Safety Evaluation of Machine Learning Classifiers through Statistical Difference Measures (Implementation in R)

Koorosh Aslansefat
UNIVERSITY OF HULL | 23/03/2020

Table of Contents

List of Figures

List of Tables

Abstract: Nowadays the application of advanced machine learning or deep learning in various industries is growing rapidly. In parallel with this massive growth, the concern about the safety and explainability of such an algorithm is also in high demand. This report is proposing a solution to do a safety verification through manipulating existing statistical difference measure algorithms such as Corn-off, Kolmogorov-Smirnov. A number of hypothetical examples are provided to clarify the limitations and capabilities of the proposed method.

Keyword: Safety, SafeML, SafeAI, Machine Learning, Deep Learning, Artificial Intelligence, Statistical Difference

I. Introduction

Artificial Intelligence (AI) is growing fast in many applications. In parallel with this rapid growth, the concern of the society regarding the reliability, and safety assurance of those algorithms is raising. For safety-critical applications where human life or privacy, environment and economy are the topics in which the existing concern would be more serious. For instance, AI in medicine by (Wiens et al., 2019), AI in autonomous systems like self-driving cars by (Burton et al., 2020; Du-Harpur et al., 2020), AI in Military (Sharkey, 2019), and AI in economic by (Davenport et al., 2019). In addition, different organizations and governmental institutes are trying to establish new rules, regulations and standards for AI (Evans, 2020; *ISO/IEC JTC 1/SC 42: Artificial intelligence*, 2020).

The artificial intelligence and specifically machine learning can have harmful behaviour and put our life, privacy, environment and money in risk. It can be accidentally because of poor design, specifying the wrong objective function, having implementation error, choosing the wrong learning process and using a poor or incomprehensive dataset for training. Thus, AI safety can be defined as a set of actions or endeavour to prevent any harm to humanity through AI. However, there are many perspectives and directions to be defined for AI Safety. In fact, (Amodei et al., 2016) have addressed different research problems of certifying the modern machine learning systems operating in the field. They have categorized safety issues into five categories including A) Safe exploration, B) Robustness to distributional shift, C) Avoiding negative side effects, D) Avoiding “reward hacking” and “wire heading”, E) Scalable oversight.

Statistical distance measure can be considered as a common method to measure distributional shift. Furthermore, in modern ML algorithms like Generative adversarial nets (GANs), different statistical distance or divergence is applied as a loss function such as Jensen-Shannon divergence (Goodfellow et al., 2014), Wasserstein distance (Gulrajani et al., 2017), and Cramer distance (Bellemare et al., 2017). For dimension reduction, t-SNE (t-distributed stochastic neighbour embedding) algorithm uses the Kullback-Leibler divergence as a loss function (Laurens van der Maaten, 2014). In this report, the main focus will be on safety evaluation based on the statistical distance which is related to the robustness and distributional shift aspect of AI safety. A comprehensive study on dataset shift has been provided by (Quiñonero-Candela & Schwaighofer, 2009) and the dataset issues such as projection and projectability, simple and prior probability shift have been discussed. However, the study did not address the use of statistical distance and error bound to evaluate the dataset shift.

A resampling uncertainty estimation (RUE)-based algorithm has been proposed by (Schulam & Saria, 2019) to ensure the pointwise reliability of the regression when the test or field data set is different from the training dataset. The algorithm has created predictions ensemble through the modified gradient and Hessian functions for ML-based regression problems. By investigating through existing literature, there are is no conducted research work so far in which safety and accuracy of the ML-based classification be addressed. In this paper, a modified version of statistical distance is used to compare the dataset during the training procedure and the field test. Then a novel human-in-loop procedure has been proposed to estimate and certify the accuracy of the system in different scenarios.

Different examples have been provided to show the capabilities and limitations of the proposed approach.

The rest of the report is organised as follows: In section II, the problem definition is provided. The proposed method is addressed in section III. Numerical results are demonstrated in section IV with a brief discussion. The capabilities and limitations of the proposed method are summarised in section V and the paper terminates with a conclusion.

II. Problem Definition

The classification algorithms have different applications. For instance, abnormality detection can be one of them. A simple classifier can be a line or threshold. Consider a hypothetical measurement (e.g. Temperature) as shown in Figure 1 and can be defined as (1).

$$D(t) = \begin{cases} Class\ 1 & 0 \leq t \leq 100 \\ Class\ 2 & 101 \leq t \leq 200 \end{cases} \tag{1}$$

The measurement $D(t)$ has included two classes: "Class 1" and "Class 2". For instance, they can represent the normal and abnormal state of a system. From time 0 to 100 is class 1 and from 101 to 200 is class 2.

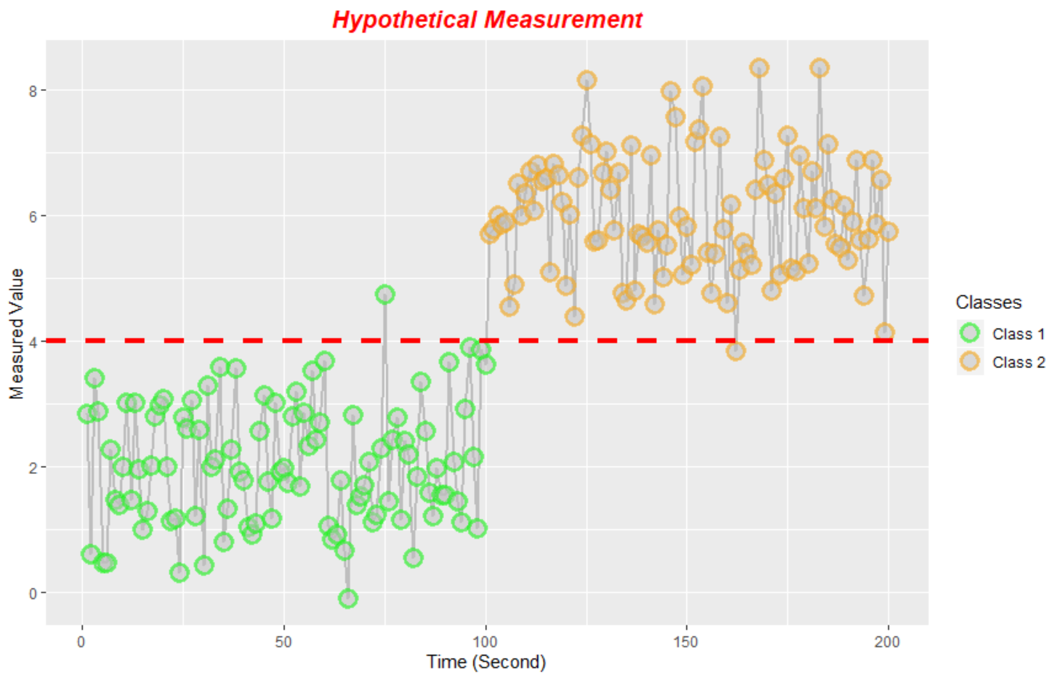


Figure 1. A hypothetical measurement (i.e. from 0 to 100 is Class 1 and from 101 to 200 is Class 2).

The probability density function of the measurement can be estimated as shown in Figure 2. In this figure, the threshold has been represented with a red vertical dash-line and value of four. The area with an overlap in this figure can cause missed and false detection (also called false positive/type I error and false-negative/type II error).

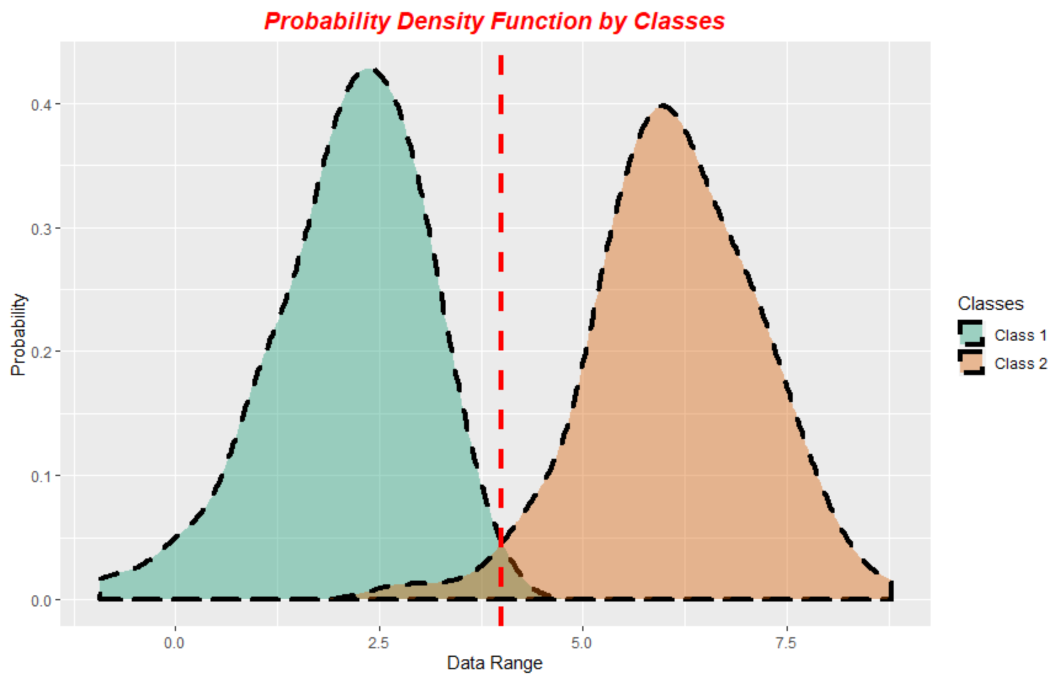


Figure 2. The estimated probability density function for both Class 1 and Class 2 with a threshold equal to four.

Looking at the figure of probability density functions, the area that two probability density functions are merging can cause the miss-classification and consequently the error. Probability of the error or miss-classification can be calculated using (2) (Theodoridis & Koutroumbas, 2009). It should be noted that the error probability is also related to the threshold (x considered as threshold value). For more details check (Aslansefat et al., 2019).

$$P(error) = \int_{-\infty}^{+\infty} P(error|x)P(x)dx \tag{2}$$

In which the $P(error|x)$ can be calculated through calculating the minimum of both probability density functions as (3). The minimization is subject to variation of threshold value from -inf to +inf.

$$P(error|x) = min[P(Class\ 1|x), P(Class\ 2|x)] \tag{3}$$

By dividing the space into two regions as R_1 and R_2, the probability of error can be written with parts.

$$\begin{aligned} P(error) &= P(x \in R_1, Class\ 1) + P(x \in R_2, Class\ 2) \\ &= \int_{R_1} P(x|Class\ 1)P(Class\ 1)\,dx + \int_{R_2} P(x|Class\ 2)P(Class\ 2)\,dx \end{aligned} \tag{4}$$

To ease the minimization problem, consider the following inequity (Fukunaga, 1992).

$$min[a,b] \leq a^{\lambda}b^{1-\lambda}\ where\ a,b\ \geq 0\ and\ 0 \leq \alpha \leq 1 \tag{5}$$

Equation (3) can be rewritten as (6).

$$\begin{aligned} P(error|x) &= min[P(Class\ 1|x), P(Class\ 2|x)] \\ &= min\left[\frac{P(x|Class\ 1)P(Class\ 1)}{P(x)}, \frac{P(x|Class\ 2)P(Class\ 2)}{P(x)}\right] \end{aligned} \tag{6}$$

Using the inequity rule and equation (6), the conditional probability of error can be derived as (7).

$$P(error|x) \leq \left(\frac{P(x|Class\ 1)P(Class\ 1)}{P(x)}\right)^{\lambda}\left(\frac{P(x|Class\ 2)P(Class\ 2)}{P(x)}\right)^{1-\lambda} \tag{7}$$

The equation (8) can be obtained using equations (2) and (7).

$$P(error) \leq \left(P(Class\ 1)\right)^{\lambda}\left(P(Class\ 2)\right)^{1-\lambda} \int_{-\infty}^{+\infty} \left(P(x|Class\ 1)\right)^{\lambda}\left(P(x|Class\ 2)\right)^{1-\lambda} dx \tag{8}$$

It safety assurance, it is so important to consider the worth case scenario which can lead us to (9) known as Chernoff upper bound of error (Fukunaga, 1992).

$$P(error) = \left(P(Class\ 1)\right)^{\lambda}\left(P(Class\ 2)\right)^{1-\lambda} \int_{-\infty}^{+\infty} \left(P(x|Class\ 1)\right)^{\lambda}\left(P(x|Class\ 2)\right)^{1-\lambda} dx \tag{9}$$

If the probability distributions of the features obey normal or exponential distribution families, the integral part of (9) can be solved through (10) (Fukunaga, 1992).

$$\int_{-\infty}^{+\infty} \left(P(x|Class\ 1)\right)^{\lambda}\left(P(x|Class\ 2)\right)^{1-\lambda} dx = e^{-\theta(\lambda)} \tag{10}$$

The $\theta(\lambda)$ can be calculated using (11) where μ and Σ are mean vector and variance matrix of each class respectively.

$$\theta(\lambda) = \frac{\lambda(1-\lambda)}{2}[\mu_2 - \mu_1]^T[\lambda\Sigma_1 + (1-\lambda)\Sigma_2]^{-1}[\mu_2 - \mu_1] + 0.5\ log\frac{|\lambda\Sigma_1+(1-\lambda)\Sigma_2|}{|\Sigma_1|^{\lambda}|\Sigma_2|^{(1-\lambda)}} \tag{11}$$

If you consider $\alpha = 0.5$ the equation (11) become the Bhattacharyya distance which can be proven that this value is the optimal value when $\Sigma_1 = \Sigma_2$ (Fukunaga, 1992; Nielsen, 2014). In this study, the Bhattacharyya distance will be used to demonstrate the idea. It should be noted that is some case the calculated error bound might be higher than the real value. However, it is accepted because in Safety evaluation the worth case scenario is needed to be considered. As the $P(error)$ and $P(correct)$ are complimentary. Then the probability of having correct decision can be calculated using (12).

$$P(correct) = 1 - \sqrt{P(Class\ 1)P(Class\ 2)}\, e^{-\theta(\lambda)} \tag{12}$$

The Chernoff upper bound of error is usually used as a measure of separability of two classes of data, but with the above equation measures the similarity of two classes. In other words, in an ideal situation if you calculate the $P(error)$ of a class, with itself, the response should be equal to one while $P(correct)$ should be zero. The idea is to show whether data distribution in the training procedure is the same as the data distribution in the field or not. The following assumptions are needed to be considered for this method:

- The covariance matrix of the data should be positive-definite matrix to have determinant and be inversible. Therefore, in each dataset, columns with all zero, Nan or Inf values should be removed. Also, columns with zero variance should be removed or it can be possible to assign a small number like 1e-6 as their variance.
- Non-coherent datasets (e.g. XOR datasets) can reduce the accuracy of the algorithm.
- For datasets like Circular and Spiral one, it is suggested to convert the data to the polar coordination before the analysis.

Note: Using dimension reduction algorithms such as PCA and t-SNE can solve the existing limitations of the proposed algorithm.

There is a relation between existing statistical distance measures as illustrated in figure Figure 3. So, it will be possible to extend the error bound probability calculations for other distance or divergence measures (Nielsen, 2018). However, computing the error bound using some distance or divergence measure algorithms like Kullback-Leibler divergence would be computationally complex (Zahm et al., 2018).

$$\begin{tikzcd}
\text{Statistical distances} & & \text{Parameter divergences} \\
\mathrm{Bhat}_\alpha(p:q) = -\log\int p(x)^{1-\alpha}q(x)^\alpha \mathrm{d}x \arrow[rr, leftrightarrow] \arrow[d, "\lim_{\alpha\to 0+}\frac{1}{\alpha}\mathrm{Bhat}_\alpha(p:q)"'] & {\begin{matrix} p(x) = p(x;\theta_p) \\ q(x) = p(x;\theta_q) \end{matrix}} & J_F^\alpha(\theta_p:\theta_q) = (F(\theta_p)F(\theta_q))_\alpha - F((\theta_p\theta_q)_\alpha) \arrow[d, "\lim_{\alpha\to 0+}\frac{1}{\alpha}\mathrm{J}_F^\alpha(p:q)"] \\
\mathrm{KL}(p:q) = \int p(x)\log\frac{p(x)}{q(x)}\mathrm{d}x \arrow[rr, leftrightarrow] & & B_F(\theta_q:\theta_p) = F(\theta_q) - F(\theta_p) - (\theta_q-\theta_p)^\top\nabla F(\theta_q) \\
\text{Generic distributions} & {\begin{matrix}\text{Exponential families} \\ p(x;\theta) = \exp(\theta^\top x - F(\theta))\end{matrix}} &
\end{tikzcd}$$

Figure 3. The relation between statistical distances like Bhattacharyya and Kullback-Leibler and parameter divergences like Jensen divergence and Burbea-Rao distances (Nielsen, 2018)

Considering $P(Class\ 1) = P(Class\ 2)$, and converting the integral part to the cumulative distribution function as (13).

$$P(error) = \left(\int_{R_1} P_{Class\ 1}(x)\,dx + \int_{R_2} P_{Class\ 2}(x)\,dx\right) = \left(\left(1 - F_{Class\ 1}(x)\right) + \left(F_{Class\ 2}(x) - 0\right)\right) \tag{13}$$

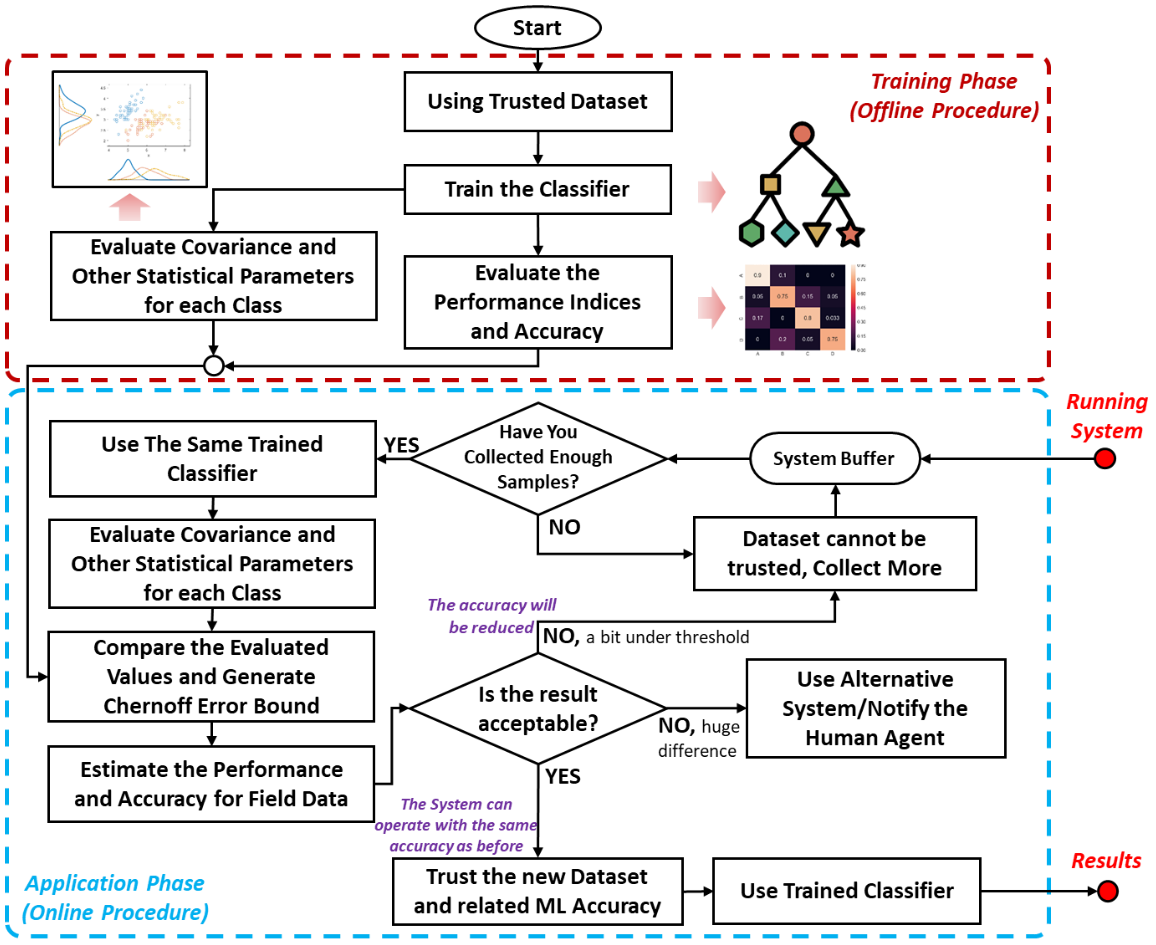
Using Kolmogorov-Smirnov distance, probability of error upper bound can be achieved as (14).

$$P(error) \approx \sup_x\left(F_{Class\ 2}(x) - F_{Class\ 1}(x)\right) \tag{14}$$

As future research, other novel statistical distance measures can be used for upper bound error probability estimation. For example, (Hadjeres & Nielsen, 2020) have proposed a novel Schoenberg-Rao distances that used both Entropy-based and geometry-aware statistical Hilbert distances.

III. Proposed Method

Figure 4 illustrates the flowchart of the proposed approach. In this flowchart, there are two main sections including training phase and application phase. A) The training phase is an offline procedure in which a trusted dataset will be used to train the intelligent algorithm that can be a machine learning or deep learning algorithm. This study will focus on the classification ability of machine learning. Thus, using a trusted dataset the classifier will be trained and its performance will be measured with existing KPIs. Meanwhile, the probability density function and statistical parameters of each class will be estimated and stored to be used for comparison. B) The second phase or application phase is an online procedure in which real-time and unlabelled data is going to be feed to the system. For example, consider an autonomous car the has been trained to detect obstacles and it should prevent a collision. Therefore, in the application phase, the trained classifier should distinguish between the road and other objects. One important and critical issue in the application phase is that the data does not have any label. So, it cannot be assured that the classifier can operate as accurate as of the training phase. In the application phase, the untrusted labels of the classifier will be used and similarly, the probability density function and statistical parameters of each class will be extracted. Using modified Chernoff Error bound the statistical difference of each class in the training phase and application phase is compared. If the statistical difference was very low, the classifier results and accuracy can be trusted (In this example the autonomous car continuous its operation), if the statistical difference was low, the system can ask for more data and re-evaluation to make sure about the distance. In case of having more statistical difference, the classifier results and accuracy is no longer valid, and the system should use alternative approach or notify human agent (In this example, the system will ask the driver to take the control of the car).



Start
Training Phase (Offline Procedure)
Using Trusted Dataset
Train the Classifier
Evaluate Covariance and Other Statistical Parameters for each Class
Evaluate the Performance Indices and Accuracy
Running System
System Buffer
Have You Collected Enough Samples?
YES
NO
Use The Same Trained Classifier
Dataset cannot be trusted, Collect More
Evaluate Covariance and Other Statistical Parameters for each Class
The accuracy will be reduced
NO, a bit under threshold
Compare the Evaluated Values and Generate Chernoff Error Bound
Is the result acceptable?
NO, huge difference
Use Alternative System/Notify the Human Agent
Estimate the Performance and Accuracy for Field Data
YES
The System can operate with the same accuracy as before
Trust the new Dataset and related ML Accuracy
Use Trained Classifier
Results
Application Phase (Online Procedure)

Figure 4.Flowchart of the proposed approach

IV. Numerical Results

There are many existing benchmarks to be used to evaluate the proposed method. However, to ease the demonstration, three simple examples including one 1-D and tow 2-D hypothetical example are used to show the capabilities and limitations of the proposed method.

A. Example 1: A 1-D hypothetical normal distribution

In this example, the same 1-D data as mentioned in problem definition section is used and five classifier algorithms are applied including the Linear discriminant analysis (LDA), Classification and Regression Tree (CART), Random Forest (RF), K-Nearest Neighbours (KNN) and Support Vector Machine (SVM). Figure 5 shows the dot plot of both accuracy and Kappa performance measure KPIs. As can be seen, all methods have a high rate of accuracy and KNN the best one. In should be mentioned LDA result are also good and they have less variance in different folds.

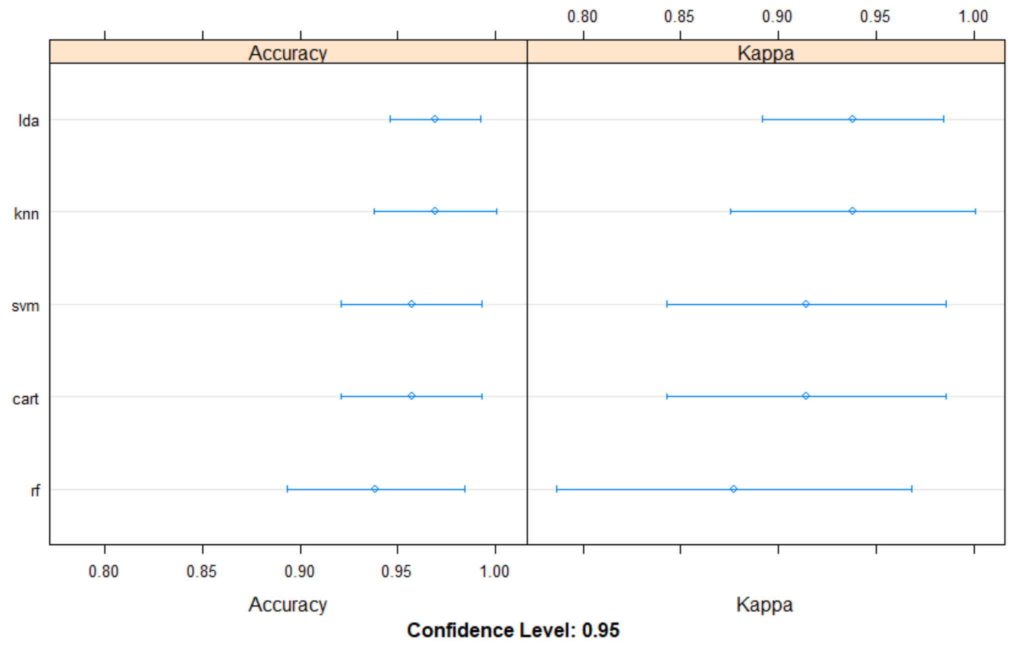
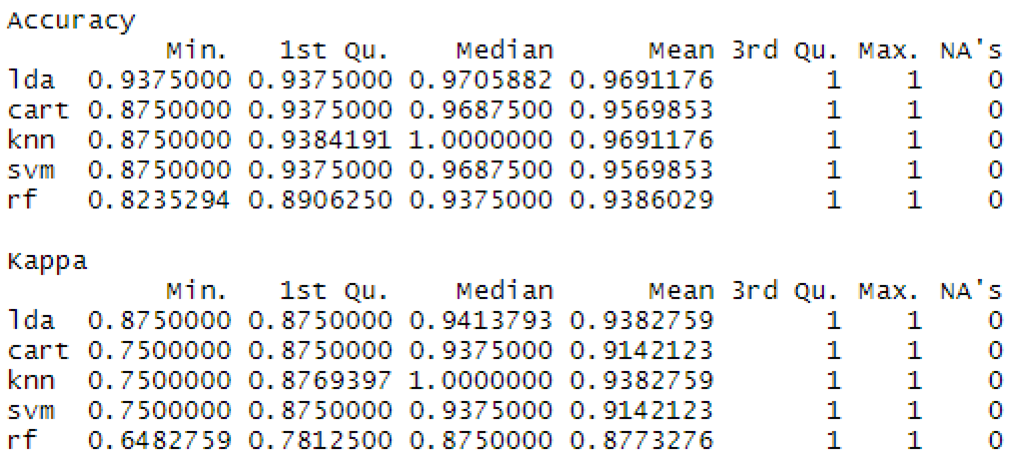


Figure 5. The Accuracy and Kappa results for each ML-based classifier (A dot plot) for 1-D hypothetical dataset

To have more details about the accuracy and Kappa measures, the Figure 6 is provided. In this figure, min, median, 1st quarter, 3rd quarter, and mean values can be checked.



```
Accuracy
          Min.   1st Qu.    Median      Mean 3rd Qu. Max. NA's
lda  0.9375000 0.9375000 0.9705882 0.9691176       1    1    0
cart 0.8750000 0.9375000 0.9687500 0.9569853       1    1    0
knn  0.8750000 0.9384191 1.0000000 0.9691176       1    1    0
svm  0.8750000 0.9375000 0.9687500 0.9569853       1    1    0
rf   0.8235294 0.8906250 0.9375000 0.9386029       1    1    0

Kappa
          Min.   1st Qu.    Median      Mean 3rd Qu. Max. NA's
lda  0.8750000 0.8750000 0.9413793 0.9382759       1    1    0
cart 0.7500000 0.8750000 0.9375000 0.9142123       1    1    0
knn  0.7500000 0.8769397 1.0000000 0.9382759       1    1    0
svm  0.7500000 0.8750000 0.9375000 0.9142123       1    1    0
rf   0.6482759 0.7812500 0.8750000 0.8773276       1    1    0
```

Figure 6. The Accuracy and Kappa results for each ML-based classifier (A dot plot) for 1-D hypothetical dataset (Details)

Table 1 compares the measured accuracy vs. the estimated accuracy based on Chernoff and Kolmogorov-Smirnov distances. The Chernoff-based results cannot be accepted in this case. However, Kolmogorov-Smirnov-based accuracy estimation provides close results. This method can be a good candidate to be used for safety certification. As mentioned in the previous figure, the LDA based method has less variance in different folds and, interestingly, Kolmogorov-Smirnov-based method has distinguished between LDA and KNN.

Table 1. Comparison between true accuracy and the estimated accuracy based on the statistical distance for a 1-D hypothetical example

Methods	Accuracy	Chernoff Acc.	Kolmogorov–Smirnov Acc.
LDA	0.9691176	0.4999481	0.9063291
CART	0.9569853	0.4990425	0.8854732
KNN	0.9691176	0.4990425	0.8854732
RF	0.9386029	0.4969803	0.8530239
SVM	0.9569853	0.4990425	0.8854732

B. Example 2: A 2-D hypothetical Spiral dataset

In this example, a two-dimensional spiral dataset with 3000 sample and two classes has been generated using bench library. To understand the data, the following feature plot has been provided in which blue point belongs to class one and purple points belongs to the second class. As it is clear, the existing data cannot be easily separated through the simple threshold and it is expected to have better performance in nonlinear classifiers. In addition, one another possible scenario can be that to convert the cartesian coordinate to a polar coordinate and then solve the problem.

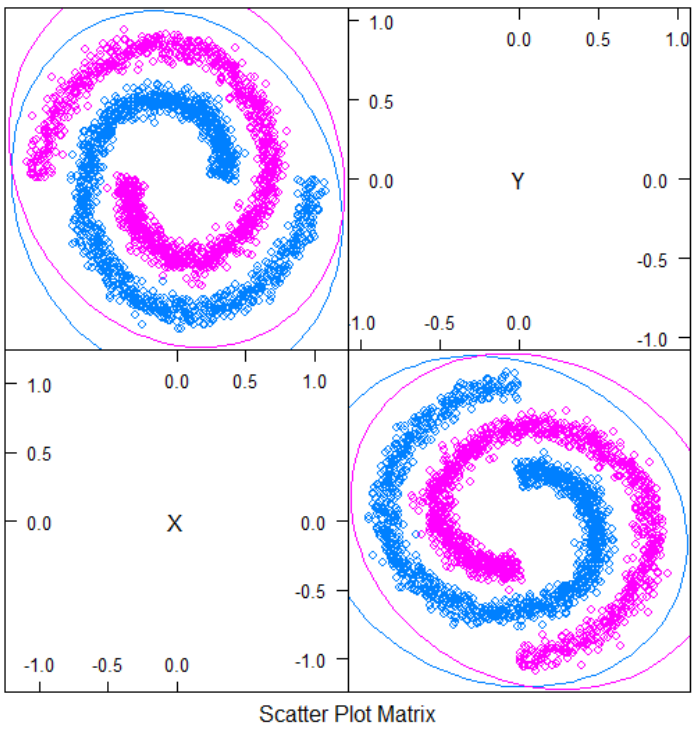


Figure 7. Features scatter plot matrix of Spiral dataset: Class one is blue and Class two is purple.

Figure 8 shows the box plot of the features for each class and as can be seen, the mean values are almost the same and it is a bit different in the variances.

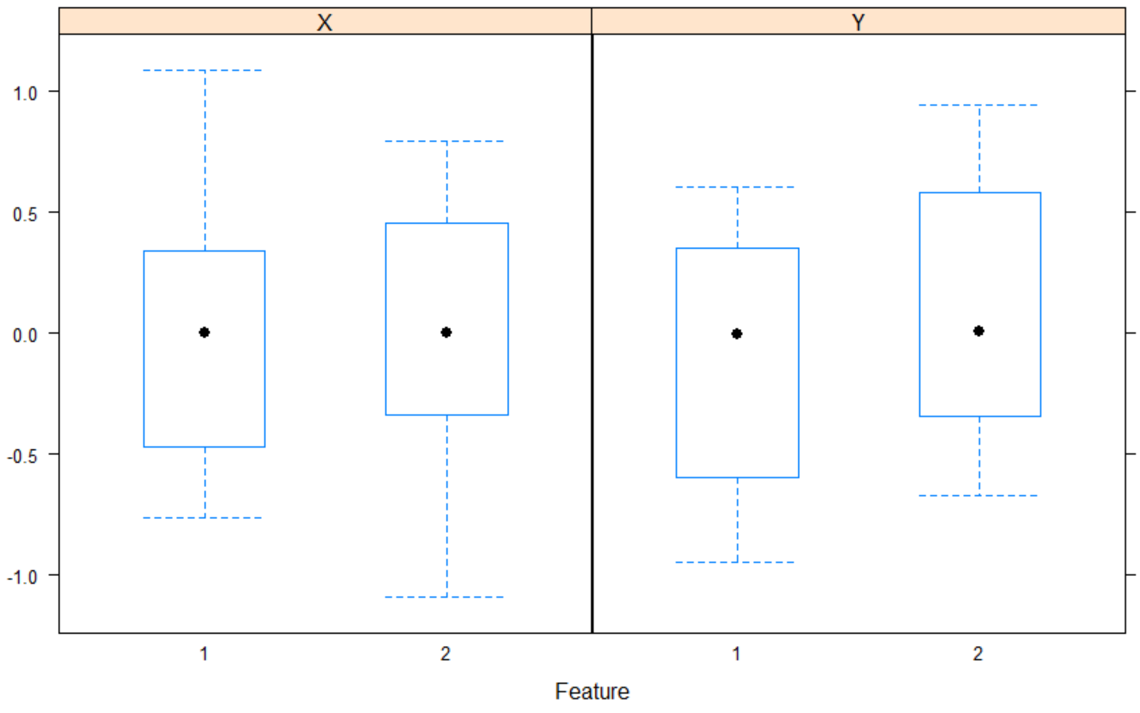


Figure 8. Features box plot for Spiral dataset

To have a better understanding of the data distribution, drawing the estimated probability density functions can be helpful. Figure 9 illustrates the probability distribution of each feature for each class. The data distributions are not following the normal or exponential distributions and we can consider them as a gaussian mixture or hyper-exponential families.

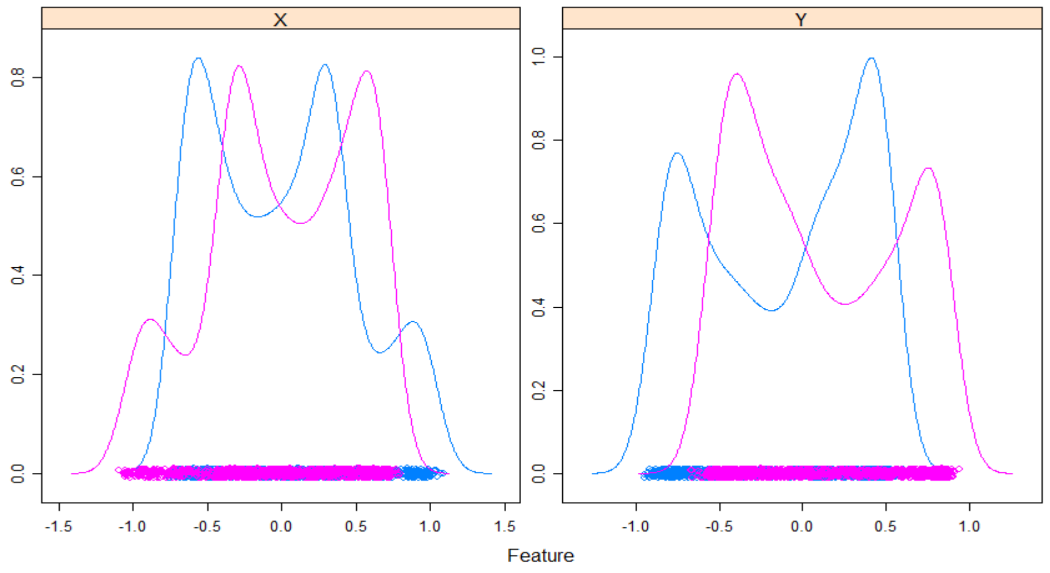


Figure 9. Feature probability density function plot for Spiral dataset

To start the ML-based classification, 80 percent of the dataset has been used for train and test and 20 percent of the dataset has been used for validation. The k-fold cross-validation is also applied where k is 10. Both linear and nonlinear classifiers have been selected for classification. The Linear discriminant analysis (LDA) and the Classification and Regression Tree (CART) are used as linear methods. Moreover, The Random Forest (RF), K-Nearest Neighbours (KNN) and Support Vector Machine (SVM) are applied as nonlinear methods. Furthermore, the accuracy and Kappa measure are used to measure the performance of each classifier. Figure 10 provides a dot-plot of the measures. It is clear, that the SVM and KNN outperformed other methods. RF also has a good performance, but LDA and CART have a low rate of performance as it was expected. By comparing the Kappa and accuracy measures it seems that Kappa is somehow magnifying the problem.

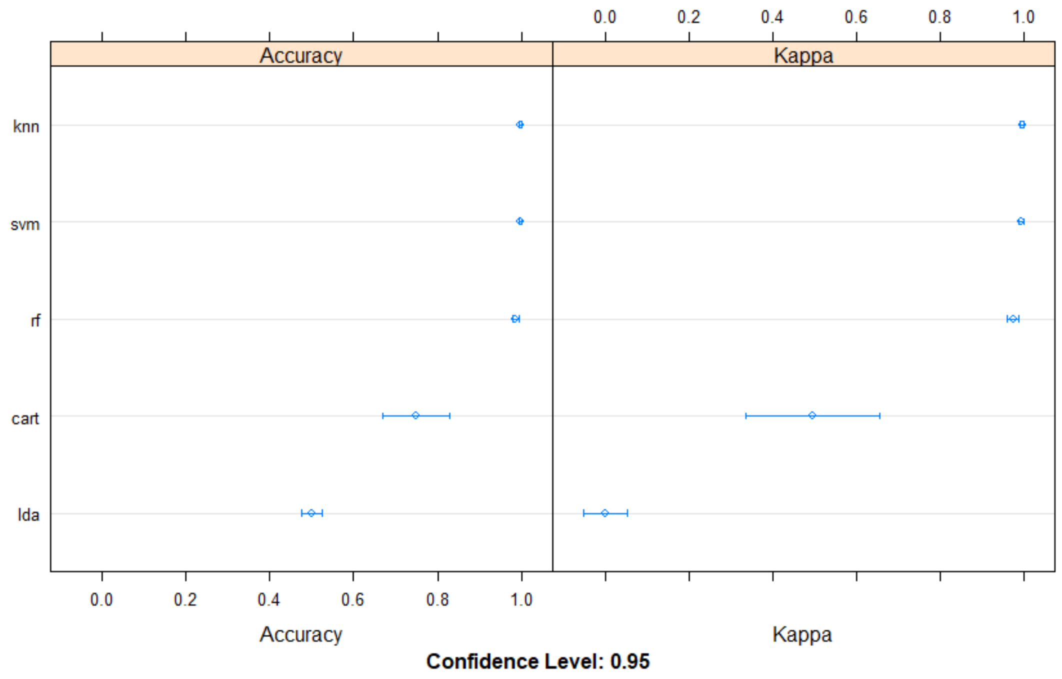
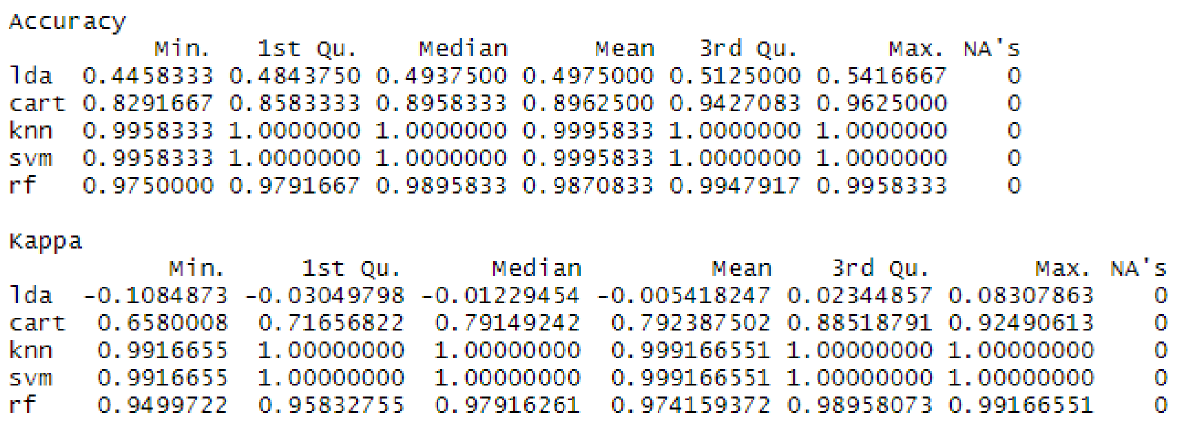


Figure 10. The Accuracy and Kappa results for each ML-based classifier (A dot plot) for Spiral dataset

To have the exact values of both Accuracy and Kappa, Figure 11 is provided. In this example, KNN and SVM both with the mean accuracy of 0.9995833 were the best one and LDA with the mean accuracy of 0.4975000 was the worst one.



```
Accuracy
          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda  0.4458333 0.4843750 0.4937500 0.4975000 0.5125000 0.5416667    0
cart 0.8291667 0.8583333 0.8958333 0.8962500 0.9427083 0.9625000    0
knn  0.9958333 1.0000000 1.0000000 0.9995833 1.0000000 1.0000000    0
svm  0.9958333 1.0000000 1.0000000 0.9995833 1.0000000 1.0000000    0
rf   0.9750000 0.9791667 0.9895833 0.9870833 0.9947917 0.9958333    0

Kappa
           Min.     1st Qu.      Median         Mean    3rd Qu.       Max. NA's
lda  -0.1084873 -0.03049798 -0.01229454 -0.005418247 0.02344857 0.08307863    0
cart  0.6580008  0.71656822  0.79149242  0.792387502 0.88518791 0.92490613    0
knn   0.9916655  1.00000000  1.00000000  0.999166551 1.00000000 1.00000000    0
svm   0.9916655  1.00000000  1.00000000  0.999166551 1.00000000 1.00000000    0
rf    0.9499722  0.95832755  0.97916261  0.974159372 0.98958073 0.99166551    0
```

Figure 11. The Accuracy and Kappa results for each ML-based classifier in Spiral dataset

Table 2 provides a comparison between real measured accuracy and the estimated accuracy through Chernoff distance (Bhattacharyya distance because lambda was 0.5) and Kolmogorov-Smirnov distance. The results show that for this example Chernoff-based estimated accuracy is just near the LDA error value and has failed to estimate the accuracy of the others. Meanwhile, the Kolmogorov-Smirnov based accuracy was somehow close to the real accuracy measures. However, consider for an application 0.99 accuracy (as achieved in this example) is needed, this statistical distance-based accuracy measure may not certify the accuracy. If the designer accepts 5% tolerance in its accuracy, then it may work.

Table 2. Comparison between real measured accuracy and the estimated accuracy based on statistical distance

Methods	Accuracy	Chernoff Acc.	Kolmogorov–Smirnov Acc.
LDA	0.4975000	0.4107213	0.5657740
CART	0.8962500	0.4818252	0.8533826
KNN	0.9995833	0.4992740	0.9568501
RF	0.9895833	0.4992256	0.9582609
SVM	0.9995833	0.4992740	0.9568501

C. Example 3. A 2-D hypothetical XOR dataset

A two-dimensional XOR dataset with 3000 sample and two classes is generated similarly using bench library. Figure 12 delineates a feature plot of this dataset in which blue points are class one and purple points are class two.

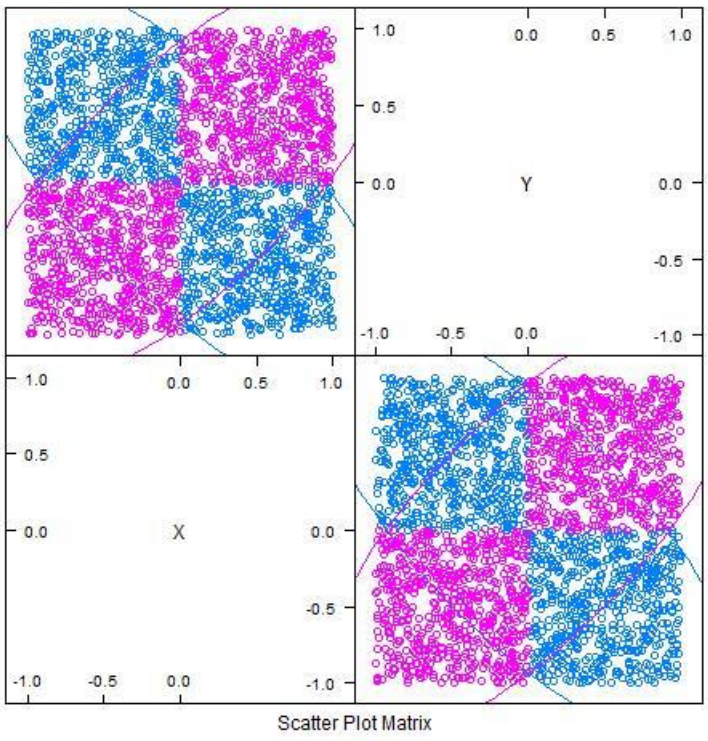


Figure 12. Features scatter plot matrix of XOR dataset: Class one is blue and Class two is purple.

Figure 13 shows the box plot for X and Y features of both class one and class two in XOR dataset. As can be seen, they are so close and have small differences.

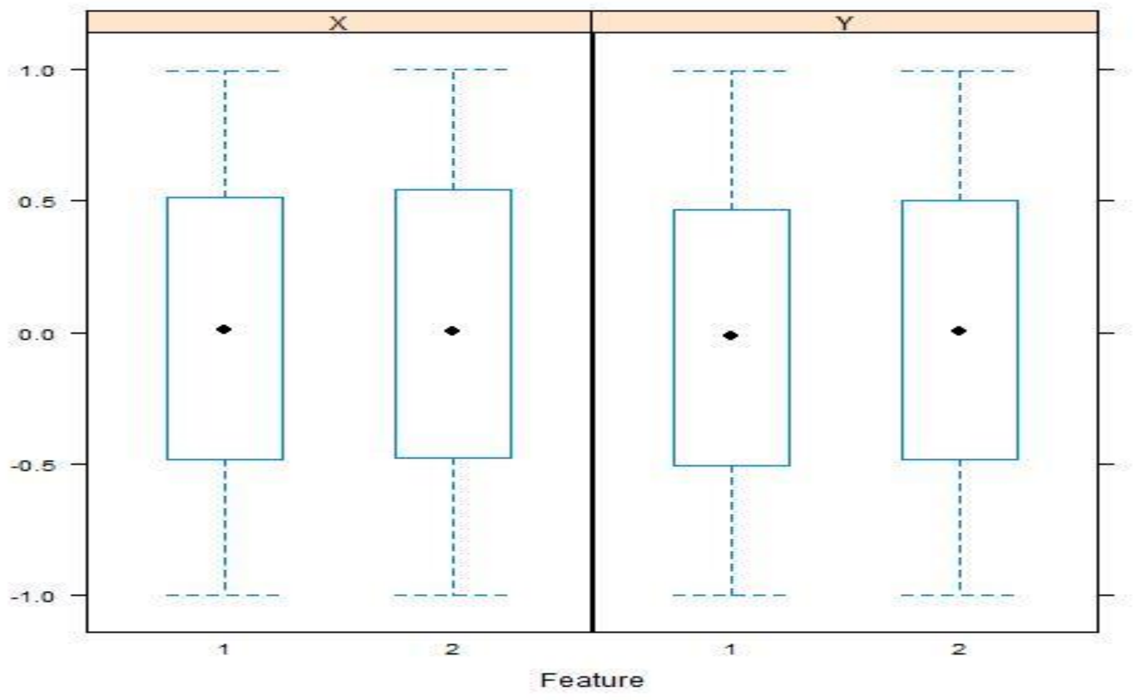


Figure 13. Features box plot for XOR dataset

The probability density functions also show the high rate of statistical similarity (Figure 14). Just like the previous example, the probability density functions are belonging to Gaussian mixture or hyper-exponential distribution families.

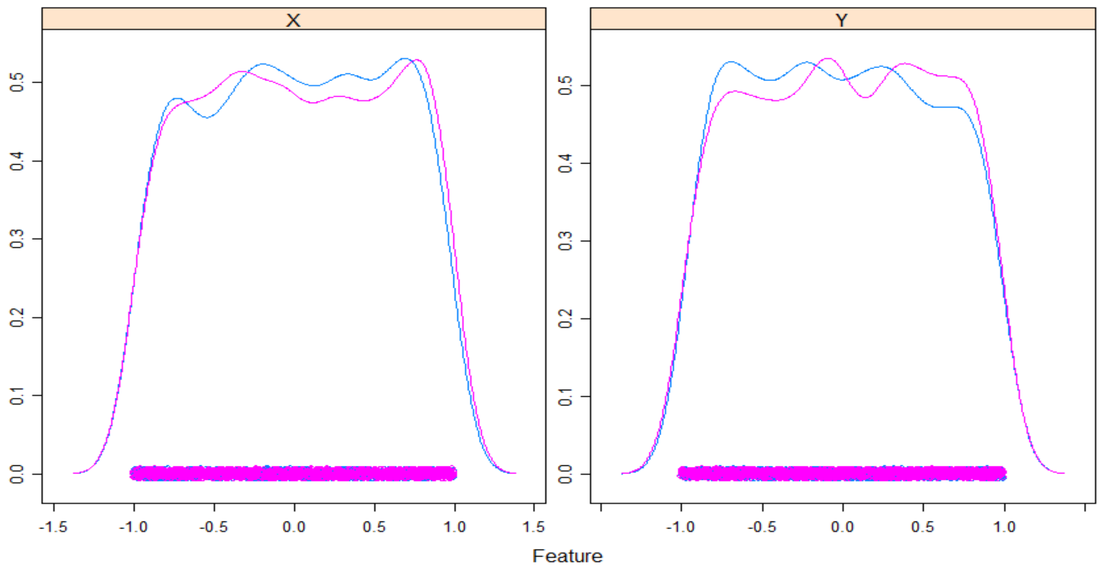


Figure 14. Feature probability density function plot for XOR dataset

Having executed ML-based classifiers, the accuracy and Kappa measure can be illustrated using a dot plot as follows. Based on this plot, all methods have a high rate of accuracy except LDA one.

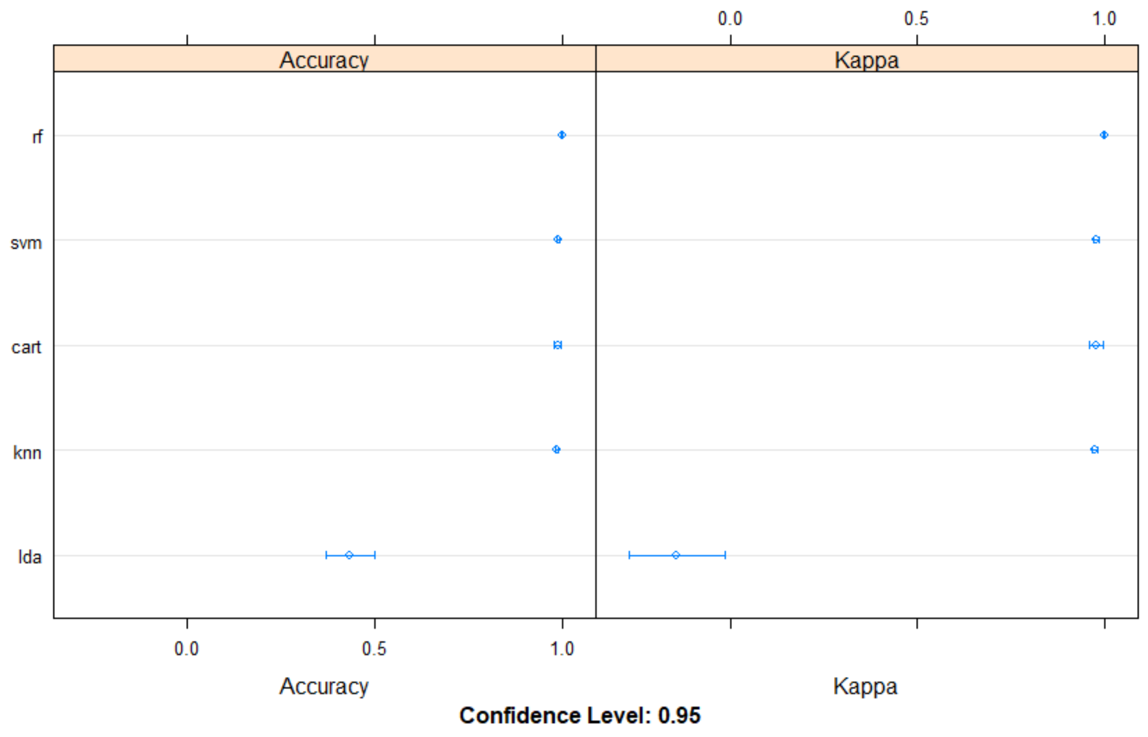


Figure 15. The Accuracy and Kappa results for each ML-based classifier (A dot plot) for Spiral dataset

Figure 16 provides exact values for each classifier and as can be seen, the best performance belongs to the random forest.

```
Accuracy
          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
lda  0.3208333 0.3666667 0.4259327 0.4337982 0.4937500 0.5833333    0
cart 0.9541667 0.9916667 0.9916667 0.9887500 0.9958290 1.0000000    0
knn  0.9750000 0.9843750 0.9875000 0.9862465 0.9906380 0.9916667    0
svm  0.9791667 0.9874608 0.9916667 0.9891649 0.9916667 0.9958333    0
rf   0.9958333 1.0000000 1.0000000 0.9995833 1.0000000 1.0000000    0

Kappa
           Min.    1st Qu.     Median       Mean     3rd Qu.      Max. NA's
lda  -0.3710921 -0.2794221 -0.1552100 -0.1429014 -0.02362776 0.1551676    0
cart  0.9082951  0.9833180  0.9833264  0.9774870  0.99165480 1.0000000    0
knn   0.9500000  0.9687283  0.9749896  0.9724799  0.98126871 0.9833333    0
svm   0.9583044  0.9749025  0.9833235  0.9783168  0.98332870 0.9916632    0
rf    0.9916632  1.0000000  1.0000000  0.9991663  1.00000000 1.0000000    0
```

Figure 16. The Accuracy and Kappa results for each ML-based classifier in XOR dataset

A comparison of measured accuracy vs. estimated accuracy based on the statistical distance for XOR dataset is provided by Table 3. With Chernoff-based estimated accuracy, LDA can only be rejected but for other methods, the proposed procedure cannot make a wise decision. While the Kolmogorov-Smirnov-based accuracy provides a better estimation and it is a better option to provide a safety or accuracy certification based on this method.

Table 3. Comparison of real measured accuracy vs the estimated accuracy based on statistical distance in XOR dataset

Methods	Accuracy	Chernoff Acc.	Kolmogorov–Smirnov Acc.
LDA	0.4337982	0.1545056	0.3350311
CART	0.9887500	0.4972425	0.9326045
KNN	0.9862465	0.4971021	0.9359829
RF	0.9995833	0.4968563	0.9386254
SVM	0.9891649	0.4967314	0.9352240

The difference of true accuracy and the KS-based estimated accuracy for each classification method and each example is provided as Table 4. Based on this table, the highest difference is about 0.098 and less that 0.1. Also, the lowest difference is 0.031.

Table 4. The difference between true accuracy and the KS-based estimated accuracy for different examples and different methods

Methods	XOR	SPIRAL	1D Normal
LDA	0.098767	0.068274	0.062789
CART	0.056146	0.042867	0.071512
KNN	0.050264	0.042733	0.083644
RF	0.060958	0.031322	0.085579
SVM	0.053941	0.042733	0.071512

V. Brief Summary

In this report, the importance and different categories of AI safety evaluation have been briefly addressed. The mathematic definition of the problem and statistical distance is also provided. The proposed method has been discussed through a flowchart and different examples have been provided to show how the proposed method work. Generally, the following capabilities and limitation can be mentioned for the proposed approach.

A. Capabilities of the proposed approach

- Through modifying the existing statistical distance and error bound measures, the proposed method enables to estimate the accuracy bound of the trained ML algorithm in the field with no label on the incoming data.
- A novel proposed human-in-loop procedure is made to certify the ML algorithm in a real-time manner. The procedure has three levels of operation: I) runtime estimated accuracy, II) Lack of enough data and need for buffering more samples (it may cause a delay in decision-making), and III) Not low runtime estimated accuracy and a human agent is needed.
- The proposed approach is easy to implement, and it can support a variety of distribution (Exponential and normal distribution families).

B. Limitations of the proposed approach and possible solutions

- The proposed algorithm is only tackling the safety evaluation problem of the machine-learning based classification. However, it can be easily expanded for clustering, dimension reduction or any problem that can be evaluated through statistical difference.

- Some of the machine learning algorithms can be robust to a certain distributional shift or variation by nature. However, in safety science, if the system robustness is not validated in the training phase, the accuracy in the application phase should not be validated.
- The proposed method will not be able to some types of adversarial attacks. For example, the one-pixel adversarial attack will not change the probability density function and the proposed method cannot detect it (Su et al., 2019).

VI. Conclusion

The rapid growth of AI applications in human life and correlated issues of its safety is an inevitable problem to be studied. The safety problem of AI and ML algorithms can be considered from different perspectives and investigating the distributional shift in datasets is one of them. Some researchers are investigating this issue and trying to make the existing ML classifiers more robust. They usually use different approaches such as adding noise, perturbation, and rotation to make sure their classifier is robust to those changes. In this report, a different perspective of the problem was selected. The statistical approaches were used to somehow estimate the accuracy of the classifier when there is no label. Among those tested approaches Kolmogorov-Smirnov was successful. The proposed human-in-loop procedure use this statistical distance to choose the decision-making source, either AI or human. The study still has a long journey to become mature, but this report can be a starting point. Some of the most important limitation and capabilities of the proposed approach were discussed in this report.

References

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., & Mané, D. (2016). *Concrete Problems in AI Safety*. http://arxiv.org/abs/1606.06565

Aslansefat, K., Bahar Gogani, M., Kabir, S., Shoorehdeli, M. A., & Yari, M. (2019). Performance evaluation and design for variable threshold alarm systems through semi-Markov process. *ISA Transactions*. https://doi.org/10.1016/j.isatra.2019.08.015

Bellemare, M. G., Danihelka, I., Dabney, W., Mohamed, S., Lakshminarayanan, B., Hoyer, S., & Munos, R. (2017). *The Cramer Distance as a Solution to Biased Wasserstein Gradients*. http://arxiv.org/abs/1705.10743

Burton, S., Habli, I., Lawton, T., McDermid, J., Morgan, P., & Porter, Z. (2020). Mind the gaps: Assuring the safety of autonomous systems from an engineering, ethical, and legal perspective. *Artificial Intelligence*, *279*, 103201. https://doi.org/10.1016/j.artint.2019.103201

Davenport, T. H., Brynjolfsson, E., McAfee, A., James, H., & Wilson, R. (2019). *Artificial Intelligence: The Insights You Need from Harvard Business Review*. Harvard Business Review.

Du-Harpur, X., Watt, F. M., Luscombe, N. M., & Lynch, M. D. (2020). What is AI? Applications of artificial intelligence to dermatology. *British Journal of Dermatology*, bjd.18880. https://doi.org/10.1111/bjd.18880

Evans, J. (2020). *Artificial Intelligence and Public Standards: A Review by the Committee on Standards in Public Life*.

Fukunaga, K. (1992). *Introduction to Statistical Pattern Recognition (Second Edition)*. Academic Press.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). *Generative Adversarial Nets*. http://www.github.com/goodfeli/adversarial

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. (2017). *Improved Training of Wasserstein GANs Montreal Institute for Learning Algorithms*. https://github.com/igul222/improved

Hadjeres, G., & Nielsen, F. (2020). *Schoenberg-Rao distances: Entropy-based and geometry-aware* *statistical Hilbert distances*. http://arxiv.org/abs/2002.08345

ISO/IEC JTC 1/SC 42: Artificial intelligence. (2020).

Laurens van der Maaten. (2014). Accelerating t-SNE using Tree-Based Algorithms. *Journal of Machine Learning Research*, *15*, 3221–3245. https://doi.org/10.1007/978-1-60761-580-4_8

Nielsen, F. (2014). Generalized Bhattacharyya and Chernoff upper bounds on Bayes error using quasi-arithmetic means. *Pattern Recognition Letters*, *42*(1), 25–34. https://doi.org/10.1016/j.patrec.2014.01.002

Nielsen, F. (2018). The Chord Gap Divergence and a Generalization of the Bhattacharyya Distance. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *2018-April*, 2276–2280. https://doi.org/10.1109/ICASSP.2018.8462244

Quiñonero-Candela, J., & Schwaighofer, A. (2009). *Dataset Shift in Machine Learning*. MIT Press.

Schulam, P., & Saria, S. (2019). *Can You Trust This Prediction? Auditing Pointwise Reliability After Learning*. http://arxiv.org/abs/1901.00403

Sharkey, A. (2019). Autonomous weapons systems, killer robots and human dignity. *Ethics and Information Technology*, *21*(2), 75–87. https://doi.org/10.1007/s10676-018-9494-0

Su, J., Vargas, D. V., & Sakurai, K. (2019). One Pixel Attack for Fooling Deep Neural Networks. *IEEE Transactions on Evolutionary Computation*, *23*(5), 828–841. https://doi.org/10.1109/TEVC.2019.2890858

Theodoridis, S., & Koutroumbas, K. (2009). Pattern Recognition. In *Pattern Recognition*. Elsevier Inc. https://doi.org/10.1016/B978-1-59749-272-0.X0001-2

Wiens, J., Saria, S., Sendak, M., Ghassemi, M., Liu, V. X., Doshi-Velez, F., Jung, K., Heller, K., Kale, D., Saeed, M., Ossorio, P. N., Thadaney-Israni, S., & Goldenberg, A. (2019). Do no harm: a roadmap for responsible machine learning for health care. *Nature Medicine*, *25*(9), 1337–1340. https://doi.org/10.1038/s41591-019-0548-6

Zahm, O., Cui, T., Law, K., Spantini, A., & Marzouk, Y. (2018). *Certified dimension reduction in nonlinear Bayesian inverse problems*. http://arxiv.org/abs/1807.03712

Appendix

Code for Figure 1 and Figure 2.

```
# Adding Needed Libraries
library(ggplot2)
library(viridis)
library(dplyr)
# Nuromal Distributed Random Number Generation for Class 1
set.seed(20000)
df = data.frame(
  LBs = factor(rep(c("Class 1", "Class 2"), each=100)),
  D_Val = c(rnorm(100, mean=2, sd=1),rnorm(100, mean=6, sd=1)),
  x = seq(1,200,1)
)
plot1 <- ggplot(df, aes(x=D_Val, fill = LBs)) +
  geom_density(alpha=0.4, size = 1.5, linetype="dashed") +
  scale_fill_brewer(palette = "Dark2") +
  ylab("Probability") +
  xlab("Data Range") +
  geom_vline(xintercept = 4, colour = "red", size = 1.5, linetype="dashed") +
  ggtitle("Probability Density Function by Classes") +
  labs(fill = "Classes") +
  theme(plot.title = element_text(color="red", size=14, face="bold.italic", hjust = 0.5))
plot1
plot3 = ggplot(df, aes(x = x, y = D_Val, colour = LBs)) +
      geom_line( color="grey", size = 1) +
      geom_point(shape = 21, size = 4, fill = "grey", alpha = 0.5, stroke = 2) +
      geom_hline(yintercept = 4, colour = "red", size = 1.5, linetype="dashed") +
      ggtitle("Hypothetical Measurement")+
      ylab("Measured Value") +
      xlab("Time (Second)") +
      labs(colour = "Classes") +
      theme(plot.title = element_text(color="red", size=14, face="bold.italic", hjust = 0.5)) +
      scale_color_manual(values = c("green", "orange")) +
      scale_fill_manual(values = c("green", "orange"))
plot3
```

Code for Examples 1, 2 and 3.

```
# For Plots
library(ggplot2)
# For ML and Plots
library(caret)
# For Datasets
library(mlbench)
# For ROC Curve
#library(pROC)
# For Statistical Distance Measure
library(MASS)
library(Compositional)
library(stats)
# For matrix calculations
library(matlib)
# For %>%
library(dplyr)

# For example 1, the dataset has been generated using previous code

# Generating Spiral Dataset for example 2
Spiral = mlbench.spirals(3000,1,0.05)

# XOR = mlbench.xor(3000,2) # The data can be replaced with this line for Example 3.

df = data.frame(X = Spiral$x[,1], Y = Spiral$x[,2], LBs = factor(Spiral$classes))

# create a list of 80% of the rows in the original dataset we can use for training
validation_index = createDataPartition(df$X, p=0.80, list=FALSE)
# select 20% of the data for validation
validation = df[-validation_index,]
# use the remaining 80% of data to training and testing the models
dataset = df[validation_index,]

dim(dataset)

# list types for each attribute
sapply(dataset, class)

head(dataset)

# summarize the class distribution
percentage <- prop.table(table(dataset$LBs)) * 100
cbind(freq=table(dataset$LBs), percentage=percentage)

summary(dataset)

# Separating input and output
x = dataset[,1:2]
y = dataset[,3]
```

```
# boxplot for each attribute on one image
par(mfrow=c(1,2))
for(i in 1:2) {
  boxplot(x[,i], main=names(df$LBs)[i])
}

# scatterplot matrix
# jpeg("featurePlot_spiral.jpg", width = 866, height = "553")
pdf("featurePlot_spiral.pdf")
featurePlot(x=x, y=factor(y), plot="ellipse")
dev.off()

# box and whisker plots for each attribute
# jpeg("box_featurePlot_spiral.jpg", width = 866, height = "553")
featurePlot(x=x, y=factor(y), plot="box")
# dev.off()

# density plots for each attribute by class value
# jpeg("density_spiral.jpg", width = 866, height = "553")
scales = list(x=list(relation="free"), y=list(relation="free"))
featurePlot(x=x, y=factor(y), plot="density", scales=scales)
# dev.off()

control = trainControl(method="cv", number=10)
metric = "Accuracy"

# a) linear algorithms
set.seed(7)
fit.lda = train(LBs~., data=dataset, method="lda", metric=metric, trControl=control)
# b) nonlinear algorithms
# CART
set.seed(7)
fit.cart <- train(LBs~., data=dataset, method="rpart", metric=metric, trControl=control)
# kNN
set.seed(7)
fit.knn <- train(LBs~., data=dataset, method="knn", metric=metric, trControl=control)
# c) advanced algorithms
# SVM
set.seed(7)
fit.svm <- train(LBs~., data=dataset, method="svmRadial", metric=metric, trControl=control)
# Random Forest
set.seed(7)
fit.rf <- train(LBs~., data=dataset, method="rf", metric=metric, trControl=control)

#summarize accuracy of models
results <- resamples(list(lda=fit.lda, cart=fit.cart, knn=fit.knn, svm=fit.svm, rf=fit.rf))
summary(results)

dotplot(results)
print(fit.svm)
```

```
# estimate skill of LDA on the validation dataset
predictions_lda = predict(fit.lda, validation)

predictions_cart = predict(fit.cart, validation)

predictions_rf = predict(fit.rf, validation)

predictions_knn = predict(fit.knn, validation)

predictions_svm = predict(fit.svm, validation)

# confusionMatrix(predictions, validation$LBs)

# a = dataset$X[which(dataset$LBs == 1)]
a = dataset[which(dataset$LBs == 1),1:2]

# b_lda = validation$X[which(predictions_lda == 1)]
b_lda = validation[which(predictions_lda == 1),1:2]

b_cart = validation[which(predictions_cart == 1),1:2]

b_rf = validation[which(predictions_rf == 1),1:2]

b_knn = validation[which(predictions_knn == 1),1:2]

b_svm = validation[which(predictions_svm == 1),1:2]

Chernoff_Dist_nD = function(a, b)
{
# alpha = 0.5
  mean_diff = abs(colMeans(a) - colMeans(b))
  var_avg = (var(a) + var(b)) * 0.5
  ln_coeff = 0.5 * log( det(var_avg) / sqrt(det(var(a)) * det(var(b))))

  dist = 0.125 * (t(mean_diff) %*% ginv(var_avg) %*% mean_diff) + ln_coeff

  1 - sqrt(0.25)*exp(-dist)
}

c_dist_lda = Chernoff_Dist_nD(a, b_lda)
c_dist_cart = Chernoff_Dist_nD(a, b_cart)
c_dist_rf = Chernoff_Dist_nD(a, b_rf)
c_dist_knn = Chernoff_Dist_nD(a, b_knn)
c_dist_svm = Chernoff_Dist_nD(a, b_svm)

# Kolmogorov-Smirnov Tests
ks_lda1 = ks.test(a[,1], b_lda[,1])
ks_lda2 = ks.test(a[,2], b_lda[,2])
ks_lda = max(ks_lda1$statistic, ks_lda2$statistic)

ks_cart1 = ks.test(a[,1], b_cart[,1])
```

```
ks_cart2 = ks.test(a[,2], b_cart[,2])
ks_cart = max(ks_cart1$statistic, ks_cart2$statistic)

ks_rf1 = ks.test(a[,1], b_rf[,1])
ks_rf2 = ks.test(a[,2], b_rf[,2])
ks_rf = max(ks_rf1$statistic, ks_rf2$statistic)

ks_knn1 = ks.test(a[,1], b_knn[,1])
ks_knn2 = ks.test(a[,2], b_knn[,2])
ks_knn = max(ks_knn1$statistic, ks_knn2$statistic)

ks_svm1 = ks.test(a[,1], b_svm[,1])
ks_svm2 = ks.test(a[,2], b_svm[,2])
ks_svm = max(ks_svm1$statistic, ks_svm2$statistic)

method_vec = c("LDA","CART","RF","KNN","SVM")
c_dist_vec = c(1-c_dist_lda, 1-c_dist_cart, 1-c_dist_rf, 1-c_dist_knn, 1-c_dist_svm)
ks_vec = c(ks_lda, ks_cart, ks_rf, ks_knn, ks_svm)
```