количества упражнений, которые система должна рекомендовать) или конкретного подхода (невозможность выбрать несколько упражнений одновременно – для Multiclass Classification, меньше производительность и количество поддерживаемых языков программирования – для Multilabel Classification).

Для обучения нейронной сети использовались выборки различных размеров (от 1 млн. записей и более), генерировались через самостоятельно написанную программу, которая генерировала заданное количество записей и записывала их в .CSV (comma-separated values) файл.

АНАЛИЗ, УПРАЖНЕНИЕ, МУЛЬТИКЛАССОВАЯ КЛАССИФИКАЦИЯ, МУЛЬТИЛЕЙБЛОВАЯ КЛАССИФИКАЦИЯ, НЕЙРОННАЯ СЕТЬ, ПРОГРАММНАЯ СИСТЕМА, РЕКОМЕНДАЦИЯ, СЕРДЕЧНО-СОСУДИСТАЯ СИСТЕМА, ТРАВМА

Буценко М.О., Афанасьєва І.В., Голян Н.В., Каменюк Н. Нейромережевий підхід для автоматичного підбору комплексу реабілітаційних вправ. Ця стаття присвячена рішенню проблеми автоматичного підбору комплексу відновлювальних вправ під час травм з урахуванням стану серцево-судинної системи людини за допомогою використання нейронних мереж. Для вирішення даної задачі необхідно було обрати один із двох класичних підходів – Multiclass Classification або Multilabel Classification, кожний з яких вирішує проблему класифікації даних через власний алгоритм, та використати обрану архітектуру нейронної мережі для написання програмної системи. Під час роботи над цією системою також необхідно було вирішити певні проблеми, що стосувалися кожного з цих підходів (необхідність у великій вибірці через велику кількість вправ, що система має рекомендувати) або конкретного підходу (неможливість обрати декілька вправ одночасно – для Multiclass Classification, менша продуктивність та кількість підтримуваних мов програмування – для Multilabel Classification).

Для навчання нейронної мережі використовувались вибірки різних розмірів (від 1 млн. записів та більше), що генерувались через самостійно написану програму, що генерувала задану кількість записів та записувала їх у .CSV (comma-separated values) файл.

АНАЛІЗ, ВПРАВА, МУЛЬТИКЛАСОВА КЛАСИФІКАЦІЯ, МУЛЬТИЛЕЙБЛОВА КЛАСИФІКАЦІЯ, НЕЙРОННА МЕРЕЖА, ПРОГРАМНА СИСТЕМА, РЕКОМЕНДАЦІЯ, СЕРЦЕВО-СУДИННА СИСТЕМА, ТРАВМА

1. Introduction

Every day, many people receive injuries of varying severity. In most cases, in order to fully restore the function of the injured part of the body, it is necessary to regularly perform restorative exercises, some of them due to the high intensity can put a heavy strain on the cardiovascular system [1].

Depending on the location and degree of damage, the patient needs restorative exercises of varying intensity.

Making the right list of such exercises can help information about the age and sex of the person. But the fact is that such data are not enough to make such a set of exercises that is guaranteed not to harm the health of the patient.

That is why there is a need to develop a system that provides a set of exercises for home use, namely physical therapy for injuries and fractures, considering the state of the cardiovascular system of the patient and his sex, as well as age.

2. Analysis of competitors and problem statement

There are systems [2, 3, 4] that provide general advice on a set of restorative exercises, but without reference to the state of the cardiovascular system. Simply put, they simply provide sets of restorative exercises.

That is, there is no example of software on the market that could provide a set of exercises, using not only the injury and the number of fractures, dislocations or sprains, but also related indicators of the cardiovascular system and blood tests.

Accordingly, the question arises as to how to analyze such an array of data. For this purpose, it is appropriate to use neural networks [5], as they can provide recommendations with some accuracy, using a sample, which is presented, for example, in the form of a .CSV-file.

An example of the structure of such a file is shown in Figure 1.

It can be seen that the data in it is divided into rows and columns, which in turn are separated by a comma with a space.

3. Multiclass classification

In this case, since we solve the problem of selecting a set of exercises with reference to certain indicators of the cardiovascular system, it is logical to use an approach called Multiclass Classification (it solves the problem of classifying specimens into one of several classes) [6].

The diagram showing the main problem solved by this approach is shown in Figure 2.

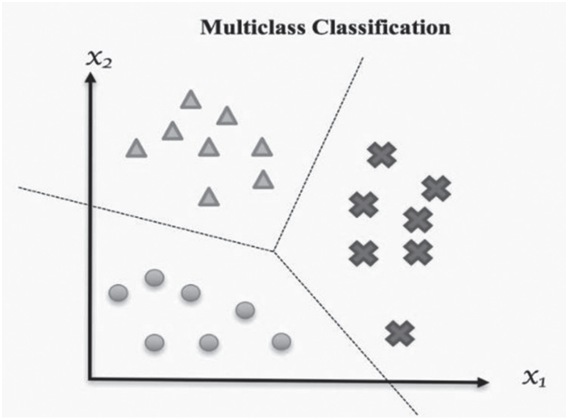
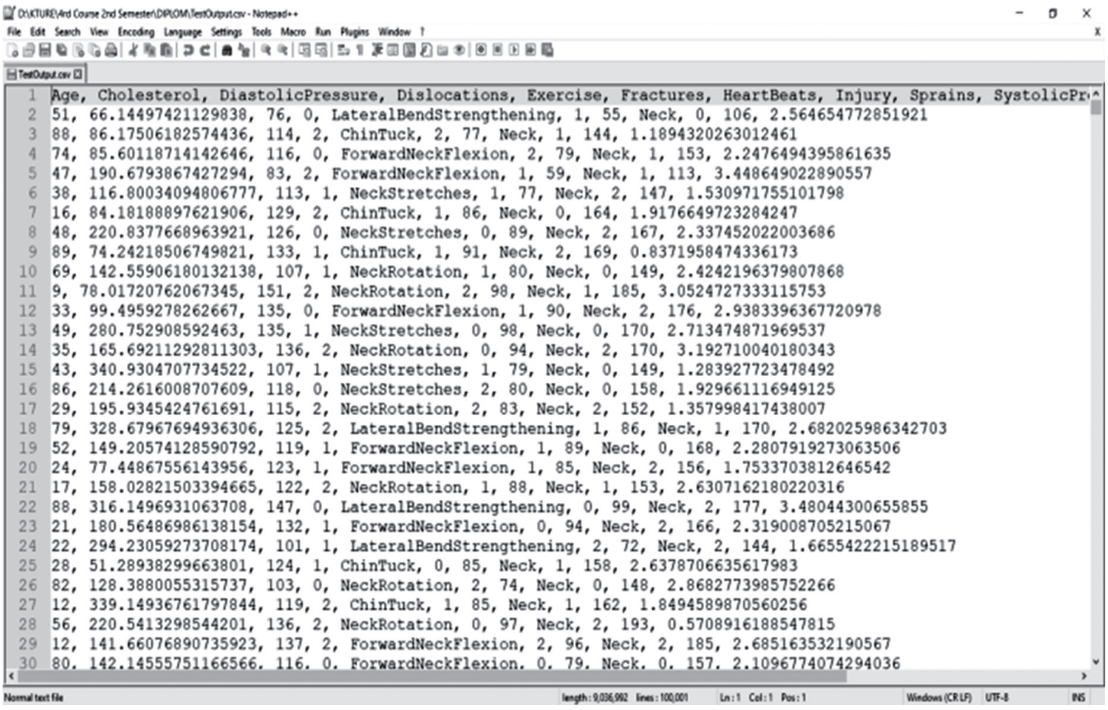


Fig. 2. A diagram illustrating the problem that the Multiclass classification solves

The algorithm responsible for the selection of exercises for recovery works as follows: the entrance is given the age



```
Age, Cholesterol, DiastolicPressure, Dislocations, Exercise, Fractures, HeartBeats, Injury, Sprains, SystolicPr
51, 66.14497421129838, 76, 0, LateralBendStrengthening, 1, 55, Neck, 0, 106, 2.564654772851921
88, 86.17506182574436, 114, 2, ChinTuck, 2, 77, Neck, 1, 144, 1.1894320263012461
74, 85.60118714142646, 116, 0, ForwardNeckFlexion, 2, 79, Neck, 1, 153, 2.2476494395861635
47, 190.6793867427294, 83, 2, ForwardNeckFlexion, 1, 59, Neck, 1, 113, 3.448649022890557
38, 116.80034094806777, 113, 1, NeckStretches, 1, 77, Neck, 2, 147, 1.530971755101798
16, 84.18188897621906, 129, 2, ChinTuck, 1, 86, Neck, 0, 164, 1.9176649723284247
48, 220.8377668963921, 126, 0, NeckStretches, 0, 89, Neck, 2, 167, 2.337452022003686
89, 74.24218506749821, 133, 1, ChinTuck, 1, 91, Neck, 2, 169, 0.8371958474336173
69, 142.55906180132138, 107, 1, NeckRotation, 1, 80, Neck, 0, 149, 2.4242196379807868
9, 78.01720762067345, 151, 2, NeckRotation, 2, 98, Neck, 1, 185, 3.0524727333115753
33, 99.4959278262667, 135, 0, ForwardNeckFlexion, 1, 90, Neck, 2, 176, 2.9383396367720978
49, 280.752908592463, 135, 1, NeckStretches, 0, 98, Neck, 0, 170, 2.713474871969537
35, 165.69211292811303, 136, 2, NeckRotation, 0, 94, Neck, 2, 170, 3.192710040180343
43, 340.9304707734522, 107, 1, NeckStretches, 1, 79, Neck, 0, 149, 1.283927723478492
86, 214.2616008707609, 118, 0, NeckStretches, 2, 80, Neck, 0, 158, 1.929661116949125
29, 195.9345424761691, 115, 2, NeckRotation, 2, 83, Neck, 2, 152, 1.357998417438007
79, 328.67967694936306, 125, 2, LateralBendStrengthening, 1, 86, Neck, 1, 170, 2.682025986342703
52, 149.20574128590792, 119, 1, ForwardNeckFlexion, 1, 89, Neck, 0, 168, 2.2807919273063506
24, 77.44867556143956, 123, 1, ForwardNeckFlexion, 1, 85, Neck, 2, 156, 1.7533703812646542
17, 158.02821503394665, 122, 2, NeckRotation, 1, 88, Neck, 1, 153, 2.6307162180220316
88, 316.1496931063708, 147, 0, LateralBendStrengthening, 0, 99, Neck, 2, 177, 3.48044300655855
21, 180.56486986138154, 132, 1, ForwardNeckFlexion, 0, 94, Neck, 2, 166, 2.319008705215067
22, 294.23059273708174, 101, 1, LateralBendStrengthening, 2, 72, Neck, 2, 144, 1.6655422215189517
28, 51.28938299663801, 124, 1, ChinTuck, 0, 85, Neck, 1, 158, 2.6378706635617983
82, 128.3880055315737, 103, 0, NeckRotation, 2, 74, Neck, 0, 148, 2.8682773985752266
12, 339.14936761797844, 119, 2, ChinTuck, 1, 85, Neck, 1, 162, 1.8494589870560256
56, 220.5413298544201, 136, 2, NeckRotation, 0, 97, Neck, 2, 193, 0.5708916188547815
12, 141.66076890735923, 137, 2, ForwardNeckFlexion, 2, 96, Neck, 2, 185, 2.685163532190567
80, 142.14555751166566, 116, 0, ForwardNeckFlexion, 0, 79, Neck, 0, 157, 2.1096774074294036
```

Fig. 1. An example of a .CSV file

and sex of the person, his injury, as well as indicators responsible for the work of his cardiovascular system.

These indicators include::

— systolic and diastolic pressure;

— patient's age (Age);

— as well as key indicators of recent blood tests - cholesterol and triglycerides [7].

The initial data in this case is the result provided by the input data: age, sex, pressure, blood tests, etc. A set of exercises represented as a single object, such as a string in a .CSV file. Together, these data form the training and test samples needed to train the model and form the neural network.

4. Multilabel classification

In addition to the Multiclass Classification approach, we can use the Multilabel classification [8]. An example is shown in Figure 3.

This approach is a generalized version of the multiclass classification, but with one difference — the Multiclass Classification provides only one object (label) as the source data, while the Multilabel classification has no restrictions on the amount of source data.

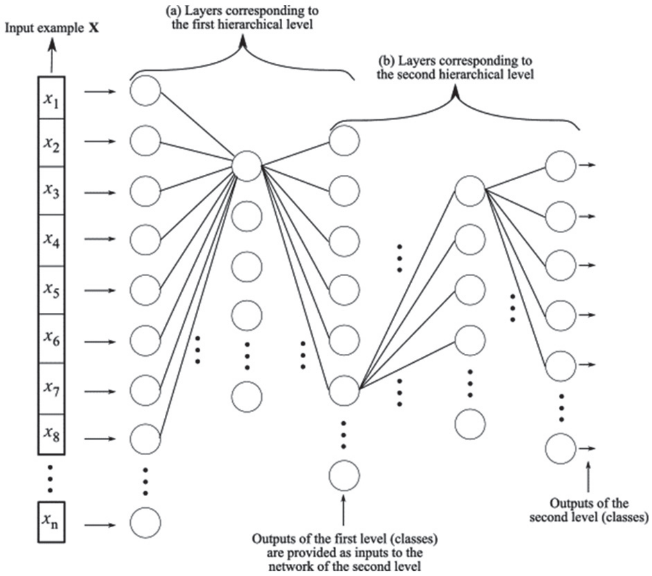


Fig. 3. An example of a Multilabel classification

That is, if the first approach provides one of the precreated complexes, then in the case of Multilabel classification, this complex is formed automatically from individual exercises.

5. Features of approaches

Figure 4 shows the difference between this approach and multiclass classification.

The Multilabel classification approach is potentially more accurate, but is not supported by all programming languages. For example, ML.NET — a framework from Microsoft for the C # programming language — does not currently support Multilabel classification; it is suggested to use Multiclass Classification instead.

Fig. 4. The main difference between Multiclass Classification and Multilabel classification

Also, this approach in most systems, including ours, is less productive, because the recommendation of a set of exercises is a costlier operation in terms of time and resources than the recommendation of a single exercise.

Both of these approaches are relevant to our system, but they have the same drawback — the need for a large sample.

For example, the San Francisco Department of Health Restaurant Table, which lists all inspections with detected violations and their low-medium-high-risk classifications, is posted on the Microsoft website and used to explain the Multiclass Classification using a forecast degree of risk [9], contains approximately 50 thousand records. That is, in order to teach the system to classify violations into three categories with high accuracy — in this case the accuracy is about 100% — a very large data set was needed.

Since there are only more than 3 types of injuries, and each injury requires 3-5 exercises to assemble recovery complexes, the first calculations to ensure high accuracy (90% or more) may require a large sample of tens of millions of records.

Such an array of data can be obtained only by collecting data from a large number of hospitals, or by generating them yourself, referring to scientific advice on the selection of exercises for recovery from injury.

6. Neural network architecture using ML.NET and multiclass classification

At the beginning of the work it is necessary to form a sample, which is used to create a neural network.

The sample size directly depends on the exercises that the system can recommend for recovery — as the number of exercises increases, the sample itself should increase to ensure high accuracy of the recommendation (90% or more).

After its formation it is necessary to pass to creation of a neural network. With the ML.NET framework, this can be done in two ways: through the Model Builder GUI or directly through the API. In the case of Multiclass Classification, it is appropriate to use Model Builder, because it supports the data approach. The principle of its operation is shown in Figures 5 and 6. They show that when using it ML.NET will independently divide the sample into Test and Train Data and form a neural network; for this he only needs a sample with the specified Features and Label.

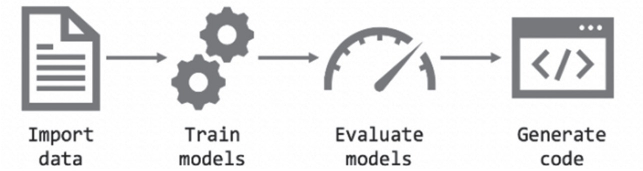
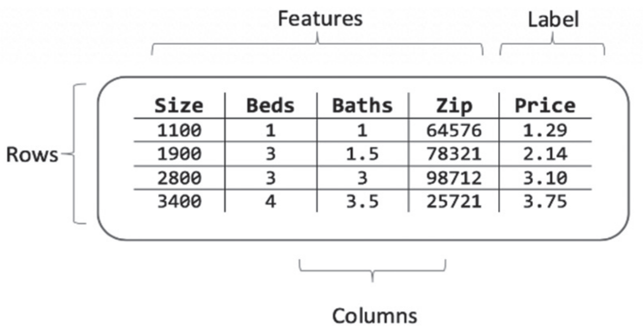


Fig. 5. The principle of ML.NET when using Model Builder

At the Figure 6 we can see also sample distribution on Features and Label.



Size	Beds	Baths	Zip	Price
1100	1	1	64576	1.29
1900	3	1.5	78321	2.14
2800	3	3	98712	3.10
3400	4	3.5	25721	3.75

Fig. 6. Sample distribution on Features and Label

The neural network is in the layer of business logic; it must provide high accuracy in order to adequately recommend exercises for rehabilitation after injuries.

The system should expand the sample provided in order to continuously train the neural network.

For ease of use of the neural network, the sample is stored in a separate database table.

The structure (columns) of this table are as follows:

- injury ID (InjuryID);
- patient's age (Age);
- fractures quantity (Fractures);
- dislocations quantity (Dislocations);
- sprains quantity (Sprains);
- cholesterol;
- triglycerides;
- systolic pressure (SystolicPressure);
- diastolic pressure (DiastolicPressure);
- heart beats per minute (HeartBeats);
- exercise ID (ExerciseID).

This table uses trauma and exercise identifiers, as this table is not isolated throughout the system, but is a full participant.

There are other tables in the system, in particular for the same injuries and exercises referred to in the table above, using these identifiers. It is designed so that when deleting any injury or exercise (i.e., records from the main tables), its records that use the identifiers of the deleted entities are not deleted by the system, but are ignored during the selection of recommended injuries for recovery.

In the future, they can be used again if the deleted records are restored while retaining their IDs.

The part of the system that uses the created neural network is presented as a separate project. It receives requests from controllers that use HTTP methods (mainly Get and Post versions), which in turn receive requests from medical devices that collect indicators of the state of the cardiovascular system, or from the emulator of these devices.

The neural network analyzes the obtained indicators and selects the exercise or their complex according to the table above.

After the work is done, it sends a signal to the controller that the selection process has been successfully completed, and the controller sends it to the medical device or emulator. After that, the user is expected to go to the client part of the application to view the result in the form of selected exercises, as well as collected tests for a general understanding of the state of his cardiovascular system.

7. Integration and implementation

The neural network project is integrated into the system at the level of business logic. With the ML.NET feature, you can access non-transferable data types transferred to their current table, which can be written as an optional table that stores files created through files.

The file was generated through an individual algorithm, which randomly generated age and sex (age, gender), then selected successors according to individual medical rules:

- cholesterol;
- triglycerides;
- systolic pressure (SystolicPressure);
- diastolic pressure (DiastolicPressure);
- heart beats per minute (HeartBeats)

Examples of value selection rules are given in the code below. They use the age and sex of the user as initial parameters, and at the output give indicators of cardiovascular condition in the form of systolic, diastolic pressures and heart rate.

```
switch (gender)
    {
      case MaleGender:
        switch (age)
        {
          case >= MinAge and < 20:
            heartParametersTuple.systolicPressure =
NumbersGenerator.GenerateRandomInt(114, 126);
            heartParametersTuple.diastolicPressure =
NumbersGenerator.GenerateRandomInt(70, 83);
            heartParametersTuple.heartBeats =
NumbersGenerator.GenerateRandomInt(60, 80);
            break;
          case >= 20 and < 30:
            heartParametersTuple.systolicPressure =
NumbersGenerator.GenerateRandomInt(120, 131);
            heartParametersTuple.diastolicPressure =
NumbersGenerator.GenerateRandomInt(74, 86);
```

```
                    heartParametersTuple.heartBeats =
NumbersGenerator.GenerateRandomInt(50, 90);
                    break;
                case >= 30 and < 40:
                    heartParametersTuple.systolicPressure =
NumbersGenerator.GenerateRandomInt(124, 133);
                    heartParametersTuple.diastolicPressure =
NumbersGenerator.GenerateRandomInt(76, 88);
                    heartParametersTuple.heartBeats =
NumbersGenerator.GenerateRandomInt(60, 90);
                   break;
}
```

After generating the values, this .CSV file was used as the source tool for learning the system. Figure 7 shows the first step of adding a note to a neural network as a sample for model learning. It is from this sample that the parameters used as properties in the model are selected.

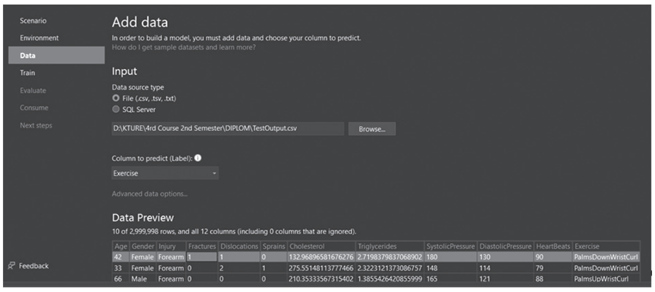


Fig. 7. Initial process of adding data to ML.Net through Model Builder

Figure 8 shows an example estimate that the system uses to train a neural network. The Time to train value increases in proportion to the sample size.

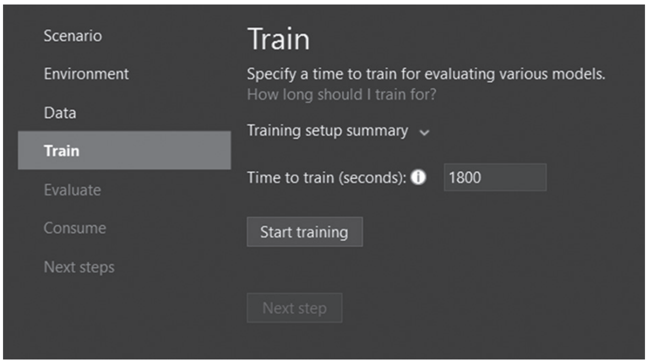


Fig. 8. Time to train estimated by ML.NET and Model Builder

Figure 9 shows the beginning of sampling training. Initially, the accuracy was unsatisfactory (21%), as the most optimized model for training had not yet been selected.

Figure 10 shows the optimal results and model in the selection process.

The accuracy of 91.8% is the value that satisfies the condition of "sufficient accuracy", which was set at the beginning.

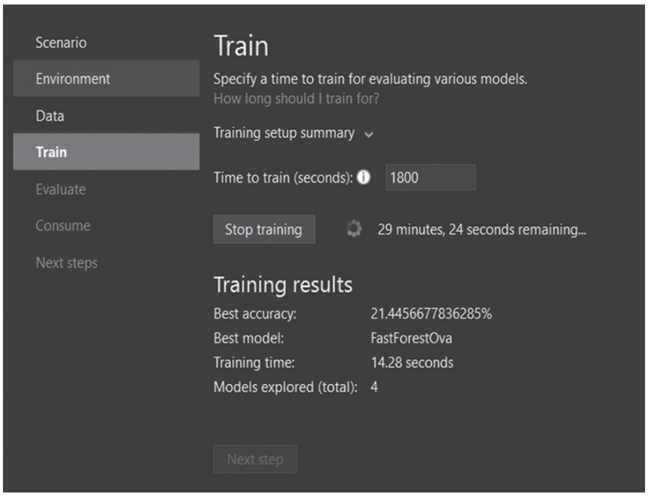


Fig. 9. Non-optimized model and its respective accuracy value

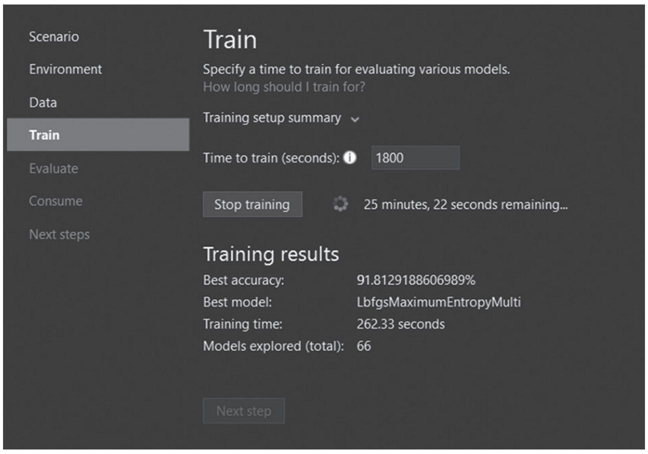


Fig. 10. Non-optimized model and its respective accuracy value

The release of ML.NET and Model Builder released a trained model that provides basic interfaces for testing and initial verification. This principle of learning became the basis for the creation of a neural network based on the principle of Multiclass Classification. In theory, the same approach could work with Multilabel Classification, but at the time of writing, this neural approach was not supported by the ML.NET framework.

8. Conclusions

The paper considers the use of neural network approach for the automatic selection of a complex of rehabilitation exercises during injuries, considering the state of the human cardiovascular system through the use of neural networks. A study is presented to investigate scenarios for applying multiclass and multilabel classification methods to select the most appropriate exercises for each particular case.

It can be concluded that both of these algorithms are appropriate to resolve the issue although each method has its own advantages and disadvantages. It is important to notice that the multiclass classification is considerably simpler and it has a wider list of programming languages and frameworks which natively support the method. On another side, the multilabel classification can solve more complex problems and can be an appropriate solution when there is a need to select a complex entity which cannot be retrieved by using that the multiclass classification algorithm [10]. In the research, the model built by ML.NET framework was used. It contained 10 properties called features; they were used to describe the injury, and one single property called labels which were used as a respective exercise to heal the injury.

As a point for further research, it is proposed to build the complex multilabel classification model using the same framework. Since ML.NET does not offer any native support for such method, it can be taken as a goal to solve this problem and compare results with the native multiclass classification ones.

Conflict of Interest

The authors declare no conflict of interest.

References

[1] Walter R. F. Rehabilitation of Sports Injuries // R. Frontera Walter., 2002.

[2] Physical Therapy Exerises // abayapps. 2019. –https://play.google.com/store/apps/details?id=com.rehabilitation.physicaltherapyexercises.

[3] Injurymap - Effective exercise therapy // Injurymap, 2020. – https://play.google.com/store/apps/details?id=com.injurymap.injurymap.

[4] Knee Pain Relieving Exercises // Dr.Kavin Khatri, 2016. –https://play.google.com/store/apps/details?id=com.technoherpes.kneepain.

[5] Dybowski R. Clinical Applications of Artificial Neural Networks // R. Dybowski., 2001.

[6] Kolo B. Binary and Multiclass Classification // B. Kolo. – Weatherford: Weatherford Press, 2010.

[7] Walker H. K. Clinical Methods: The History, Physical, and Laboratory Examinations // H. K. Walker, W. D. Hall, J. W. Hurst., 1990.

[8] Multilabel Classification: Problem Analysis, Metrics and Techniques // F.Herrera, F. Charte, A. J. Rivera, M. J. del Jesus. – New York: Springer, 2016.

[9] Tutorial: Classify the severity of restaurant health violations with Model Builder // Microsoft, 2019. – https://docs.microsoft.com/en-us/dotnet/machine-learning/tutorials/health-violation-classification-model-builder.

[10] Shubham J. Solving Multi-Label Classification problems (Case studies included) // Jain Shubham, 2017. –https://www.analyticsvidhya.com/blog/2017/08/introduction-to-multi-label-classification.

The article was delivered to editorial stuff on the 18.01.2021