Identifying the Machine Learning Family from Black-Box Models[⋆]

Raül Fabra-Boluda✉, Cèsar Ferri, José Hernández-Orallo,
Fernando Martínez-Plumed, and M. José Ramírez-Quintana

DSIC, Universitat Politècnica de València, Spain
{rafabbo,cferri,jorallo,fmartinez,mramirez}@dsic.upv.es

Abstract. We address the novel question of determining which *kind* of machine learning model is behind the predictions when we interact with a black-box model. This may allow us to identify families of techniques whose models exhibit similar vulnerabilities and strengths. In our method, we first consider how an adversary can systematically query a given black-box model (oracle) to label an artificially-generated dataset. This labelled dataset is then used for training different surrogate models (each one trying to imitate the oracle's behaviour). The method has two different approaches. First, we assume that the family of the surrogate model that achieves the maximum Kappa metric against the oracle labels corresponds to the family of the oracle model. The other approach, based on machine learning, consists in learning a meta-model that is able to predict the model family of a new black-box model. We compare these two approaches experimentally, giving us insight about how explanatory and predictable our concept of family is.

Keywords: machine learning families · black-box model · dissimilarity measures. · adversarial machine learning

1 Introduction

Machine Learning (ML) is being increasingly used in confidential and security-sensitive applications deployed with publicly-accessible query interfaces, e.g., FICO or credit score models, health, car or life insurance application models, IoT Systems Security, medical diagnoses, facial recognition systems, etc. However, because of these public interfaces, an attacker can query the model with special

[⋆] This material is based upon work supported by the Air Force Office of Scientific Research under award number FA9550-17-1-0287, the EU (FEDER), and the Spanish MINECO under grant TIN 2015-69175-C4-1-R, the Generalitat Valenciana PROMETEOII/2015/013. F. Martínez-Plumed was also supported by INCIBE under grant INCIBEI-2015-27345 (Ayudas para la excelencia de los equipos de investigación avanzada en ciberseguridad). J. H-Orallo also received a Salvador de Madariaga grant (PRX17/00467) from the Spanish MECD for a research stay at the CFI, Cambridge, and a BEST grant (BEST/2017/045) from the GVA for another research stay at the CFI.

chosen inputs, get the results and learn how the model works from these input-output pairs –using ML techniques. This corresponds to the typical adversarial machine learning problem [5, 11, 15]. In an attack scenario, the attacker can take advantage of the knowledge of the type of learning technique (the ML family) the attacked model was derived from (and, in some cases, the true data distribution used to induce it) in order to explore intrinsic flaws and vulnerabilities. In this regard, several previous works have introduced specific strategies for attacking, extracting and stealing ML models of particular families such as *Support Vector Machines* [3], *(deep) Neural Networks* [18, 19], *Naive Bayes* [12], or even several online prediction APIs [22].

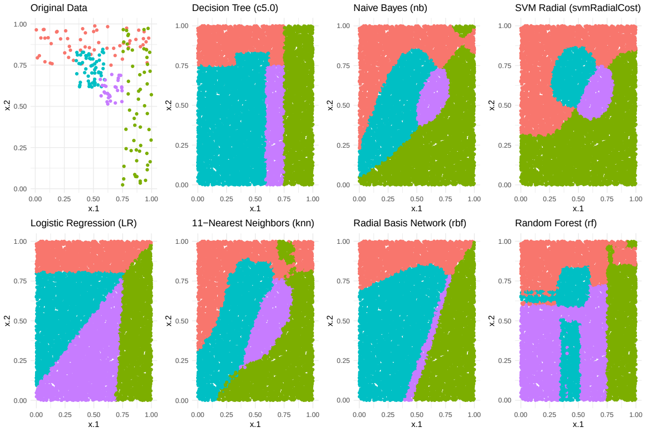

Fig. 1: Behaviour of different models learned over the same set of data (shown on the top-left plot). From left to right and from top to bottom: a decision tree, Naïve Bayes, a SVM, logistic regression, 11-Nearest Neighbour, Neural Network, and Random Forest.

One of the main reasons for not having *general* techniques for exploiting black-box models may be due the intrinsic differences between ML techniques: different models constructed using different ML techniques might disagree not only on decision boundaries but also on how they extrapolate on areas with little or no training examples. Figure 1 illustrates this, where, on the top-left plot we see the original train data of a bivariate dataset that we use to learn several ML models (using different techniques). What we observe is that all the models behave similarly on dense zones (where the training examples are originally located), but their behaviour on sparse areas (without training examples) is unpredictable and depends on the learning technique used. We may say that these less dense zones are those more likely to contain vulnerabilities. Hence, for many applications, some characteristics of the model are more relevant than the model itself (e.g., what topologies the decision boundaries have). In other words, for many attacks it is more important to know what the model looks like than its full semantics.

In this paper we address the problem of determining which *kind* of ML technique (family) has been used to construct a model that behaves as a black box, given a relevant subset of queries. This could be seen as an initial step for an adversarial learning procedure where, once we have obtained some knowledge about the ML family of the model to be attacked, it is possible to apply specific successful techniques, such as those mentioned above. Regarding the method proposed, and given that, other than its behaviour, we do not have access to any information about the black-box model (original data distribution nor the learning algorithm used to train it). We just use the black-box model as an *oracle* for labelling a synthetic dataset, generated by following a specific query strategy. This is then used to learn different models (using different ML techniques belonging to different learning families) trying to imitate the oracle behaviour. We denote these new models as *surrogate models*. We analyse the performance of these surrogate models in order to predict the original family of the target black-box model. Furthermore, a more elaborated technique is also proposed employing meta classifiers and dissimilarity measures between surrogate models.

The paper is organised as follows. Section 2 briefly outlines the most relevant related work. Section 3 addresses the problem of predicting the ML family given a black-box model. The experimental evaluation is included in Section 4. Finally, Section 5 closes with the conclusions and future work.

2 Related Work

Although there is an extensive literature on the topic of model extraction related to learning theory (such as the probably approximately correct (PAC) model of learning [23] and its query-based variants [1, 2]), as well as in the larger field of adversarial machine learning [5, 11, 15], in the vast majority of these approaches, the type of the model is assumed to be known [3, 12, 17–19, 22].

However, we also find different approaches focused on the replication of the functionality of models whose type may be unknown. A simple way to capture the semantics of a black-box ML model consists of mimicking it to obtain an equivalent one. This can be done by considering the model as an oracle, and querying it with new synthetic input examples (queries) that are then labelled by the oracle and used for learning a new declarative model (the *mimetic model*) that imitates the behaviour of the original one. Domingos et al. [7] addressed this problem by creating a comprehensible mimetic model (decision trees) from an ensemble method. Similar posterior proposals focused on generating comprehensible mimetic models that also exhibit a good performance. In this regard, Blanco-Vega et al. analysed the effect of the size of the artificial dataset in the quality of the replica and the effect of pruning the mimetic model (a decision tree) on its comprehensibility [4], developing also an MML-based strategy [24] to minimise the number of queries. Papernot et al. [17] also applied a mimicking strategy aiming at crafting examples that game a black-box model in order to obtain the desired output: the crafted examples that are able to cheat the replica are likely to cheat the original model, by the property of transferability that they studied in this same work.

Unlike the previous approaches, in this paper we consider that an attacker's goal is not to replicate or extract the ML model, but to obtain key actual model characteristics, such as the ML family. This would be, in some cases, more relevant than the model itself as it can be considered as a crucial first step before applying those more specific techniques or approaches aforementioned. Regarding our approach, such as in the mimetic approach, we also consider the black-box model as an oracle that is used to label a set of artificially-generated input examples (we also assume that the original data is not available). But instead of learning one mimetic model, we use the labelled data for learning several surrogates models using different learning techniques that will be prove successful for the task at hand.

3 Model Family Identification

One way of determining the *ML family* of a model that is to be attacked could be to mimick it and analyse their particular decision boundaries layouts. Therefore, given the extracted topologies the boundaries of a black-box model have, we will be able to identify the ML family that usually has that kind of boundary. In this section we describe two approaches. Both just interact with the black-box model by means of queries.

3.1 Learning Surrogate Models

As we already mentioned in the introduction, in order to generate surrogate datasets, we can query an oracle O (the black-box model to be attacked) with artificial examples so that O labels them (Figure 2 illustrates this). This allows us to build an artificial dataset labelled by O (what we call the *surrogate dataset* SD). As this dataset (the surrogate dataset SD) contains the output labels of O, it tends to capture the boundary patterns of O.

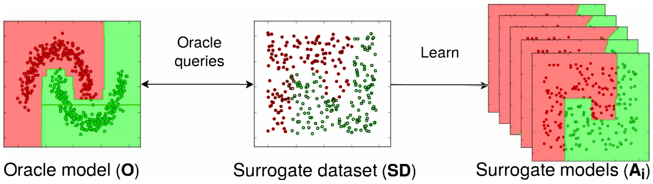

Fig. 2: Black-box models (oracles), trained over an unknown original dataset, are used to label synthetic surrogate datasets (generated following specific query strategies), which are then used to train surrogate models.

We can follow different strategies to query O. A basic (and effective) strategy consists in generating the artificial examples following a uniform distribution (SD, the surrogate dataset), which provides a good coverage of the feature space. We expect that such coverage includes non-dense areas where the behaviour of the induced black-box models is likely to be unexpected, possibly containing intrinsic vulnerabilities for specific techniques. Since SD is likely to capture the decision boundaries from O, we can thus learn other models from SD that mimic the behaviour of O. These surrogate models, denoted by A_i, $1 \leq i \leq N$, with N is the number of model families we consider in this work. In other words, we learn from SD a surrogate model A_i per model family $i \in N$. Each surrogate model A_i belongs to a different model family, then each A_i might provide a different characterisation of SD and, indirectly, a characterisation of O.

The most straightforward procedure to analyse how a model A_i behaves with respect to a dataset SD consists on evaluating this SD with A_i by cross-validation. As we want to use the set of surrogates models to identify the oracle's family, it makes sense to use the *Cohen's kappa* coefficient (κ) [14] as a dissimilarity metric to estimate the degree of inter-rate agreement for qualitative items. Unlike other evaluation measures, the kappa statistic is used not only to evaluate a single classifier, but also to evaluate classifiers between themselves (corrected by chance), and it is generally thought to be a more robust measure than a simple percentage of the agreement (see [10]). Actually, with kappa we can also take into account how different classifiers disagree on boundaries caused by extrapolation noise, as we have previously seen in Figure 1. In our case, we measure the agreement between the oracle (that has been used to label SD) and the surrogate model A_i when explaining the same dataset SD. We can evaluate the surrogate dataset SD (e.g., by using a train-test split or cross-validation) with the different surrogate models A_i, so that we would obtain a κ_i for each A_i. Figure 3 illustrates this procedure.

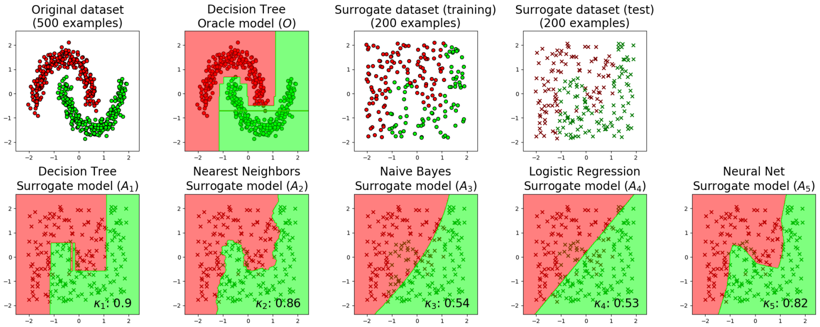

Fig. 3: Synthetic example to show the κ measure for different surrogate models A_i when compared to an specific oracle model O using a decision tree technique.

In the top of Figure 3 we see (from left to right) the original dataset, the oracle model O, the surrogate training set (SD) and the test set (these two sets labelled by O). The training set SD is used to learn all the surrogate models A_i shown in the bottom row. The test set is used to evaluate each of the A_i (i.e., to obtain each κ_i score). It is easy to see that the surrogate model with greatest resemblance to O is the decision tree (bottom-left plot), as confirmed by its kappa value ($\kappa_1 = 0.90$), although some other techniques are also doing a good job (e.g., nearest neighbours). The surrogate decision tree shares more commonalities with the oracle (both models have high expressive power and their boundaries are always parallel to the axis) than any other since the techniques used to induce them, although different, belong to the same family. Regarding the nearest neighbours-based model, although it also has a high expressive power, the boundaries are clearly not parallel to the axes (they present a sawtooth pattern which would be particularly difficult to achieve using decision trees). This is reflected by a lower κ value. Other surrogate models such as Logistic Regression or Naïve Bayes (with less expressive power) obtain much worse results, as we can see either visually or on the basis of the κ measure.

3.2 Model Family Identification

From the previous example (Figure 3) we saw that the family of the oracle model O was the one among the surrogate models A_i that achieved the highest κ. These suggests a straightforward procedure for family identification: we may expect that the family of O corresponds to the family of the A_i which provides the highest κ_i, since it implies the A_i achieves a high degree of agreement with O when explaining SD. Therefore, one first method we introduce consists in, given O, evaluating (using Cohen's Kappa) the generated SD with different surrogate models A_i (each A_i belonging to a different model family $i \in N$), and assigning the family of the A_i with highest κ value. However, this simple method might have its issues: it could be the case that other learning algorithms (different from the oracle's technique family), under certain parameters, obtain better κ using SD than a surrogate model based on the oracle family technique. This suggests that more complex methods should be tried to approach the problem more robustly, which takes us to our second approach.

The second approach consists in learning a meta-model for predicting the family of an oracle from a collection of meta-features (based on the κ value of the surrogate models) that abstractly describe the oracle. More concretely, instead of returning the model family of the surrogate model A_i with highest κ_i, we represent the oracle O by the kappa values of the surrogate models learned from SD. That is, given an oracle O belonging to a learning family $y \in N$, we represent it as the tuple $\langle \kappa_1(SD), \kappa_2(SD), \dots, \kappa_N(SD) \rangle$. For a given original dataset, we can learn as many oracles as families $y \in N$, each one represented by the tuple of κ values of its corresponding surrogate models, as follows:

$$\begin{gathered}\langle \kappa_1(SD_1), \kappa_2(SD_1), \dots, \kappa_N(SD_1) \rangle \\ \vdots \\ \langle \kappa_1(SD_N), \kappa_2(SD_N), \dots, \kappa_N(SD_N) \rangle\end{gathered}$$

If we apply this procedure for a set of D original datasets, we can construct a dataset collecting the tuples with the κ-based representation of the $D \times N$ oracles generated and adding the oracle family y to each tuple. This dataset can be used as the training set to learn a meta-model capable of predicting the model family y (the output) of a new incoming black-box model represented as a series of κ metrics (the input attributes of the meta-learning problem). This meta-model represents a similar approach to the top meta-model of Stacking [25], but in this case for the identification of families.

4 Evaluation

In this section we explain the experiments performed to validate our method for family identification[1]. All the experiments have been performed using `R` and, in particular, the package `Caret` to train the different ML models.

For the experiments, we have considered a number of ML techniques which have been widely used in practice, usually grouped in different families in terms of their learning strategy (see [8, 9, 16, 21]). In particular, we have considered the following $N = 11$ model families: *Discriminant Analysis* (DA), represented by RDA technique; *Ensembles* (EN), represented by Random Forest; *Decision Trees* (DT), represented by C5.0 algorithm; *Support Vector Machines* (SVM) with Radial Basis Function Kernel; *Neural Networks* (NNET), represented by MLP, *Naïve Bayes* (NB), *Nearest Neighbours* (NN), *Generalized Linear Models* (GLM), represented by the *glmnet* techinque; *Partial Least Squares and Regression* (PLSR), *Logistic and Multinomial Regression* (LMR) and *Multivariate Adaptive Regression Splines* (MARS). Model parameters were left to their default values.

In order to generate a dataset of meta-features that is large enough to evaluate our second approach, we employed $D = 25$ datasets (see Table 1).

Table 1: Characterisation of the 25 datasets from the UCI repository [6] used in the experiments: number of numerical (*#num*) and discrete (*#disc*) attributes, number of instances (*#inst*), and number of classes (*#class*).

	bank-market	*banknote-auth*	*breast-cancer*	*car*	*cmc*	*colic*	*credit-a*	*credit-g*	*diabetes*	*haberman*	*heart-c*	*heart-h*	*hepatitis*	*iris*	*kr-vs-kp*	*labor*	*lymph*	*monks-2*	*tae*	*tic-tac-toe*	*trains*	*vehicle*	*vote*	*waveform5000*	*zoo*
#num	10	4	-	-	2	7	6	7	8	2	6	6	6	4	-	8	3	-	3	-	-	18	-	40	1
#disc	10	-	9	6	7	15	9	13	-	1	7	7	13	-	36	8	15	6	2	9	32	-	16	-	16
#inst	4119	1372	286	1728	1473	368	690	1000	768	306	303	294	155	150	3196	57	148	601	151	958	10	846	435	5000	101
#class	2	2	2	4	3	2	2	2	2	2	2	2	2	3	2	2	4	2	3	2	2	4	2	3	7

For each dataset, we trained $N = 11$ models (oracles) belonging to the different families introduced above (we learned $D \times N = 25 \times 11 = 275$ oracle models). For each of these oracles, we generated a surrogate dataset SD that we use to learn and evaluate the surrogate models A_i, belonging to the same N model families. Every example of SD was generated at random following the uniform distribution: for each numerical feature, we randomly generated a number between its minimum and maximum values. For each discrete feature, we randomly picked one of the possible values it may hold. The number of examples per SD was 100 per number of dimensions. The evaluation of A_i w.r.t. SD was performed using 5-fold stratified cross-validation to obtain each of the κ_i values (as previously described). At this point, we have a collection of 275 oracles represented as a series of κ_i measures, whose model family is known. This represents a dataset of meta-features, along with the family as label. Once the dataset of

[1] For reproducibility and replicability purposes, all the experiments, code, data and plots can be found at https://github.com/rfabra/whos-behind

oracles is created, as explained in Section 3.2, we use a support vector machine algorithm to learn the meta-model, and we apply a leave-1-out evaluation, such that in each iteration the instances corresponding to one original dataset are used for test and the rest are used for training.

Table 2 (left) shows the confusion matrix for the experiments using the approach based solely on surrogate models and dissimilarity measures (kappa values). We observe that LMR, DT and MARS are the those with a higher positive identifications, with 22, 16 and 14 correct identifications respectively. Other cases perform very poorly, such as the DA, EN, SVM, NNET NB or PLSR families, with none or very few correct identifications. There are many cases in which a model is strongly confused with other models. For instance, the DA family tends to be confused with GLM (14 wrong identifications). SVM, NNET and GLM are often confused with LMR (12, 15 and 10 wrong identifications, respectively), NB gets confused with MARS (13 wrong identifications), and something similar happens with NN with SVM (11 wrong identifications). The overall accuracy has been 30%. All this suggests that the decision boundaries between families might not be so clear. In any case, as this is an 11-class classification problem, a random classification (baseline) would obtain an accuracy around 9%, so this straightforward approach improves the random baseline significantly and suggests that dissimilarity measures might be partially useful for the model family identification task.

Table 2: Confusion matrix (Real vs. Predicted Class) obtained when evaluating the approach based solely on surrogate models (Left, Acc: 30%) and the one based on meta-models (Right, Acc: 56%).

Family	DA	EN	DT	SVM	NNET	NB	NN	GLM	PLSR	LMR	MARS
DA	**1**	0	0	7	0	0	1	14	0	1	1
EN	0	**7**	2	4	0	0	1	1	0	6	4
DT	0	7	**16**	0	0	0	0	1	0	0	1
SVM	0	1	2	**5**	0	0	2	0	0	12	3
NNET	0	0	0	0	**2**	0	0	8	0	15	0
NB	2	1	1	0	2	**1**	0	4	0	1	13
NN	0	5	4	11	0	0	**4**	0	0	1	0
GLM	0	2	0	1	0	0	0	**11**	0	10	0
PLSR	1	0	0	0	2	0	0	18	**0**	3	1
LMR	0	0	0	0	0	0	0	2	0	**22**	1
MARS	0	2	3	1	0	0	0	0	0	4	**14**

Family	DA	EN	DT	SVM	NNET	NB	NN	GLM	PLSR	LMR	MARS
DA	**9**	1	1	0	1	2	0	0	10	0	1
EN	1	**12**	4	3	0	1	2	1	0	1	0
DT	0	1	**21**	0	0	1	0	0	0	0	2
SVM	1	5	1	**10**	0	0	4	1	0	3	0
NNET	0	0	0	0	**19**	0	0	5	1	0	0
NB	5	1	0	0	0	**13**	1	1	3	0	1
NN	1	2	1	1	0	0	**19**	0	0	0	1
GLM	0	1	0	0	10	1	0	**9**	1	1	1
PLSR	4	0	0	0	2	2	0	0	**15**	1	1
LMR	0	1	2	2	2	0	1	2	1	**13**	1
MARS	1	0	2	0	0	1	1	2	0	3	**14**

Table 2 (right) shows the confusion matrix obtained for the meta-model-based approach. In this case, we can observe a general improvement in the results with respect to the previous approach accuracy increases from 30% to 56%). Here, DT is the easiest family to identify, with 21 of 25 correct identifications. For other families, such as NNET or NN, the method also performs well, with 19 correct identifications. DA is mainly confused with PLSR, with 10 wrong identifications. Similarly, GLM is confused with NNET, also with 10 wrong identifications. From the results we can see GLM and DA are the hardest families to identify, since they are strongly confused with others. The rest of families (i.e., NB, LMR, PLSR or MARS) are correctly identified in most cases, showing no specific confusion patterns between them and other models.

In general terms, the results show that, although this is a particularly complex problem, the use of dissimilarity measures to differentiate ML families from one another seems an effective approach.

5 Conclusions and Future Work

In this work we addressed the problem of identifying the model family of a black-box learning model. We presented two approaches based on dissimilitary measures such as the Cohen's kappa coefficient. The first one consists in learning several surrogate models (from different learning families) from a set of artificial examples labelled by the black-box model (which acts as an oracle). Then we select the family of the surrogate model with the best kappa value as the family of the black-box model. The second approach consists in using the kappa values as meta-features for representing the black-box model, and learning a meta-model that, given a black box model, predicts its learning family.

The experiments show that the first proposed approach, although it performs poorly as a ML family identification method, it is able to improve significantly the accuracy we would obtain using a random baseline. The second approach based on the meta-model performs much more accurately than the first one, laying special emphasis on the potential of using meta-models trained with abstract meta-features (based on dissimilarities) characterising the oracles.

In order to improve the results of our approach based on the meta-model, we plan to investigate the use of other measures for model divergence and diversity [13, 20] as meta-features. Another aspect that could be of interest would be the use of other query strategies for generating the surrogate dataset. Finally, as our approaches for model identification rely on the kappa measure, we also intend to establish a new definition of model family by applying a hierarchical learning clustering algorithm using the kappa metric to group the learning techniques for conforming the different families.

References

1. Angluin, D.: Queries and concept learning. Machine learning **2**(4), 319–342 (1988)
2. Benedek, G.M., Itai, A.: Learnability with respect to fixed distributions. Theoretical Computer Science **86**(2), 377–389 (1991)
3. Biggio, B., Corona, I., Nelson, B., Rubinstein, B.I., Maiorca, D., Fumera, G., Giacinto, G., Roli, F.: Security evaluation of support vector machines in adversarial environments. In: Support Vector Machines Applications, pp. 105–153 (2014)
4. Blanco-Vega, R., Hernández-Orallo, J., Ramírez-Quintana, M.: Analysing the trade-off between comprehensibility and accuracy in mimetic models. In: Discovery Science. pp. 35–39 (2004)
5. Dalvi, N., Domingos, P., Sanghai, S., Verma, D., et al.: Adversarial classification. In: Proc. of the 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining. pp. 99–108. ACM (2004)
6. Dheeru, D., Karra Taniskidou, E.: UCI machine learning repository (2017), http://archive.ics.uci.edu/ml

7. Domingos, P.: Knowledge discovery via multiple models. Intelligent Data Analysis **2**(3), 187–202 (1998)
8. Duin, R.P., Loog, M., Pkalska, E., Tax, D.M.: Feature-based dissimilarity space classification. In: Recognizing Patterns in Signals, Speech, Images and Videos, pp. 46–55. Springer (2010)
9. Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems. J. Mach. Learn. Res **15**(1), 3133–3181 (2014)
10. Ferri, C., Hernández-Orallo, J., Modroiu, R.: An experimental comparison of performance measures for classification. Pattern Recognition Letters **30**(1), 27–38 (2009)
11. Giacinto, G., Perdisci, R., Del Rio, M., Roli, F.: Intrusion detection in computer networks by a modular ensemble of one-class classifiers. Information Fusion **9**(1), 69–82 (2008)
12. Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I., Tygar, J.: Adversarial machine learning. In: Proc. of the 4th ACM workshop on Security and artificial intelligence. pp. 43–58 (2011)
13. Kuncheva, L.I., Whitaker, C.J.: Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. Machine learning **51**(2), 181–207 (2003)
14. Landis, J.R., Koch, G.G.: An application of hierarchical kappa-type statistics in the assessment of majority agreement among multiple observers. Biometrics pp. 363–374 (1977)
15. Lowd, D., Meek, C.: Adversarial learning. In: Proc. of the 11th ACM SIGKDD Int. Conf. on Knowledge Discovery in Data mining. pp. 641–647. ACM (2005)
16. Martınez-Plumed, F., Prudêncio, R.B., Martınez-Usó, A., Hernández-Orallo, J.: Making sense of item response theory in machine learning. In: Proceedings of 22nd European Conference on Artificial Intelligence (ECAI), Frontiers in Artificial Intelligence and Applications. vol. 285, pp. 1140–1148 (2016)
17. Papernot, N., McDaniel, P., Goodfellow, I.: Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277 (2016)
18. Papernot, N., McDaniel, P., Jha, S., Fredrikson, M., Celik, Z.B., Swami, A.: The limitations of deep learning in adversarial settings. In: Security and Privacy (EuroS&P), 2016 IEEE European Symposium on. pp. 372–387. IEEE (2016)
19. Papernot, N., McDaniel, P., Wu, X., Jha, S., Swami, A.: Distillation as a defense to adversarial perturbations against deep neural networks. In: Security and Privacy (SP), 2016 IEEE Symposium on. pp. 582–597. IEEE (2016)
20. Sesmero, M.P., Ledezma, A.I., Sanchis, A.: Generating ensembles of heterogeneous classifiers using stacked generalization. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery **5**(1), 21–34 (2015)
21. Smith, M.R., Martinez, T., Giraud-Carrier, C.: An instance level analysis of data complexity. Machine learning **95**(2), 225–256 (2014)
22. Tramèr, F., Zhang, F., Juels, A., Reiter, M.K., Ristenpart, T.: Stealing machine learning models via prediction apis. In: USENIX Security Symp. pp. 601–618 (2016)
23. Valiant, L.G.: A theory of the learnable. Communications of the ACM **27**(11), 1134–1142 (1984)
24. Wallace, C.S., Boulton, D.M.: An information measure for classification. The Computer Journal **11**(2), 185–194 (1968)
25. Wolpert, D.H.: Stacked generalization. Neural networks **5**(2), 241–259 (1992)