MDPI

Article

Recommendation of Medical Exams to Support Clinical Diagnose Based on Patient's Symptoms

Hugo Cunha [1,†], Maria Pires [2,†], and Susana Marques [3,†]

[1] a84656@alunos.uminho.pt

[2] a86268@alunos.uminho.pt

[3] a84165@alunos.uminho.pt

† These authors contributed equally to this work.

Abstract: Nowadays, it is essential that the error in the decisions made by health professionals is as small as possible. This applies to any area, including the recommendation of medical exams through the diagnosis of diseases based on certain symptoms. This article's main goal is to be able to take advantage of Machine Learning techniques to improve the confidence of the medical exams prescribed by professionals and reduce the probability of mistakes, particularly having a lower probability of not recommending an exam they should have to recommend. Thus, in this project, six Machine Learning models were applied and optimized, Random Forest, Decision Tree, K-Nearest, Neighbours, Naive Bayes, SVM and RNN, in order to choose the best model for the problem at hand. From these six models and after the optimizations and the choice of the most correct accuracy metrics for this case, we can conclude that the best models were Random Forest, Decision Tree and KNN. The results are quite promising achieving high accuracies.

Keywords: Recommender System; Medical Exams; Diagnosis; CRISP-DM; Multi-class Classification; MCDTs;

1. Introduction

The decisions made by health care professionals in a clinical diagnosis have direct impact on their patients' treatment outcome. Due to the accelerated medical and technological growth, new options appear regularly, resulting in difficulties in choosing the most appropriate exams for patients [1]. Thus, the need to create recommender systems in order to assist professionals in the decision-making process becomes evident. It should be noted that in a system of this nature, in addition to the need to be fed by real and reliable data, because in a medical scenario a program that does not have high quality cannot be applied, the final product must only be used by a professional. Only with the combination of human experience and the efficiency provided by the program it is possible to achieve superior care.

The field of medicine is possibly the one that presents the greatest challenges in the integration of machine learning techniques. Starting from the basis of learning, these challenges lie in the nonexistence of datasets and in the difficulty of creating them due to inherent privacy issues. Clinical data are of a highly complex nature and are regularly incomplete and coming from various sources so they do not follow the same standards and records for the same problem may be incompatible [2]. Additionally, there is a wide range of end-users with specific and diverging needs which increases the complexity of producing accurate results. That said, currently, recommender systems in a medical context are an area that is just beginning to be developed.

In a generic way, a recommender system can be defined as a system that guides users in a personalized way to interesting or useful objects in a large space of possible objects or produces such objects as outputs [3]. In a medical perspective these objects can be the medical examinations that the patients will have to undergo and the users the health care professionals who will have to prescribe them.

In this context, this project has emerged, consisting in the development and exploration of machine learning algorithms for decision support in recommender exams for patients according to their symptoms. It should be noted that the recommendation is based on their symptoms only and not on their diseases, making the problem at hand more difficult since the intended goal is creating algorithms that can make a sort of intermediate diagnosis, managing to map the symptoms to the necessary exams without the need to take the intermediate step which is to think about which diseases the patient may have.

2. Related Work

There are several articles, and sources on recommender systems in health care and on the respective best methods to map symptoms to diseases.

This article has a greater goal and intends to go further by mapping the symptoms directly to the intended medical exams, not requiring a diagnosis beforehand. In this particular area there is a greater scarcity in research and it is intended that the research carried out and exposed in this article can be an asset for future research work and a good basis for further advances.

Focusing first on general recommender systems in the health field, it is common knowledge that there are treatments for diseases that can be time consuming and a great monetary burden. To avoid this, there is a need for systems that can detect disease symptoms as quickly as possible and even help professionals make better choices when treating patients. Thus, recommender systems have already been proposed to predict risk factors (such as possible complications or future illnesses) that a certain patient with a chronic disease may face in the future [4,5]. In this particular systems it is applied *Collaborative Filtering*, which recommends items to a user based on the following idea: "If users shared the same interests in the past, then they would have similar tastes". In the context of these systems, this approach can be interpreted as follows: "Patients who share similar diseases and health status might face the same risk factors" [6]. In the same way, IBM's artificial intelligence machine Watson Health is already able to find a suitable treatment for patients based on other patients' outcome and evidence-based medicine. IBM claims that 81% of healthcare executives familiar with Watson Health agreed that it has a positive impact on their business [7]. This demonstrates that using technology and analytics has become increasingly important in healthcare.

The research and development of predictions in the domain of medical examinations, also called the MCDTs (Complementary Means of Diagnosis and Therapeutics) is still quite early and has not been extensively explored. We argue that this is due to the specifics of benchmarking criteria in medical scenarios and the enormous context complexity of the medical domain. Here both risk perceptions towards data security and privacy as well as trust in safe technical systems play a central role, particularly in the clinical context. These aspects dominate acceptance of such systems.

Through the Kaggle website, it was found research related to this topic with the same or similar dataset in the past few months. There are studies with models such as Support Vector Machine (SVM) using as input a patient's symptoms and location to predict the most likely disease that the patient may suffer from, with an accuracy of approximately 94.44% [8]. There is also a study with a Random Forest model around 100% on the precision and recall functions and on the accuracy for each disease predicted by the symptoms of the test dataset [9].

After analysing this research and results, it becomes clear that with the dataset in use it is possible to obtain very satisfactory results which leads to believe in equally satisfactory results in this project. Taking into account that this project is deeper and more robust as it brings to the equation the prediction of medical exams and not diseases, we can have a good perspective on how the results might be, but as it is something innovative, those conclusions cannot be drawn with great certainty in the diseases prediction results.

3. Methodology

The benchmarking process followed the CRISP-DM (Cross Industry Standard Process for Data Mining) Methodology, one of the most popular methodologies used in Machine Learning and Data Mining projects worldwide.

One of the main advantages of its application is that it allows the construction and implementation of a Machine Learning model that can be used in a real environment, helping support business decisions [10].

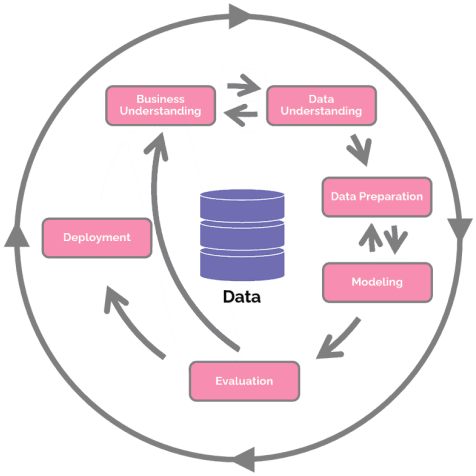

Figure 1. Stages of the CRISP-DM Methodology

As it can be seen from the previous image, the CRISP-DM methodology defines the project's life cycle, dividing it into six different stages:

- **Business Understanding:** In this first stage, the first thing to be done is to really understand what the problem is to be solved, looking for all the details about its impact on the field and what the objectives are in relation to the work.
- **Data Understanding:** This stage consists of organizing and documenting all the data that is available. This is where the data exploration work really begins, as it is necessary to be able to identify which data is important for solving the problem. At this point, the investigative side must step in so that the data reveals business problems, solutions and trends.
- **Data Preparation:** Now that the data has been identified, documented and analyzed, it's time to apply the technical analysis part of it. Now, the databases will be prepared and the formats and technical questions of the analysis will be defined. At this stage, the choice of data that will be worked on and how they will be crossed to solve the company's problem is made.
- **Modelling:** It is at this stage that the Machine Learning techniques are actually applied, based on the objectives identified in the first moment.
- **Evaluation:** Now actually comes what is done after the work itself. This is a very important moment, as it is when the results will be monitored in relation to the objectives and also the application of the knowledge obtained with the models. In this article there is a separate section just for this phase due to its importance.
- **Deployment:** The final phase of the methodology consists in the practical implementation of the resulting models, where we take the evaluation results and determine a strategy for their development [11]. This phase will no be taken in consideration in this article although in our project, a Platform (Web App) was created where health care professionals could eventually use a patient's symptoms as input and obtain the clinical exams that the patient should undergo as a result.

By applying this methodology, with data mining combined with predictive analytics, we gain great help in managing and controlling risks, whether internal or external. It is very important to remember that all planning and solution is based on consistent information and data, not just intuition. That way, it's quite possible to make smarter decisions.

After implementing the CRISP-DM cycle, we have real-time analysis as the situation and scenario change, enabling immediate and personalized changes for each moment. Agility in decision making and problem solving are, of course, important competitive advantages. Thus, the CRISP-DM methodology will certainly bring several solutions and also automate information, facilitating decision making and bringing a huge competitive advantage.

In the following subsections each phase of the CRISP-DM in relation to our project will be discussed in more detail.

3.1. Business Understanding

Interactive recommender systems aim to fulfil the evaluation criteria transparency. Providing recommendations on likely diagnoses of a patient to a doctor or nurse is an approach that can be very similar to case-based reasoning approaches. If instead of a diagnose we choose to provide medical exams to help the professional decision we have to understand that in particular in the area of health and medicine, the limiting resource is the (possibly) non-replenish-able health of the patient. The recommender system that we want should not only avoid failures and support decision making, but it should also understand the patient, the attitudes, the requirements, the values in the context of disease and health management. This makes the applicability of health recommender systems trickier [12].

Considering the complexity of medical data, as it is often unstructured, incomplete, non-standardized and stems from various sources or because large parts of data are not generated in a computer (as typical recommendations are) but stem from paper-based health-records that are often digitized afterwards it's a challenge to provide accurate medical recommendations, specially because datasets for health recommendations are rare as we've mentioned before.

At this stage of the project it is important to have a clear understanding of the main goals in order to ensure that the process is carried out rigorously and that an efficient recommendation system is therefore achieved. Hence, certain goals were established for this study: achieve a competitive advantage in the evaluation of medical exams by health professionals; be able to make personalized recommendations taking into account the type and number of symptoms; and find the recommendation algorithm and parameterization that leads to the highest overall performance in the recommendation system.

3.2. Data Understanding

The data used in this study comes from the Disease Symptom Prediction[13] dataset. The first dataset contains 4920 entries regarding 41 diseases and the symptoms experienced by different subjects suffering from that disease. There are 131 different symptoms in the dataset and for every disease there's exactly 120 entries with combinations of symptoms

experienced. The second dataset, associates to every symptom a severity weight on a scale from one to seven. Figure 2 shows the distribution of the severity of the diseases present in the dataset.

Due to the sensitivity of medical data and lack of datasets the mapping of the exams required to detect diseases had to be done manually through intensive research of reliable medical sources. Thus, a dataset mapping every disease, present in the first dataset, was created. The number of medical exams per diseases ranges from two to seven. Some diseases are linked to very specific sets of medical exams while others require nothing more than a simple physical examination or a blood test to be detected by the professional. 102 different medical exams were compiled, figure 3 displays the most common exams prescribed, where it's visible that prescribing blood tests and being physically examined by an health care professional is an important standard practice to understand and guide further diagnosis.

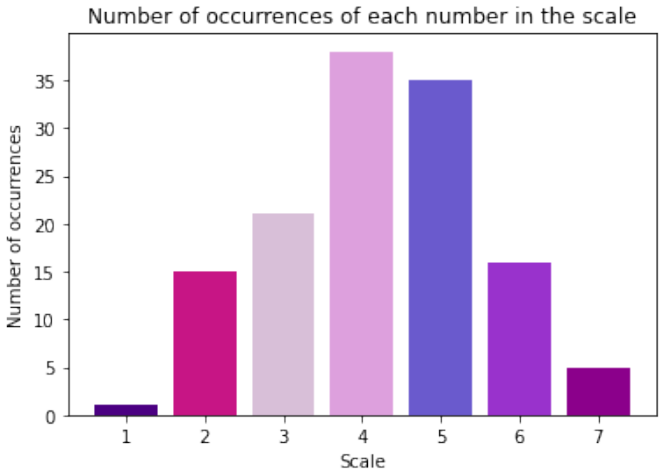

Figure 2. Distribution of the severity of the diseases

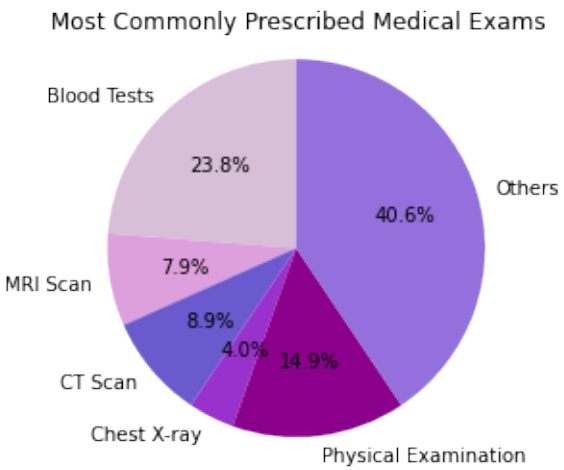

Figure 3. Most Prescribed Medical Exams

3.3. Data Preparation

Preparing the data is a crucial step to maximize the performance of the chosen models, therefore the datasets had to be merged into one large dataset and the raw data properly treated to be able to be fed to the models afterwards. The first step consisted of associating the symptoms to it's severity keeping the disease as the index. Then the exams were added according to the disease. Given that all the data, except the severity, is categorical, one-hot encoding was applied so that every symptom and every medical exam would be a column associated to a disease and it would be the value 0 or 1 in case that symptom or exam applied to the disease. After merging all the data, since we intend to abstract the disease, which is the common factor of both symptoms and medical exams, the same was eliminated. Finally, the dataset was split using 80% of the data to train the models and 20% to test them.

3.4. Modelling

In this stage we proceeded with the implementation of six Machine Learning models and the hyperparameters' optimization in order to determine which model performs best in the evaluation/results stage. An exhaustive search over specified parameter values was performed for the first five algorithms shown below.

Decision Tree Classifier

The goal of a Decision Tree is to create a training model that can predict the class or value of a target variable by learning simple decision rules inferred from the features of the training data. In Decision Trees, in order to predict a class label for a record we start from the rules on the root of the tree. We compare the values of the root attribute with the record's attribute, we follow the branch corresponding to that value and jump to the next node. For optimization purposes we used Gini's index and entropy as criteria to measure which features should be in the nodes. The actual feature for each node can be chosen randomly from a distribution based on the metric of each feature or just by choosing the one with the best value, the thresholds picked in each node are always the most optimal. There's no pruning applied to the tree however we tested limiting the tree growth to a max of 1, 10 or 20 nodes of depth as well as without a depth limit.

Random Forest Classifier

A Random Forest is a meta-estimator that uses the average of multiple decision tree classifiers, each decision tree can be

trained with a fraction of the dataset if the parameter bootstrap is True. As with the decision tree classifier the growth can be limited to 1, 10 or 20 nodes. The premise of this algorithm is that by using multiple tree classifiers with high randomness we can reduce the overall variance perceived in the resulting classification. Since this is a meta-estimator it's possible to define how many simple estimators to use, we evaluated the performance with 10, 20, 30, 50, 100, 200 and 1000 classifiers.

K-Nearest Neighbours

The K-Nearest Neighbors classifier is a type of instance-based learning as it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the K nearest neighbors of the point.

The numbers of neighbors we tested were 3, 5, 7, 11, 13, 15, 17 or 25. Using uniform weights, the value assigned to a query point is computed from a simple majority vote of the nearest neighbors however with weights based on distance, it assigns weights proportional to the inverse of the distance from the query point. The distance can be calculated using either the Euclidean distance formula or the Manhattan distance and the search algorithm can be by brute force or using a ball tree or KD-tree as an acceleration structure. [14]

Naive Bayes

Naive Bayes methods are a set of supervised learning algorithms based on applying Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable [15]. In each implementation of the Naive Bayes classifier the main difference is how to compute the likelihood of the features.

Using the Gaussian formula we can define the smoothing added to the variance as 1e-11, 1e-10 or 1e-9 of the maximum variance reported. Using the Bernoulli formula we can choose if the algorithm should learn the prior probabilities before training, we can also pick the additive smoothing parameter 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.9 or 1 which also works for the polynomial implementation of Naive Bayes algorithm.

Support Vector Machine

Support Vector Machines (SVMs) are a set of supervised learning methods [16] whose goal is to find a hyper-plane in an N-dimensional space, where N is the number of features, that distinctly classifies the data points. SVMs are a binary classifier therefore to be applied in this context there had to be generated N-binary classifier models. One-vs-Rest is an heuristic method that for each classifier, the class is fitted against all the other classes [17] achieving a multi-class classification. Three different kernels were tested, `poly, rbf, linear`, along with the optimization of the gamma value, which defines how far the influence of a single training example reaches, the c value, which is the regularization parameter controls the miss-classification, and the class weight that affects directly the c value.

Recurrent Neural Network

Neural networks, also known as artificial neural networks (ANNs) are a subset of machine learning and are at the heart of deep learning algorithms. Artificial neural networks are comprised of node layers, containing an input layer, one or more hidden layers, and an output layer. Each individual node is its own linear regression model, composed of input data, weights, a bias (or threshold), and an output. The output of any individual node is passed through a non-linear activation function before sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network. Neural networks rely on great amounts of training data to learn and improve their accuracy over time. [18] Neural networks generally perform supervised learning tasks, building knowledge from data sets where the right answer is provided in advance. The networks then learn by tuning themselves to find the right answer on their own, increasing the accuracy of their predictions.

To do this, the network compares initial outputs with a provided correct target. A cost function is used to modify initial outputs based on the degree to which they differed from the target values. Finally, cost function results are then pushed back across all neurons and connections to adjust the biases and weights in a technique called back-propagation, this is the key to how a neural network learns a particular task.[19]

Genetic Algorithm for Parameter Optimization

Neural networks and genetic algorithms demonstrate powerful problem solving ability. They are based on quite simple principles, but take advantage of their mathematical nature: non-linear iteration. Neural networks with back propagation learning showed results by searching for various kinds of functions. However, the choice of the basic parameters such as network topology, learning rate, initial weights often already determines the success of the training process. The selection of these parameters follow in practical use rules of thumb, but their value is at most arguable. Genetic algorithms

are global search methods, that are based on principles like selection, crossover and mutation. By combining genetic algorithms with neural networks (GANN), the genetic algorithm is used to find these parameters. A genetic algorithm tries to simulate the natural evolution process, its purpose is to optimize a set of parameters. Similarly to the original idea proposed by John Holland, the genetic information is encoded in an array of fixed length, called the individual. A possible value of each position on the array is called an allele. Each individual represents a possible solution to the examined problem. For the GANN problem, it contains information about the construction of a neural network. The quality of the solution is measured by its fitness value. [20]

In the scope of this study, the genetic algorithm implemented generates a population with a wide range of variety. The chromosomes that make up the population are generated randomly by a combination of eight different optimization functions, eight activation functions, two categorical loss functions, three different learning rates, four different values of dropout, last but not least, the architecture of the network may vary, the number of inside layers is chosen randomly ranging from one to five.

This algorithm run with an initial population of ten chromosomes, for ten generations. The five best individuals of each generation were chosen to create the next population with an additional mutation of only one gene per chromosome to assure diversity. Figure 4 shows the elements that constitute the first population.

```
Initial population:
Middle layers: 1 | Activation: softplus | Optimization: Adagrad | Loss: categorical_hinge | LR: 1e-05 | Dropout: 0.5 |
Middle layers: 4 | Activation: softmax | Optimization: Ftrl | Loss: categorical_hinge | LR: 0.0001 | Dropout: 0.15 |
Middle layers: 2 | Activation: relu | Optimization: Adamax | Loss: categorical_hinge | LR: 0.001 | Dropout: 0.5 |
Middle layers: 2 | Activation: softmax | Optimization: Adagrad | Loss: categorical_hinge | LR: 0.001 | Dropout: 0.3 |
Middle layers: 2 | Activation: softsign | Optimization: RMSprop | Loss: categorical_hinge | LR: 0.001 | Dropout: 0.25 |
Middle layers: 4 | Activation: softmax | Optimization: RMSprop | Loss: categorical_hinge | LR: 1e-05 | Dropout: 0.25 |
Middle layers: 5 | Activation: elu | Optimization: RMSprop | Loss: categorical_crossentropy | LR: 0.0001 | Dropout: 0.25 |
Middle layers: 4 | Activation: relu | Optimization: Ftrl | Loss: categorical_crossentropy | LR: 0.001 | Dropout: 0.5 |
Middle layers: 1 | Activation: elu | Optimization: Adamax | Loss: categorical_crossentropy | LR: 0.001 | Dropout: 0.25 |
Middle layers: 1 | Activation: relu | Optimization: Adam | Loss: categorical_crossentropy | LR: 0.001 | Dropout: 0.25 |
```

Figure 4. Genetic Algorithm's randomly generated initial population

The fitness score created to chose the five best parents was based on the accuracy calculated by sci-kit's learn accuracy metric and our own accuracy metric that will be explained in the following section.

Accuracy Metrics

The goal of this study is to recommend a list of possible medical exams for a set of symptoms, however the output of the neural network isn't a list of medical exams, most of the time the output of the network doesn't converge to a binary list, the results have to be processed so that we can evaluate them. Starting by obtaining an array of booleans with the one hot encoding standard, we apply a form of thresholding made by averaging the maximum and minimum values in the resulting array.

Now we can use any kind of metric, however when we're talking about a medical recommender system we must pick the metrics that are truly useful and return relevant information. Here we discussed the difference between the impact a false positive and a false negative so we thought it would be interesting to create a new metric which shows simple statistics about the correctness of each answer. The Real Accuracy metric represents the percentage of completely correct predictions done in a test dataset. In testing each prediction is matched with the ground truth and is assigned the ratio of correctly predicted tests, we can use this array of fidelity values to decide if this distribution fits well enough. Our main objective is to get 100% of precision on every prediction however a 99% correct answer is considered extremely precise, under 95% the error starts getting overwhelming thus we shouldn't have the average of the answers rounding this value, results under 95% correctness should only be accepted in a context of watching the evolution of the model as it is expected for the distribution of the results to shift from average 70% correct matches to over 95%.

We must keep in mind that our list of possible exams isn't much over 100 meaning that a 90% correctness implies that the model recommended incorrectly over 10 exams which might be catastrophic in a clinical setting depending on the patient's case and medical professional's interpretation.

4. Results and Discussion

Keeping in mind what was discussed in the previous section it is not surprising that the values present in table 1 are this high. The results compiled on said table refer to the algorithm with the best parameters found by performing grid search. We can conclude that K-Nearest Neighbours is the quickest, taking only 3 seconds per fold and reaching 100% for both accuracy metrics. SVM is the second quickest with only 4 seconds per fold. As for the Naive Bayes algorithm, the highest accuracy was obtained applying the bernoulli distribution, its real accuracy is only 82% which means that in a universe of 1170 sets of tests, 86 were classified incorrectly. Classifying a set incorrectly means not being able to

recommend the minimum required exams to detect the disease. The neural network's results are even lower and its training time is eighty times higher than KNN's execution time.

Table 1: Best Results obtained using Grid Search and Cross Validation

	Accuracy	**Real Accuracy**	**Execution Time (s)**
Random Forest	100%	100%	30
Decision Tree	100%	100%	50
K-Nearest Neighbours	100%	100%	15
Naive Bayes (Bernoulli)	94%	82%	26
SVM	100%	100%	20
RNN (Optimal)	73%	62%	1200

4.1. Genetic Algorithm

The score of the genetic algorithm is obtained using the following formula:

$$score = accuracy - (accuracy \times real\,Accuracy)$$

where *accuracy* is the value calculated by sci-kit learn's accuracy metric and *real Accuracy* is the metric developed for this study, as mentioned before. In this formula the real accuracy is divided by 1000 before applying it so that it's weight creates a fair score according to the number of exams that must be prescribed in order to provide a decent diagnosis.

The best hyperparameters found when applying the genetic algorithm were: two middle layers, `selu` as the activation function, `SGD` for the optimization function, `categorical_hinge` as the loss function, a learning rate of 0.001, a dropout of 0.15 which achieved a score of 0.71. Figure 5 describes the evolution of the accuracy of the models though the generations. We argue that the small improvement experienced by tuning such a wide range of parameters is due to the size and lack of variety of the dataset. In theory far better results were expected.

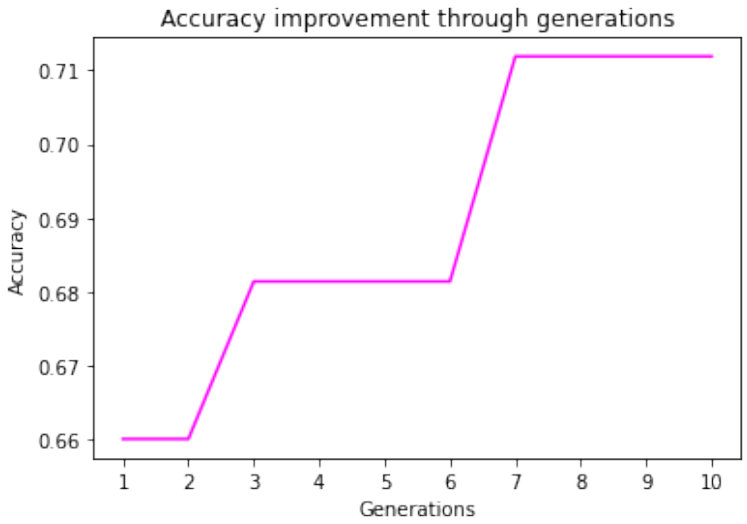

Figure 5. Accuracy improvement of the population through generations

Figure 6 shows the improvement of the optimal model's accuracy and the decrease of its loss. The state of convergence is achieved early in the process. There is a clear overfit of the model, not only due to the lack of diversity in the data but also because the model behaves in a way that it ensures it recommends a very generic and somewhat large set with very small variations in the recommendations. While this might seem problematic, exams such as blood tests and a physical examination are the basic diagnosis for several diseases.

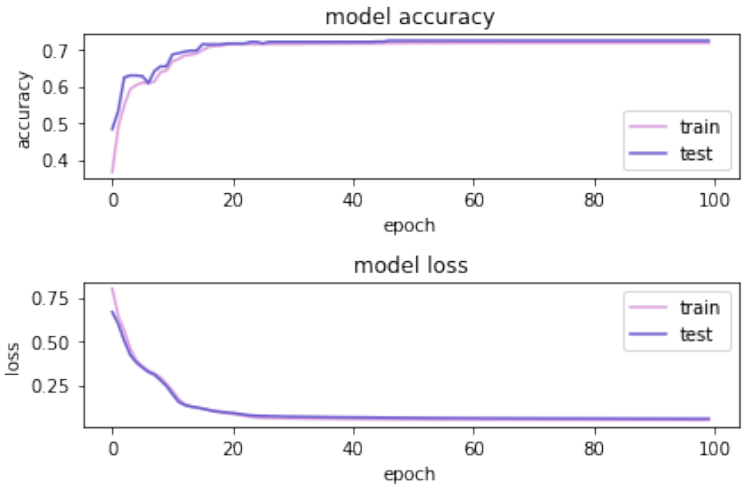

Figure 6. Model accuracy and model loss of the optimal recurrent neural network

5. Conclusion

As most healthcare professionals nowadays belong to a group that is characterized by the growth of the internet and technology, recommender systems require constant innovation and improvement. Although these systems in medicine have already a decent evolution in today's world, still much more is desired. There is a huge demand for technological assistance for healthcare professionals so that they can base their decisions in the most accurate way and with the lowest possible error rate.

This study focused mostly on efficient ways to implement the proposed recommender system and to fulfill all the main goals in terms of achieving a competitive advantage with the research carried out, finding the best algorithm for the proposed problem and to serve as a launching point for future research in the field of health care, particularly, in medical exams. We found that the use of models like Random Forest, Decision Tree and KNN has shown great potential in this case with an accuracy close to 100%. We argue that these results have to be understood carefully due to the shape and distribution of data and there should also be taking into account the way in which accuracy metrics are implemented in this situation.

In conclusion, this project has proved that it is possible to combine human expertise with computational power to improve health care and provide a thorough recommendation with all the required medical exams to detect a set of diseases that might be manifesting in the patient. In this particular case it is beneficial to recommend as many medical exams as necessary to understand what is causing the symptoms therefore the results are very satisfactory since the minimum of necessary tests are always recommended. Also, the accuracy metrics developed in this study ensures that it satisfies the concern stated before and explains the high results obtained. Ideally the models chosen to perform the benchmark on this study should be applied to a more diverse and realistic database, in this scenario it is expected that the recurrent neural network would perform better given the increase in the amount of data.

In the future, due to the scarcity of data in this area, we hope to find recommender systems built on the same model with larger datasets that include more qualitative data from more parts of the world to verify the generalization of the chosen model and to improve professional criticism and patient health.

References

[1] Tran, T.N.T., Felfernig, A., Trattner, C. et al. Recommender systems in the healthcare domain: State of the art and research issues. J Intell Inf Syst (2020).

[2] Valdez, A.C; Ziefle, M.; Verbert, K.; Holzinger, A. Recommender Systems for Health Informatics: State-of-the-Art and Future Perspectives.

[3] Adomavicius, G.; Tuzhilin, A. Toward the next generation of recommender systems: A survey of thestate-of-the-art and possible extensions.IEEE Trans. Knowl. Data Eng.2005,17, 734–749.

[4] Davis, D.A., Chawla, N.V., Christakis, N.A., Barab asi, A.L. (2009). Time to care: a collaborative engine forpractical disease prediction.Data Mining and Knowledge Discovery,20, 388–415.

[5] Nasiri, M., Minaei, B., Kiani, A. (2016). Dynamic recommendation: Disease prediction and prevention usingrecommender system. International Journal of Basic Science in Medicine, (pp. 13–17).

[6] Tran, T.N.T., Felfernig, A., Trattner, C. et al. Recommender systems in the healthcare domain: state-of-the-art and research issues. J Intell Inf Syst (2020).

[7] Stark, B., Knahl, C., Aydin, M., Elish, K., A Literature Review on Medicine Recommender Systems

[8] Disease Type Prediction using Symptoms. Available online: https://www.kaggle.com/naga26/disease-type-prediction-using-symptoms (accessed on 13 June 2021).

[9] Disease Prediction, Random Forest ,100% acc. Available online: https://www.kaggle.com/swapniljena/disease-prediction-random-forest-100-acc (accessed on 13 June 2021).

[10] Lopes, J., Ferreira, D., Neto, C., Abelha, A., Machado, J., Predicting the Survival of Primary Biliary Cirrhosis Patients.

[11] Azevedo AIRL, Santos MF (2008) KDD, SEMMA and CRISP-DM: a parallelloverview. IADS-DM

[12] Valdez, A. C. et al. "Recommender Systems for Health Informatics: State-of-the-Art and Future Perspectives." Machine Learning for Health Informatics (2016).

[13] Kaggle. Available online: https://www.kaggle.com/itachi9604/disease-symptom-description-dataset (accessed on 2 March 2021).

[14] Sci-kit Learn Nearest Neighbours Documentation. Available online: https://scikit-learn.org/stable/modules/neighbors.html#neighbors (accessed on 18 March 2021).

[15] Sci-kit Learn Naive Bayes Documentation. Available online: https://scikit-learn.org/stable/modules/naive_bayes.html (accessed on 20 March 2021).

[16] Sci-kit Learn Support Vector Machines Documentation. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (accessed on 19 March 2021).

[17] Sci-kit Learn OneVSRestClassifier Documentation. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsRestClassifier.html (accessed on 19 March 2021).

[18] IBM Cloud Learn Hub - What are Neural Networks? Available online: https://www.ibm.com/cloud/learn/neural-networks (accessed on 14 June 2021).

[19] Skirpan, Michael, Fast Forward Labs - How do Neural Networks Learn? Available online: https://www.kdnuggets.com/2015/12/how-do-neural-networks-learn.html (accessed on 14 June 2021).

[20] Lee, S.; Kim, J.; Kang, H.; Kang, D.-Y.; Park, J. Genetic Algorithm Based Deep Learning Neural Network Structure and Hyperparameter Optimization. Appl. Sci. 2021, 11, 744. https://doi.org/10.3390/app11020744