How to Make Machine Learning Financial Recommendations More Fair: Theoretical Explanation of Empirical Results

Tho M. Nguyen¹, Saeid Tizpaz-Niari², and Vladik Kreinovich²

¹Faculty of Banking and Finance, Ho Chi Minh City Open University
Ho Chi Minh City, Vietnam, tho.nm@ou.edu.vn

²Department of Computer Science, University of Texas at El Paso
500 W. University, El Paso, Texas 79968, USA
saeid@utep.edu, vladik@utep.edu
1. Machine learning have become ubiquitous in decision making

- In many practical situations, we need to make a recommendation.
- For example, if a person applies for a loan, the bank needs to decide:
  - whether providing this loan is worth a risk, and
  - if yes, shall the bank give the full requested amount or a smaller amount.
- Similar decisions need to be made in other financial situations: e.g.
  - when a start-up company applies for funding, or
  - when a well-established company wants the bank to finance a project.
- In all such situations, there is no ready formulas for making a decision.
- So, we have to rely on specialists.
- Some specialists are more skilled, some are less skilled.
2. Machine learning have become ubiquitous in decision making (cont-d)

- It would therefore be nice to incorporate the expertise of skilled experts into a computer system.
- This would help less experienced decision makers make good decisions.
- In some cases, experienced experts can formulate their decision making process in terms of precise rules.
- However, in financial domain, such situations are rare.
- In many cases, the decisions of experienced experts:  
  - are motivated largely by their intuition,  
  - and they cannot describe their intuition in precise terms.
- We do not have the rules.
3. Machine learning have become ubiquitous in decision making (cont-d)

- So all we know about the decision making of experienced experts is:
  - a set of cases in which they made decision,
  - i.e., the set of pairs \((x_k, y_k), k = 1, 2, \ldots\), where:
    * \(x_k\) is the information the decision makers have about the \(k\)-th case, and
    * \(y_k\) is the resulting decision.
- Based on these pairs, we need to come up with an algorithm \(f(x)\) that would:
  - given a general input \(x\),
  - produce a reasonable decision \(y = f(x)\).
- For the cases in which experienced decision makers made reasonable decisions, this algorithm should generate similar decisions.
- So, we should have \(f(x_k) \approx y_k\) for all \(k\).
4. Machine learning have become ubiquitous in decision making (cont-d)

- In computer science, the problem of generating an algorithm $f(x)$ based on known examples $(x_k, y_k)$ is known as machine learning.
- In statistics, this problem is also known as regression.
- In the last decades, many effective machine learning techniques have been developed, including deep learning.
- These techniques are actively – and successfully – used in many application areas, including finances.
- Banks and other financial institutions are using machine learning to decide whether to give a loan to a person.
5. Comment

- Such systems are used in many other applications,
- E.g., such systems are used to decide:
  - whether a person who has served a significant portion of his/her sentence should be given a parole
  - or this person needs more time to reform.
- In this talk, we concentrate on the financial applications.
- However, the same ideas and results can be (and are) used in other applications as well.
6. Main problem: insufficient fairness

- In many cases, the current software packages produce reasonable results.
- However, many of these packages have a problem with fairness:
  - while they produce good results on average,
  - for applicants from some minority groups their results are often not fair at all.
- There have been many cases when the system were tested.
- It turned out that the chances of a person to get a loan significantly increases:
  - if his/her ethnicity and/or race was changed in the application,
  - or if we replace a female name with a male name.
- How can we make the systems more fair?
7. Is there an easy solution?

- At first glance, it may seem that with machine learning approach:
  - there is not much that we can do
  - other than changing the general machine learning algorithms.

- Algorithms for training a machine learning system (such as a neural network) are themselves not biased.

- Maybe the inputs are somewhat biased.

- However, this does not explain the bias in the machine learning results.

- This bias is often much larger than the bias in the samples on which these systems have been trained.

- So, at first glance, the problem is complex.
8. **Is there an easy solution (cont-d)**

- We need to replace the current machine learning training algorithms with new algorithms:
  - that would not simply generalize the available data, but
  - that would also explicitly take into account the need for fairness.

- This is a challenging research task,

- So far, researchers have not produced such new algorithms.
9. But there is hope

- The above – somewhat pessimistic – view comes:
  - from the viewpoint of an outsider,
  - to whom machine learning (especially deep learning) is a magic tool:
    - you give it data and it immediately generates great results.

- Such magic happens sometimes.

- However, as researchers who actually apply machine learning know very well, such ideal situations are rare.

- The truth is that each machine learning tool comes with many parameters that we need to select to start training.

- These parameters are called hyperparameters – to distinguish them from the parameters describing the resulting model $y = f(x)$. 
10. But there is hope (cont-d)

- For example, to train a neural network, we need to decide:
  - how many layers to use,
  - how many neurons to have in each layer,
  - when to stop training, etc.

- For some values of these hyperparameters, we get spectacular results.

- However, for some other values, the system does not learn at all.

- In general, the choice of hyperparameters significantly affects the training result.

- So why not try to see if a proper choice of hyperparameters would make the resulting systems more fair?
11. This idea was tested

- An extensive research in this direction was actually performed by one of us (STN).

- Here is a brief description of the results.

- For most of hyperparameters, there seems to be no clear direction of the resulting effect.

- Change may be drastic, but:
  - in some cases, they go in one direction, and
  - sometimes they go in an opposite direction.
12. This idea was tested (cont-d)

- For example:
  - sometimes placing the neurons in fewer layers makes the trained system significantly more fair, while
  - in other case, a similar transformation further decreases the original system’s level of fairness.

- There are only a few hyperparameters that consistently affect fairness the same way.

- In a nutshell, these hyperparameters are related:
  - to the number of features that is taken into account, and
  - to the number of training iterations.

- In both cases, the larger these hyperparameters, the more fair the resulting trained system.

- How can we explain these empirical results?

- This is what we do in this talk.
13. Explaining the effect of number of features is straightforward

- Of course, the more features we take into account:
  - the more information we get about a person,
  - the smaller is the effect of each individual feature.

- Crudely speaking:
  - if all the banking system know is the applicant’s gender,
  - then this will be the only information that it can use to make a decision on whether to give a loan or not.

- On the other hand:
  - if the banking system takes 10 different features into account, gender being one of them,
  - then we expect that only 10% of its decision will be affected by gender.
14. Explaining the effect of number of features is straightforward (cont-d)

- And this is exactly what the above-cited empirical study observed:
  - the more features we take into account,
  - the more fair the results.
15. But how can we explain the effect of the number of iterations?

- In contrast to the effect of the number of features, the effect of the number of iterations is somewhat puzzling.
- The more iterations we perform, the closer the values $f(x_k)$ generated by the trained model will be to the corresponding values $y_k$.
- In other words:
  - as we increase the number of iterations,
  - the closer the trained system will be to the original expert decisions.
- But why would the resulting trained system be more fair?
- For some time, this was a puzzle for us, until we came up with a reasonable explanation.
16. Towards our explanation

- From the informal mathematical viewpoint, what training means is minimizing the difference between:
  - the values $f(x_k)$ generated by the model and
  - the corresponding values $y_k$.
- At first, before training, the values $f(x_k)$ are very different from the desired values $y_k$.
- As we train, the difference becomes smaller and smaller.
- This is especially true for neural networks (NN), where:
  - each training step is, in effect, an application of gradient descent
  - to minimize the objective function describing this difference.
17. How is this difference described?

- In the traditional neural networks, the difference \( d \) was described by the Least Squares expression:

\[
d = \sum_{k} (f(x_k) - y_k)^2.
\]

- This expression is usually used on statistical analysis.

- In modern machine learning algorithm, usually, somewhat more sophisticated expressions are used.

- However, the gist of our explanation does not depend on what expression we use.

- So, for simplicity, we will illustrate our idea on the example of the Least Squares expression.
18. Our explanation

- We usually stop iterations:
  - when the value of the difference $d$ becomes sufficiently small,
  - i.e., when the value $d$ does not exceed some threshold $t$.
- How do we usually select this threshold?
- Suppose that we want to predict the desired values $y_k$ with accuracy 10%.
- In case we taking about “yes”-“no” decision – with the probability of 90%.
- In other words, we want to have $|f(x_k) - y_k| \leq 0.1 \cdot y_k$ for all $k$.
- This means that we want to have $(f(x_k) - y_k)^2 \leq 0.01 \cdot y_k^2$ for all $k$.
- Thus, for the sum $d$, we want to have
  \[ d = \sum_k (f(x_k) - y_k)^2 \leq 0.01 \cdot \sum_k y_k^2. \]
19. Our explanation (cont-d)

- So, in this case, it is reasonable to select a threshold \( t = 0.01 \cdot \sum_k y_k^2 \).

- Usually, all the values \( y_k \) have about the same order of magnitude \( y_k \approx y \).

- So this natural threshold is \( t = 0.01 \cdot K \cdot y^2 \), where \( K \) is the overall number of training cases.

- After we apply this stopping criterion, we get approximate equality

\[
d = \sum_k (f(x_k) - y_k)^2 \approx t = 0.01 \cdot K \cdot y^2.
\]

- Suppose now that we have a minority population that forms about 10% of the total.

- This means, in particular, that out of \( K \) cases, approximately 10% deal with applicants from this minority population.

- Let \( S \) denote the set of all indices \( k \) corresponding to this minority group.
Let $M$ denote the number of such indices (so that $N \approx 0.1 \cdot K$).

What can we conclude about the recommendations corresponding to minority folks?

The only thing that we can conclude is that:

- the sum $\sum_{k \in S} (f(x_k) - y_k)^2$ corresponding to applicants from this population
- does not exceed the overall sum:

$$\sum_{k \in S} (f(x_k) - y_k)^2 \leq 0.01 \cdot K \cdot y^2.$$

Let us denote the average values of the difference $|f(x_k) - y_k|$ corresponding to minority population by $\delta$.

Then we get $M \cdot \delta^2 \leq 0.01 \cdot K \cdot y^2$.

Since $M \approx 0.1 \cdot K$, this implies that $\delta^2 \leq 0.1 \cdot y^2$. 

20. Our explanation (cont-d)
Thus, by taking the square root of both sides, we get $\delta \leq 0.3 \cdot y$.

So:

– while on average, the trained system reproduces the original decisions with accuracy 10%,
– for minority folks, the only thing we can guarantee is that the accuracy will be not larger than 30%.

And if the accuracy is 30% for all the minority folks, it can still lead to 10% accuracy in general.

What happens if we increase the number of iterations?

In effect, this means that we further decrease the threshold.
22. Our explanation (cont-d)

- In the above example, if we decrease the threshold to $0.001 \cdot K \cdot y^2$, then:
  
  - the resulting inequality $\sum_{k \in S} (f(x_k) - y_k)^2 \leq 0.001 \cdot K \cdot y^2$
  
  - will guarantee the average-10% accuracy for the minority population as well.

- Thus, it will lead to more fair results.

- So, an increase in the number of iterations indeed leads to more fairness.
23. Acknowledgments

This work was supported in part by:

- National Science Foundation grants 1623190, HRD-1834620, HRD-2034030, and EAR-2225395;
- AT&T Fellowship in Information Technology;
- program of the development of the Scientific-Educational Mathematical Center of Volga Federal District No. 075-02-2020-1478, and
- a grant from the Hungarian National Research, Development and Innovation Office (NRDI).