Explaining Misclassification and Attacks in Deep Learning via Random Forests

Rami Haffar, Josep Domingo-Ferrer, and David Sánchez

Universitat Rovira i Virgili
Department of Computer Engineering and Mathematics
CYBERCAT-Center for Cybersecurity Research of Catalonia
UNESCO Chair in Data Privacy
Av. Països Catalans 26
43007 Tarragona, Catalonia
{rami.haffar,josep.domingo,david.sanchez}@urv.cat

Abstract. Artificial intelligence, and machine learning (ML) in particular, is being used for different purposes that are critical for human life. To avoid an algorithm-based authoritarian society, AI-based decisions should generate trust by being explainable. Explainability is not only a moral requirement, but also a legal requirement stated in the European General Data Protection Regulation (GDPR). Additionally, it is also beneficial for researchers and practitioners relying on AI methods, who need to know whether the decisions made by the algorithms they use are rational, lack bias and have not been subjected to learning attacks. To achieve AI explainability, it must be possible to derive explanations in a systematic and automatic way. A common approach is to use a simpler, more understandable decision algorithm to build a surrogate model of the unexplainable, a.k.a. black-box model (typically a deep learning algorithm). To this end, we should avoid surrogate models that are too large to be understood by humans. In this work we focus on explaining the behavior of black-box models by using random forests containing a fixed number of decision trees of limited depth as surrogates. In particular, our aim is to determine the causes underlying misclassification by the black box model. Our approach is to leverage partial decision trees in the forest to calculate the importance of the features involved in the wrong decisions. We achieve great accuracy in detecting and explaining misclassification by deep learning models constructed via federated learning that have suffered attacks.

Keywords: Explainability · Machine learning · Deep learning · Random Forest

1 Introduction

The last two decades have witnessed major progress in artificial intelligence (AI). Deep learning, based on deep neural networks, has ushered in a breakthrough in the capability of AI systems. This forward leap has been made possible by

the increase of computing power and the availability of big data, which together allow training deep learning models. However, not all big data sources are public-domain. In order to leverage private data for learning, in [7] a new concept called federated learning was proposed. In federated learning the model is learned in a decentralized way: the model manager proposes an initial global model; then each owner of private data computes an update of the proposed model after running it on her private data set; finally, the model manager improves the global model by modifying it based on the aggregation of the received updates. This process is iterated until the global model classifies well enough all private data sets.

Both centralized deep learning and federated deep learning models achieve very accurate results, but they are black boxes in the sense that it is infeasible for humans to understand how decisions are made. This lack of transparency is problematic for both the developers who train a black-box model and the citizens who are affected by its decisions:

- Developers would like to know how decisions are made by the black-box algorithm to make sure that the algorithm takes the relevant features into account during the training phase. It may happen that wrong features are used to make decisions, such as in the well-known example of [11] where the animal is classified as a wolf if the image has snow background and as a husky dog if the image has grass background, because in the training pictures all the wolves were displayed in a snowy landscape and the huskies were not.
- Citizens are affected by the growing number of automated decisions: credit granting, insurance premiums, medical diagnoses, granting of loans, etc. While transparency measures are being implemented by public administrations worldwide, there is a risk of automated decisions becoming a ubiquitous black box. To protect citizens, explainability requirements are starting to appear in legal regulations and ethics guidelines, such as: article 22 of the EU General Data Protection Regulation [16], which states the right of citizens to an explanation on automated decisions; the European Commission's Ethics Guidelines for Trustworthy AI [3], that insist on the organizations that make automated decisions to be prepared to explain them at the request of the affected citizens; the IEEE report on ethically aligned design for intelligent systems [12].

Since it is necessary to explain the black-box behavior, the generation of explanations must be automated. A usual strategy to generate explanations for decisions made by a black-box machine learning model, such as a deep learning model, is to build a surrogate model based on simpler and more expressive machine learning algorithms, like decision rules [11, 15], decision trees [13, 2], or linear models [14]. As pointed out in [4], those simpler algorithms are intrinsically understandable by humans as long as they are not too large. The surrogate model can be trained either on the same data set as the black-box model to be explained or on new data points classified by that same model. Global surrogate models explain decisions on points in the whole domain, while local surrogate models build explanations that are relevant for a single point or a small region of the domain.

Contribution and plan of this paper

We present an approach which assumes that the party who generates the explanations has unrestricted access to the black-box model and a training data set. The training data set of the surrogate model can be smaller than the entire data used to train the black-box model: a sufficiently representative subset may be enough.

In this work, we use random decision forests to build our surrogate model. Random forests [5] consist of a fixed number of decision trees, each of which has a controlled depth and a measure reflecting the feature importance. The surrogate random forest will be trained on the same data that was used to train the black-box model.

Single decision trees have already been employed in the literature as surrogate models [2, 13]. The originality of our proposal lies in using a random decision forest rather than a single decision tree. In this forest, all trees have limited depth, but the structure of each tree may differ. The diversity in the random decision forest makes it possible to have trees whose predictions match the prediction of the black-box model even if the majority of the trees do not match it. This allows scanning the forest for trees that agree with the black-box model and using these trees for the explanation of the black-box decision.

In our work we concentrate on the cases where the black-box model made wrong predictions and our aim is to identify the features involved in those predictions. For example, if some features of the training data set were altered as a result of an attack, we can identify which of the altered features were more influential in the wrong predictions.

Section 2 identifies the desirable properties and the possible risks of explanations via surrogate models. Section 3 deals with outliers and possible attacks on machine learning data sets. Section 4 describes our surrogate model based on a random decision forest. Experimental results are provided in Section 5. Finally, in 6 we gather conclusions and sketch future research lines.

2 Requirements and risks of surrogate models for explanation

As discussed in [8], explanations of the decisions or predictions of a black-box model should satisfy the following properties:

- **Accuracy:** This property refers to how well an explanatory surrogate model predicts unseen data.
- **Fidelity:** The decisions of the explanatory surrogate model should be close to the decisions of the black-box model on unseen data. If the black-box model has high accuracy and the explanation has high fidelity, then the explanatory surrogate model also has high accuracy.
- **Consistency:** The explanations should apply equally well to any machine learning algorithm trained on the same data set.

- **Stability:** Decisions by the black-box model on similar instances should yield similar explanations.
- **Representativeness:** If the surrogate model can be applied to several decisions on several instances we can call it a highly representative explanation.
- **Certainty:** If the black-box model at study provides a measure of assurance in its predictions, an explanation of these predictions should reflect this measure.
- **Novelty:** It indicates the ability of the explanations of the surrogate model to cover cases far from the training data.
- **Degree of importance:** The explanation should highlight the important features.
- **Comprehensibility:** The explanations provided by the surrogate model should be understandable to humans. Depending on the target users, more or less complex explanations can be acceptable, but in general short explanations are more comprehensible.

No single explanation model in the current literature is able to satisfy all the above properties. In our approach we will focus on accuracy, fidelity, degree of importance, comprehensibility, and representativeness.

It must be realized that a complex surrogate model, even if based on intrinsically understandable algorithms, may fail to be comprehensible to humans. For example, we trained a decision tree on one of the studied data sets (Synthetic data set) that contains 10 feature attributes and a class attribute with two classes. The explanation tree contained 13,404 nodes and had depth 15, which does not make this tree very useful as an explanation to humans: it is very difficult to comprehend it.

3 Outliers and attacks on data sets

Machine learning models rely on the credibility of the data set used for training. However, data sets may contain some abnormal instances, also known as outliers. Although outliers can be legitimate, they can also be the outcome of attacks. On the one hand, cheating respondents may deliberately contribute bad answers when the data set is gathered. On the other hand, third-party attackers may seek to modify the training data set with a view to disrupt the training of the model.

Random attacks are the most common type of attack. In this case, the attacker intends to sabotage the machine learning model by replacing the actual values of one or more of the attributes of the training data with random values in the same range.

4 Random forest-based surrogate model

To satisfy the properties listed in Section 2, we need a method to construct surrogate models that keeps complexity at bay while being able to capture as

much as possible of the performance of black-box deep learning models. For that purpose, we use a random forest model. When trained on a fraction or the whole training data set, the random forest model outputs a vector of numerical values assessing the importance of the different features on classification decisions.

More specifically, feature importances are computed as follows [9]. First, let us define the notion of impurity of a data set, which is a measure of the homogeneity of the values in it. Impurity can be measured in several values, including Gini impurity and Shannon entropy. In particular, the Gini impurity of a data set is

$$C = \sum_{i=1}^{L} f_i(1 - f_i),$$

where L is the number of possible different labels and f_i is the relative frequency of values with the i-th label. Clearly, if all the values in the data set correspond to the same label, then $C = 0$; the more diverse the values, the higher C. Now, given a decision tree, the importance of each node j in it is

$$ni_j = w_j C_j - \sum_{k \in \text{Children}_j} w_k C_k,$$

where w_j is the weighted number of samples reaching node j, C_j is the Gini impurity of the samples reaching node j, and Children_j are the children nodes of node j. Thus, the higher the homogeneity gain of a node, the more important it is, where the homogeneity gain is the reduction of impurity between the input set of the node and its output subsets (those that go to its children nodes). In other words, an important node is one that "neatly" classifies the samples that reach it. Then, the raw importance of each feature i is

$$fi_i = \frac{\sum_{j:\text{node } j \text{ splits on feature } i} ni_j}{\sum_{k \in \text{all nodes}} ni_k}.$$

Finally, the normalized feature importance of each feature i is a number between 0 and 1 computed as

$$normfi_i = \frac{fi_i}{\sum_{j \in \text{all features}} fi_j}.$$

From now on, when we mention feature importances we will refer to normalized feature importances.

Algorithm 1 attempts to determine the causes of wrong prediction by the black-box model. First, the algorithm uses the training data set to train both the black-box model and the random forest model. Second, it evaluates both models to make sure that they can be compared with each other. The accuracy of the random forest should not be too inferior to that of the black-box model; otherwise, the explanations obtained from the random forest would be useless. Third, for each wrong prediction of the black-box model on the test data set, the algorithm scans the forest and stores the vector of feature importances of each

tree in the forest whose prediction coincides with the wrong black-box prediction. Finally, the algorithm averages all stored feature importance vectors, in order to obtain a vector containing the average importance of every feature in causing wrong decisions.

Algorithm 1: Determine the importance of features in wrong predictions

```
   Input: Dataset X
 1 Train_X, Test_X ← Split_Train_Test(X);
 2 Black_Box ← Train_Black_Box(Train_X);
 3 Forest ← Build_Random_Forest(Train_X);
   /* The accuracy of the black-box model and the random forest are
      evaluated                                                    */
 4 Score_Black_Box ← Evaluate_Black_Box(Test_X);
 5 Score_Forest ← Evaluate_Forest(Test_X);
 6 Feature_Importances_List ← {} ; // This list will contain the vectors of
   feature importances for selected trees in the forest
 7 for Each Sample in Test_X do
 8 |   if Predict (Black_Box,Sample) not correct then
 9 |   |   for Each Tree in Forest do
10 |   |   |   if Predict (Black_Box, Sample) == Predict (Tree, Sample) then
   |   |   |   |   /* The vector with the feature importances of each tree
   |   |   |   |      agreeing with the black-box model is appended to the
   |   |   |   |      list                                            */
11 |   |   |   |   Append (Feature_Importances_List, Tree.Feature_Importances);
12 |   |   |   end
13 |   |   end
14 |   end
15 end
   /* The vectors in the list are averaged to obtain the average
      feature importance vector                                    */
16 Feature_Importances←Average(Feature_Importances_List);
17 Return Black_Box, Score_Black_Box, Forest, Feature_Importances;
```

Algorithm 2 attempts to discover whether a black-box model was trained on an attacked data set and which are the training features that are most likely to have been attacked. The algorithm takes as input the outputs of Algorithm 1, that is, a trained black-box model, its reported accuracy score, a random forest explaining the black-box model and the vector of reported feature importances associated with the random forest. The algorithm also takes as input a *reliable* test data set Test_X that will be used to test whether the black-box model was trained on attacked data.

First, the algorithm evaluates the accuracy on the black-box model on the reliable test data Test_X. If the accuracy Score_Black_Box2 on Test_X is much lower than the reported accuracy Score_Black_Box, this suggests that the black-box model was trained on attacked data. In this case:

1. For each wrong prediction of the black-box model on the reliable test data Test_X, the algorithm scans the forest and stores the vector of feature importances of each tree in the forest whose prediction coincides with the wrong black-box prediction.
2. The algorithm averages all stored feature importance vectors, in order to obtain a vector Feature_Importances2 containing the average importance of every feature in causing wrong decisions.
3. The likelihood of each feature to have been attacked in the training data is proportional to the difference of the importance of that feature in Feature_Importances2 and the reported Feature_Importances. In particular, the feature with the largest difference is the most likely to have been attacked.

Algorithm 2: Discover the feature under attack

```
   Input: Black_Box, Score_Black_Box, Forest, Feature_Importances, Test_X
 1 Score_Black_Box2 ← Evaluate_Black_Box(Test_X);
 2 if Score_Black_Box − Score_Black_Box2 > threshold then
 3 |   Feature_Importance_List2 ← {};
 4 |   for Each Sample in Test_X do
 5 |   |   if Predict (Black_Box,Sample) not correct then
 6 |   |   |   for Each Tree in Forest do
 7 |   |   |   |   if Predict (Black_Box, Sample) == Predict (Tree, Sample) then
 8 |   |   |   |   |   Append
   |   |   |   |   |   (Feature_Importance_List2,Tree.Feature_Importances);
 9 |   |   |   |   end
10 |   |   |   end
11 |   |   end
12 |   end
13 |   Feature_Importance2←Average(Feature_Importance_List2);
14 |   Feature_Attack_Likelihoods = |Feature_Importance − Feature_Importance2|;
15 |   Return Feature_Attack_Likelihoods;
16 end
```

5 Experimental results

We have applied the above-described methodology to three data sets: a synthetic numerical data set, a real numerical data set and a real data set with a mix of categorical and numerical attributes.

To keep computation simple, we have made a small change when testing Algorithm 2. Instead of attacking the training data (which would require training the model first with the original training data and then with the attacked training data), we have attacked the test data Test_X used by Algorithm 2. This avoids re-training but has the same effect: the test data Test_X used in Algorithm 2

depart from the data used to train the model. We can take Test_X as being the good data and the training data as having been attacked.

5.1 Experiments on synthetic numerical data

We generated a data set consisting of $1,000,000$ records, each with 10 numeric continuous attributes and a single binary class labeled using the `make_classification` method from *Scikit-learn*[1]. We reserved 2/3 of the records to train the models, and the remaining 1/3 to test them. We took as a black-box model a neural network denoted by ANN with three hidden layers of 100 neurons each, which achieved 96.55 % classification accuracy. We also trained a random forest with 1000 trees, that had maximum depth 5 and an average size of 62 nodes. The forest classification accuracy was 90.8%. Table 1 shows the importances of features on the wrong decisions of the black-box model, as computed by Algorithm 1; features are sorted in descending order of importance. Feature number 5 turns out to be the one that has most influence on wrong decisions.

Table 1. Importance of the features of the synthetic data set on wrong predictions by the black-box model

Feature name	*Feature importance*
X[5]	0.2916
X[1]	0.2564
X[4]	0.0924
X[6]	0.0769
X[3]	0.0646
X[9]	0.0532
X[7]	0.0462
X[0]	0.0395
X[2]	0.0394
X[8]	0.0392

Also, we ran Algorithm 2 after attacking each feature individually with a threshold for the drop in the accuracy set to 4%. Our attacks were sufficient to exceed the threshold for all features. Algorithm 2 correctly detected the attacked feature in 60% of these single-feature attacks. Table 2 shows the result of attacking the three features with the highest importance (according to Table 1). It can be seen that the algorithm correctly detected attacks on the two most important features $X[5]$ and $X[1]$ (the only two whose importance is above 10%). However, the algorithm failed when the third most important feature $X[4]$ was attacked (it mistook it for the second most important feature $X[1]$). Table 3 shows the

[1] https://scikit-learn.org/stable/index.html

performance of Algorithm 2 when the three features with least importance were attacked. The algorithm detected well the attacks on the least important feature $X[8]$ and the third least important feature $X[0]$, but failed for the second least important feature $X[2]$ (which was mistaken for the second most important feature $X[1]$).

A comparison between Table 2 and Table 3 also suffices to see that the drop in the accuracy of the black-box model was greater when the high-importance features were under attack. Indeed, attacks on low-importance features entailed a milder degradation of accuracy. This also confirms that the feature importances identified by Algorithm 1 are coherent with the impact of those features on accuracy.

Table 2. Attacking the three most important features in the synthetic data set

Attacked feature	*Black-box accuracy*	*Feature detected by Algorithm 2*
X[5]	64.81%	X[5]
X[1]	64.9%	X[1]
X[4]	61.97%	X[1]

Table 3. Attacking the three least important features in the synthetic data set

Attacked feature	*Black-box accuracy*	*Feature detected by Algorithm 2*
X[8]	76.45%	X[8]
X[2]	73.38%	X[1]
X[0]	75.99%	X[0]

5.2 Experiments on real numerical data

To experiment with large real numerical data we used the "PAMAP2 Physical Activity Monitoring" data set from the UCI Machine Learning Repository[2]. This data set contains continuous measurements of 3 inertial body sensors (placed on the arm, chest, and ankle) and a heart-rate monitor worn by 9 subjects who performed 18 different activities such as walking, cycling, watching TV, etc. First, as recommended by the releasers of the activity data set [10], we created our data set by discarding the transient activity (e.g. going to the next activity location). Second, for simplicity, we mapped the various types of activity into two categories indicating whether the activity involved displacement or not (e.g. walking and cycling were mapped to "displacement" and watching TV to "not displacement"). As a result, we obtained a data set containing $1,942,872$ records

[2] https://archive.ics.uci.edu/ml/data sets/PAMAP2+Physical+Activity+Monitoring

of which 1,136,540 records were labeled as "displacement" and 806,332 as "not displacement".

Each record contained 54 numerical attributes corresponding to timestamp, label, heart rate, and 17 sensor data feeds for each of the 3 inertial sensors. Given an unlabeled record, the classification task in the experiment consisted in deciding whether the subject was performing at that instant an activity involving physical displacement. We used the same black-box model and random forest model as in the synthetic data set. In the classification of this data set the black-box achieved 99.9% accuracy and the forest 94.6% accuracy. Algorithm 1 computed the importances of the various features on wrong decisions of the black-box model: these importances ranged from 0.1616 down to 0.00017802.

We then applied Algorithm 2 with a threshold of 5% drop in the accuracy. Only 58.97% of the attacks on single features exceeded this threshold. Algorithm 2 correctly detected the attacked feature in 78.26% of these single-feature attacks. Table 4 shows that the attacks on the three most important features were detected correctly. Table 5 shows three examples of attacked features with low importance: two of those attacked features were correctly detected, but the algorithm was not able to detect the attack on acceleration_6_x_chest.

A comparison between Table 4 and Table 5 shows that accuracy drop was greater when attacking a feature with high importance than a feature with low importance. The exception was acceleration_6_x_chest which, in spite of being low-importance, caused a substantial accuracy drop and was in fact misdetected by Algorithm 2 as the high-importance feature acceleration_6_x_ankle. However, other than that our proposed method showed very promising results in detecting the importance of the features causing the black-box wrong predictions on a real numerical data and on detecting attacks on specific features.

Table 4. Attacking the three most important features in the physical activity data set

Attacked feature	*Black-box accuracy*	*Feature detected by Algorithm 2*
magnetometer_z_chest	82.66%	magnetometer_z_chest
gyroscope_z_ankle	76.23%	gyroscope_z_ankle
acceleration_6_x_ankle	86.25%	acceleration_6_x_ankle

Table 5. Attacking the three least important features in the physical activity data set

Attacked feature	*Black-box accuracy*	*Feature detected by Algorithm 2*
acceleration_6_x_hand	92.47%	acceleration_6_x_hand
acceleration_6_x_chest	86.73%	acceleration_16_x_ankle
gyroscope_x_ankle	90.68%	gyroscope_x_ankle

5.3 Experiments on real categorical data

To experiment on a real data set containing categorical data we used Adult, which is a standard data set hosted in the UCI Machine Learning Repository[3]. Adult contains 48,842 records of census income information and has 14 attributes reporting both numerical and categorical values. For each categorical attribute, we recoded categories as numbers to obtain a numerical version of the attribute. We reserved 2/3 of records to train the models, and the remaining 1/3 to validate them. We used the same black-box and forest as in the synthetic and physical activity data sets. The black-box achieved 84.53% classification accuracy and the forest achieved 84.42%. Table 6 lists the importance of features on the wrong decisions of the black-box model, as computed by Algorithm 1; features are sorted in descending order of importance.

Table 6. Importance of the features of the Adult data set on wrong predictions by the black-box model

Feature name	*Feature importance*
marital-status	0.2459
capital-gain	0.1967
relationsip	0.1902
educational-num	0.1645
age	0.012
hours-per-week	0.0451
capital-loss	0.0384
occupation	0.021
gender	0.0177
workclass	0.0035
native-country	0.00172
fnlwgt	0.00171
race	0.00094

We applied Algorithm 2 with a threshold of 3% in the accuracy drop. Only 53.84% of the attacks on single attributes exceeded this threshold. Among this 53.84%, Algorithm 2 correctly detected the attacked attribute in 85.71% of the cases. Table 7 shows the results for the three most important features: attacks on them were all correctly detected. Table 8 reports on three features with low importance: two were well detected but capital-loss was misdetected. Yet capital-loss was the only attribute in the data set whose attack exceeded the accuracy drop threshold but was not correctly detected by Algorithm 2.

A comparison between Table 7 and Table 8 also shows that the drop in the accuracy was greater when attacking a feature with high importance (according to Algorithm 1) than a feature with lower importance.

[3] https://archive.ics.uci.edu/ml/data sets/adult

Our approach holds promise, because: (i) in this real categorical data set only one single-feature attack was misdetected by Algorithm 2; (ii) the feature importances computed by Algorithm 1 anticipate the black-box model accuracy drop when each respective feature is attacked.

Table 7. Attacking the three most important features in the Adult data set

Attacked feature	*Black-box accuracy*	*Feature detected by Algorithm 2*
martial-status	80.96%	martial-status
capital-gain	63.62%	capital-gain
relationship	77.51%	relationship

Table 8. Attacking the three least important features in the Adult data set

Attacked feature	*Black-box accuracy*	*Feature detected by Algorithm 2*
hours-per-week	81.30%	hours-per-week
capital-loss	79.18%	capital-gain
educational-num	76.48%	educational-num

6 Conclusions and future research

We have presented an approach based on random decision forests with small tree depth that provide explanations on the decisions made by black-box machine learning models. Specifically, we have focused on investigating and explaining wrong decisions. Algorithm 1 computes the importance of the various features on the wrong black-box model decisions. Additionally, the visualization of the random forest trees affords further understanding of the decision making process. Algorithm 2 introduces a new way to protect against attacks altering the training data, because it allows detecting which features have been attacked.

As future work we intend to test the performance of our approach on other types of attacks such multi-feature random attacks and poisoning attacks. Also, we plan to satisfy more properties among those described in Section 2, in order to improve the quality of the explanations of the black-box model decisions.

Acknowledgments

Partial support to this work has been received from the European Commission (project H2020-871042 "SoBigData++"), the Government of Catalonia (ICREA Acadèmia Prize to J. Domingo-Ferrer and grant 2017 SGR 705), and from the Spanish Government (project RTI2018-095094-B-C21 "Consent" and TIN2016-80250-R "Sec-MCloud"). The authors are with the UNESCO Chair in Data

Privacy, but the views in this paper are their own and are not necessarily shared by UNESCO.

References

1. Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J.: Machine learning with adversaries: Byzantine tolerant gradient descent. In: Advances in Neural Information Processing Systems, pp. 119–129 (2017)
2. Blanco-Justicia, A., Domingo-Ferrer, J., Martínez, S., Sánchez, D.: Machine learning explainability via microaggregation and shallow decision trees. Knowledge-Based Systems, p. 105532 (2020)
3. European Commission High-Level Expert Group on Artificial Intelligence: Ethics Guidelines for Trustworthy AI (2019). https://ec.europa.eu/futurium/en/ai-alliance-consultation
4. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Computing Surveys **51**(5), 1–42 (2018)
5. Ho, T.K.: Random decision forests. In: Proceedings of the 3rd International Conference on Document Analysis and Recognition. IEEE (1995)
6. Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al.: Advances and open problems in federated learning. arXiv:1912.04977 (2019)
7. Konečný, J., McMahan, H.B., Yu, F.X., Richtárik, P., Suresh, A.T., Bacon, D.: Federated learning: Strategies for improving communication efficiency (v2). arXiv:1610.05492 (2017)
8. Molnar, C.: Interpretable Machine Learning (2019) https://christophm.github.io/interpretable-ml-book/
9. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
10. Reiss, A., Stricker, D.: Introducing a new benchmarked data set for activity monitoring. In: 16th International Symposium on Wearable Computers, pp. 108–109. IEEE (2012)
11. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should I trust you?" Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)
12. Shahriari, K., Shahriari, M.: IEEE standard review — Ethically aligned design: A vision for prioritizing human wellbeing with artificial intelligence and autonomous systems. In: 2017 IEEE Canada International Humanitarian Technology Conference (IHTC), pp. 197–201. IEEE (2017)
13. Singh, S., Ribeiro, M.T., Guestrin, C.: Programs as black-box explanations. arXiv:1611.07579 (2016)
14. Strumbelj, E., Kononenko, I.: An efficient explanation of individual classifications using game theory. Journal of Machine Learning Research **11**, 1–18 (2010)
15. Turner, R.: A model explanation system. In: IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1–6. IEEE (2016)
16. Voigt, P., Von dem Bussche, A.: The EU General Data Protection Regulation (GDPR): A Practical Guide, 1st Ed. Cham: Springer International Publishing (2017)