AdeptID

Research Data Scientist - Technical Exercise

August 2022

Background

Deploying models that impact people requires careful ethical decision making. At AdeptID, our Ethical AI Framework contains three components (see blog post):

- At a base level, we need to address **Algorithmic Bias**.
- Once we've accomplished this, we move onto our second phase, which is **Addressing Systematicity**, or systematic and arbitrary errors.
- Lastly, this framework needs to be responsibly governed, and requires a democratic **Governance Model**.

This technical exercise focuses on **Systematicity**. Systematic and arbitrary exclusion of **individuals** based on algorithmic decisions represents a moral harm. As we deploy talent models at scale, we have to design systems that have some degree of randomness in order to ensure that individuals are not arbitrarily blocked from economic opportunities. For example:

- Jules Doe applies to several data scientist positions at different companies. At each company, Jules meets with a different interviewer for the role.
- Because of differences in the opinions of the interviewers and the content of the interviews themselves, each of the potential interviewers will arrive at a slightly different assessment of Jules.
- If Jules is a borderline candidate for the role, this means that some interviewers will approve Jules while others will reject Jules. This variability in outcome is a desirable property as it prevents Jules from being systematically (and potentially arbitrarily) blocked from becoming a data scientist.
- Suppose instead that all companies hiring for similar roles use an ***identical*** machine learning model to assess candidates. In this scenario, a borderline candidate will receive ALL approvals (if they are above the decision threshold) or ALL rejections (if they are below the decision threshold).
- Because all models are imperfect, the correlated decisions made above represent a moral harm, **Systematicity**. Addressing Systematicity requires us to incorporate the random variation that exists in a traditional interview process into the automated decisions of ML systems.

There are several ways to accomplish this, some of which are described in The Algorithmic Leviathan (in particular see: *Section 4 Technical Solutions*). In this paper, the authors outline two primary ways of addressing systematicity:

A. Training a set of models (with similar accuracy) and randomly drawing from the set of models at prediction time
B. Directly introducing (bounded) randomness to scores at prediction time

Problem Statement

In this exercise, you will use the data provided to construct a prediction system that addresses systematicity. Please feel free to implement the proposed solution(s) described above and/or construct alternative mechanisms for mitigating systematicity. Your implementation should include two primary components:

1. A **predict()** function or method that implements one or more mechanisms for addressing systematicity (i.e. correlation of errors)

2. A set of metrics and associated charts that allow you to measure and evaluate the effectiveness of your systematicity mitigation.

Along with your code, please put together **either** a memo or presentation describing your approach and the reasoning behind the metrics you selected. We're primarily interested in understanding the logic behind your metric construction and model evaluation so please emphasize this in your deliverable. For example: What risks would your model present if it *did not* incorporate mitigation for correlated errors?

Please note: The overall accuracy of these models is *not* of particular importance to this exercise so *please don't spend too much time optimizing for model accuracy*. The goal is to evaluate your comfort level with thinking critically about system design and associated metrics for measurement.

I hope that you find this exercise interesting! Best of luck!

Data Provided

1. **data.csv** - This file contains sample features and associated binary outcomes for training simple logistic regression classifiers.
2. **readme.csv** - This file contains relevant descriptions and contextual information for each of the variables in data.csv

Deliverables

1. Code - Github repository containing the code you wrote to complete the assignment.
2. Memo or Presentation - Describe your approach, key assumptions, metrics and conclusions. Importantly, describe how you would expand this analysis were it a complete project with a larger timeline.

What this tests

1. Can you take an abstract idea and implement it in code?
2. Can you construct metrics to evaluate the performance of novel models?
3. Can you identify, reason through and address constraints from real world problems?
4. Can you communicate findings clearly?