Debiaser - AI Bias Detection and Mitigation Tool for Clinical Decision Making

CVP Data Science Team

Wei Chien
Manpreet Khural
Lauren Winstead
Cal Zemelman (POC), calzemelman@cvpcorp.com, (703) 822-7776

Video: https://youtu.be/O4GG6Ph55U8

Abstract

Artificial intelligence algorithms are increasingly being adopted as decision-making aids with the promise of overcoming biases of human decision-makers. Machine learning models used in this fashion may unintentionally amplify or even create bias because of choices made during development, or they may become biased from data that they were trained on. Due to the increasing use of AI systems to supplement regular decision-making and deep-rooted disparities in the US healthcare system where this training data comes from, there have been growing demands for model transparency, explainability, and interpretability to determine the presence of bias. CVP's Data Science Team investigated ways to automatically measure certain types of bias and mitigate them without human intervention.

Our team successfully reduced social bias when training and optimizing a LightGBM (gradient-boosted decision tree) model through effective bias measurement and mitigation. Our solution is runnable on any modern PC, in an office or in the cloud, and can be used on any structured dataset. With only one executable Python script and one Python module, it will first measure and then mitigate many types of bias, leading to more equitable healthcare outcomes across the country.

GitHub Code

Main site:
https://github.com/cvp-bias-detection-healthcare/cvp-bias-detection-healthcare.github.io

Please see this subfolder for pre-generated example visualizations:
https://github.com/cvp-bias-detection-healthcare/cvp-bias-detection-healthcare.github.io/tree/main/reports

Methodology Overview

Our goal for this project was to develop a broad, user-friendly set of diagnostics and tools for the measurement and mitigation of bias, not a one size fits all approach that may worsen the problem of automation bias. The more we investigated the various forms of bias in AI, the more we realized they are several loosely related problems all falling under one umbrella, with no single indicator capable of summarizing bias. Much like the complexity surrounding "solving" the nation's deficit, the identification, measurement, and mitigation of AI Bias requires a **variety** of information coupled with expert judgment to recognize sustainable improvements over time.

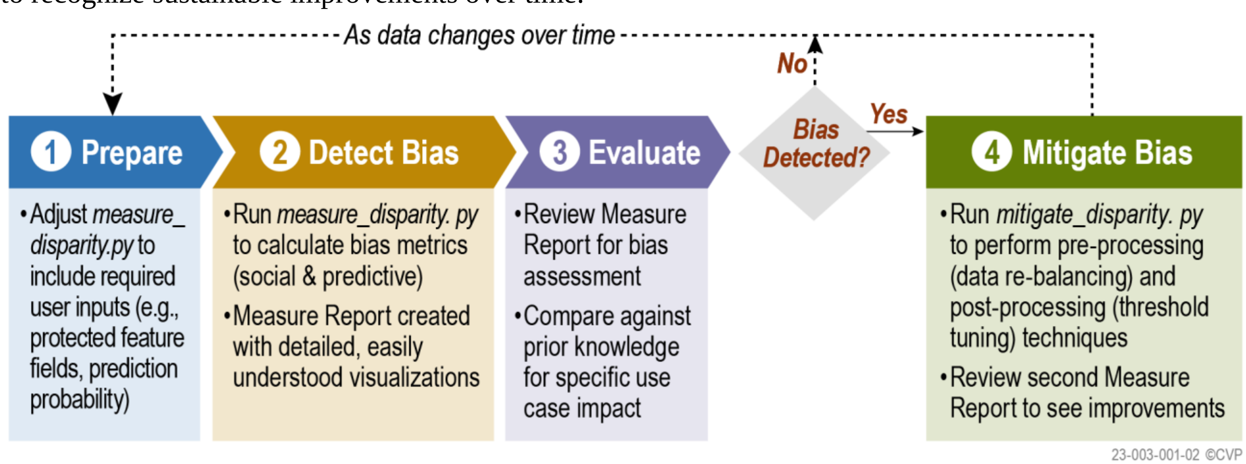

Figure 1 - Overview of Team CVP's Methodology to Identify, Mitigate and Monitor Bias in Healthcare Model

Our AI tool (Figure 1) aims to increase awareness of potential bias and facilitate stakeholder engagement and oversight by producing an automatically generated Measure Report on several measures like demographic parity and equalized opportunity. Instead of using predefined protected and reference classes, we analyze across entire demographic or protected features. We believe that the groups being discriminated against can change over time, and we do not want to introduce any bias by only examining certain classes. By examining *all* groups, we are able to track these changes and assess holistic disparity. The report dives deep into each protected feature (e.g., race, age, gender) to show where bias is detected. This allows a knowledgeable reviewer, well-informed on the topic, to quickly spot where the bias is and decide on a course of action.

When evidence of bias is detected, the reviewer has the option to run the mitigation step which consists of (1) rebalancing the train dataset to mitigate sampling bias and (2) tuning the model thresholds to improve equalized opportunity, demographic parity, precision, and the false positive rates directly, thus enhancing model performance across protected groups. While our measure script is ML model agnostic, our mitigation script is currently limited to LightGBM. The mitigation approach however could be easily expanded to other ML models or made model agnostic altogether in future iterations.

In this version of the tool we measure social and predictive bias metrics for bias measurement as seen in **Figure 2**. Equalized Opportunity and Demographic Parity were the main social bias metrics utilized. We measured predictive bias utilizing precision, which helps balance the two social bias metrics, to partially measure calibration and the Kolmogorov–Smirnov statistic to serve as the differential validity metric.

$$Demographic\,Parity = \frac{(TP + FP)}{(TP + FP + TN + FN)} \approx Equity$$

$$Equalized\,Opportunity = \frac{TP}{(TP + FN)} = Recall = TPR$$

$$Precision = \frac{TP}{(TP + FP)}$$

$$KS\,Statistic = TPR - FPR$$

Figure 2 – Measured Metric Formulas

This variety of metrics also allows us to see **retrospective bias** that may be due to sampling problems, as well as **prospective bias** where the model's predictions do not appear fair across protected classes. These metrics alone cannot explain *all* root causes of the bias but do provide visibility into a model's apparent bias and can help the user analyze fairness from both a global and local context. In future versions we hope to include other measures as they can spot other types of problems.

In **Figure** , we show a Chi-squared test to see if there was a statistically significant disparity present for a test dataset having to do with predicting hospital readmission. Statistical significance utilized at this point in the report is a succinct and familiar approach for both data scientists and doctors to show whether an apparent disparity exists across protected groups.

Demographic Parity

A fairness metric that is satisfied if the results of a model's classification are not dependent on a given sensitive attribute

Protected Group Type	Chi-Squared	P-Value	Fail
age	1288.284239	1.049026e-271	True
race	193.978868	7.399514e-41	True
gender	122.678730	1.639597e-28	True

Equalized Opportunity

Do the same proportion of each population receive positive outcomes?

Protected Group Type	Chi-Squared	P-Value	Fail
age	1215.524146	5.397750e-256	True
race	93.384419	2.513498e-19	True
gender	36.155989	1.821379e-09	True

Figure 3 – Two automated bias measures in our report

Our tool can be used to address several forms of **latent bias** by:

- Running it periodically on a data set as it evolves. By reviewing if the fairness metrics are changing over time, one can begin to see a latent bias or "model drift" may be occurring. Normally, we track this bias over time in a more explicit way by requiring a dataset include an "experiment ID" or other longitudinal tracking mechanism. To remain compliant with the challenge's input data requirements, we did not require this, though it would be relatively easy to add to our open-source solution.
- Using SMOTE, a data synthesis technique, to compensate for imbalances that may exist or develop. This makes our solution less susceptible to certain types of latent biases.

Our approach is based on a human-centered design, giving medical experts a broad diagnostic tool that supports *them* with transparency and easy to comprehend visuals, rather than *replacing them* with a black box. Instead of having the AI make an opaque recommendation that they likely will not trust, our approach makes medical experts, stakeholders, and the user community a part of the solution, improving trust, educating people about biases, and hopefully minimizing burnout.

This tool was inspired by our hands-on experience that bias can manifest in many obvious and not-obvious forms. We devised a solution which aims to support complex decision-making by giving people simple insights to make smart determinations. This also helps us identify our own inherent bias, like confirmation bias where we find what we expect. Bias is inevitable, but with effective tools, unwarranted bias (i.e., bias not inherent in the real world) can be minimized, and real-world bias can be better understood.

Value Proposition

Our tool has numerous advantages over typical, commercial tools for this purpose:

Feature	Benefit
Works *with* the medical provider to show many different measures of bias and performance.	Keeps the expert in the loop, rather than minimizing their opinion and knowledge on the ground.
Automatically calculates social bias from demographic parity ***and*** equalized opportunity as well as precision and Kolmogorov–Smirnov test results at the protected variable and class level.	Reviewers of the information can see multiple parallel evaluations of bias and performance and can compare them to one another to see bigger, systemic problems.
Shows summary of how each protected group (e.g., age) performed for a quick 30 second overview, but also pre-calculates all bias (across multiple dimensions) for individual protected groups.	Results are easy to consume both for executives and practitioners.
Visual comparison of several metrics across protected groups	Takes advantage of human ability to quickly spot differences or trends on graphs.
Automatically mitigates bias with SMOTE, random under-sampling of over-represented classes, and tuning of the model to maximize two different measures of bias	With minimal supervision, applies three different techniques to mitigate bias, without requiring substantial supervision or input.
Accepts data in CSV format	No proprietary file requirements.
Only dependencies before running are Python and several open-source libraries	No worries about software licensing.
Works with Windows, Mac OS, or Linux	Can be used anywhere from a laptop in someone's office to a high-end server in the cloud.

We mitigate social bias (that may include algorithmic bias) in several ways:

Feature	Description
Calculating the Mitigation-centric Bias Measurements	Evaluation metric calculated by taking the harmonic mean of: ➢ Demographic Parity ➢ Equalized Opportunity ➢ Precision

	➢ 1-False Positive Rate aka TNR (Replaces non-rate based KS statistic)
Rebalancing Transformation on Train SMOTE-TomekLinks	SMOTE is run separately across all combinations of protected features (e.g., subsets such as 35-yo Asian male) in the train data. The SMOTE function is used without consideration of imbalanced conditions, but combinations with very few observations were not synthesized. TomekLinks under sampling helps reduce majority label datapoints too similar to minority label ones. This dataframe is saved to help maintain an audit trail.
Pre- and Post-Hoc Imbalance Reporting	The imbalance test is run before and after the rebalancing transformation. The Imbalance Report outputs an HTML report that helps track how the data composition changes.
Population Parameter: Sample Weights	The sample weights are meant to represent the importance of each data point/patient. The model is fitted using these an input parameter. The metrics calculation also takes them as input so the metric scores are accordingly representative.
Post Processing Evaluation: Threshold Tuning	After the model predicts on the test data, the optimal thresholds are generated by first calculating the mitigation metrics at numerous threshold points for each group in each protected feature. The calculations benefit from rebalancing being performed on the predicted test data. The threshold that maximizes the harmonic mean of the four metrics is chosen as that group's threshold value. We add these thresholds to the unbalanced predicted test data as three threshold columns for each protected feature. Then the final threshold for each patient is calculated by averaging these three columns. By incorporating rebalancing and averaging, the thresholds should be less overfit.

To improve on these strategies, we would explore:
1. Alternate rebalancing methods - We used SMOTE-TomekLinks as it proved effective for our data. For other data sets, a different technique for far smaller minority label classes could prove more effective.
2. Alternative threshold tuning methods - Instead of maximizing the metrics' harmonic mean, could experiment with picking a single threshold that minimizes their Chi-Squared values
3. Add weighting to metrics - Some use cases may need to prioritize certain kinds of fairness more than others. Weights would give this flexibility to the user.
4. Incorporate metrics in custom loss functions – instead of pre- and post-processing, do in process incorporation of the metrics. Easier to implement with neural network architectures.
5. Add other metrics – We have stuck to more basic metrics for our initial iteration to prove that the framework works. Other metrics could be developed and added to target other forms of bias.

Healthcare Scenario

As a general-purpose tool that identifies and mitigates biases found in ML models, we believe this solution should generally be used *before* a direct patient-provider encounter. Healthcare data analysts, policy makers, and medical review panels could easily use it to analyze results of a study involving ML or use it to decide on best practices that inform how medicine is used in day-to-day care based on a promising but untested model.

To use our tool to evaluate and improve an ML model* used in the prediction of an adverse event or diagnosis of a disease or condition, one should:

** This challenge and our tool assume you already have a trained LightGBM model from which you have used to generate predictions for the test set. If the model is not LightGBM, steps 1-5 below are still functional*

1. Save the predicted test set as CSV.
2. Run the measurement program which will produce a report showing how the model performed across and within the protected features.
3. At a minimum, create a review group of a data scientist (ideally the one who made the model) and a medical practitioner who is familiar with treating or diagnosing the condition.

4. Have this group examine the report to determine what was found to be biased and where they think the bias is coming from. The report will highlight areas of *concern*, but no AI will have visibility into sampling problems that may have impacted the training data or domain knowledge as to whether that disparity is normal. For example, if the model was for the diagnosis of sickle cell anemia, Alzheimer's, Parkinson's or another condition that has uneven impact across protected features (e.g., age, race), this may be *expected* and not a large concern.
5. The review team should decide if the bias report has found apparent social/predictive biases or whether the issue may be systemic, sampling, or another type of bias where a new training dataset would be more appropriate.
6. If team determine the issue is due to model bias and the model algorithm is not LightGBM, proceed to run a LightGBM version of the model. Rerun measure for it the mitigation tool. This will attempt to lower the apparent rate of disparity in the model and yield a fairer outcome.

When used appropriately, CVP's tool meets the challenge objectives to identify and minimize inadvertent amplification and perpetuation of social biases in AI/ML algorithms used as clinical decision support. It can support users in determining viability of the ML model, and upon implementation, can help clinicians determine systemic diagnostic error either in explicit diagnostic weaknesses, data quality issues, or other influential factor that may inhibit the doctor from providing optimal care across groups. By helping to address all of these and highlight disparities with minimal extra effort by the clinician, our tool will improve patient outcomes, *especially for those traditionally underrepresented or underserved parts of the patient population.*

This tool can also be used to cover other clinical decision scenarios if that logic of the data meets input data requirements. Please see the **Methodology Overview** section for how our tool addresses latent bias.

Operational Requirements

To meet the requirements of the challenge, make our system easy to scale in multiple scenarios, and to provide a technology-agnostic platform, we structured this version as command line program in Python. This allows it to be run on older, low-end PCs (e.g., for a researcher in an under-resourced country) or invoked thousands of times on a back-end server for bulk processing without the need for a human to click. Our solution also requires no license, no Internet connection, no proprietary dependencies, and no vendor-specific data.

The only technical requirements before running are a Python environment setup on the computer and a Git client to retrieve the code, though you could download our code from GitHub's website if the latter is problematic. We recommend Anaconda which has setup directions here. CPU, RAM, and storage requirements are dependent on the size of the dataset you wish to measure and mitigate bias on. You should generally have RAM that is greater than or equal to the dataset.

Once Python is available and an environment is activated:

1. Run "Git clone https://github.com/cvp-bias-detection-healthcare/cvp-bias-detection-healthcare.github.io"
2. Run "pip install requirements.txt"

To use the measure_disparity.py file:
1. Open "measure_disparity.py", located in the scripts folder, in a text editor.
2. The first 15 lines of code are used for storing metadata about your dataset of interest. Please modify them as needed to locate the data, tag the protected features, and label which columns have the probabilities, the true labels and sample weights. Save your changes.

3. Run “python measure_disparity.py”. An HTML report (“measure_rerport.html”) will be automatically generated in the “reports” folder
4. Double click it or transfer to a computer with an internet connected web browser to review it.

To use the mitigate_disparity.py file:

1. As required by the challenge, the mitigate script returns a Python object/class with fit(), transform(), and predict() methods in addition to a measure() method for report generation. Read the README inline comments for the class creation instance and class methods to understand the arguments and parameters they need as input
2. The “mitigate_disparity.py” file needs to be imported as a module in another notebook or a .py script.
3. For ease of use, we have included a “run_mitigate.ipynb” sample notebook which is setup to use the object and its methods to perform a full mitigation
4. You may run this notebook as is with our sample data file (“diabetes_data”) or copy its contents for use with your own dataset
5. For the latter, you will need to define your train and test pandas dataframes
6. We recommend predefining all of the critical fields (like in the measure script) that need be passed into the class and methods
7. For this script, the LightGBM model parameters dictionary is another critical field, and you can either keep it as is or modify as you see fit for your dataset
8. Once all required parameter inputs are defined, initialize the Mitigator object and call its methods in this order: transform(), fit(), predict(), measure(). These together will generate the “imbalance_report.html” and the “mitigate_report.html” (same format analysis as “measure_report.html”) in the reports folder. It also outputs the transformed train dataset, the predicted test dataset, and the threshold tuned predicted test dataset to help build an audit trail.

For more detailed information on how the mitigation process works, please reference the code comments in the GitHub code.

We have also created a Google Colab notebook that contains the retrieval of the source code, setup of the environment, and running of our solution with example data. It is available at this link: https://colab.research.google.com/drive/1KP64rF6k-DK5F83OYkTwWidvuVzPKhmo?usp=sharing

Architectural Design

To make our solution as easy to use as possible in multiple environments (and because we are big fans of open review and collaboration), our solution consists of only Python files with easy to install dependencies. By following the directions above, the tool’s architectural capabilities include the following:

- Run locally on a laptop or desktop
- Run on physical or virtual server in the cloud
- Be loaded into a service like AWS Batch (Docker container) or AWS Lambda (Firecracker VM) which can run thousands of Python programs in parallel. Datasets could be passed in via AWS API Gateway and reports delivered as HTML hosted S3 or CloudFront.

If deployed in a cloud server such as AWS or Azure, this solution can supplement a larger setup that harnesses existing AWS and Azure AI bias fairness and implementation tools, dashboards, and scorecards and other tools capable of reviewing many different models to provide output back to reviewers.

BSD 3 License Acknowledgements

This solution has been published with a standard 3 clause BSD license.

Sustainability Plan

The tool can be used in the future without significant upkeep, but care should be taken to ensure a charter exists that defines the goals of the ML model, where the training data comes from, how the model was built, any constraints that need to be observed, and how it will be reviewed. An optimal solution to sustain the use of an ML model in the real world should include integration with existing administrative, security, systems, clinical, and billing processes, as well as a regular review of it as new research becomes available. For security purposes at a Federal agency, we recommend consulting the NIST AI Risk Management Framework (AI RMF), as it covers many of these issues in great detail.

We envision a multi-disciplinary team helping on the sustainment of machine learning (with support from our tool):

Role	Responsibility
Champion	Executive sponsor for the program, ensures it meets organizational priorities and is adequately resourced
Data Scientist	Ensures the model was designed, trained, and tested according to best practices. Runs the bias measurement and mitigation tools and modifies the technical approach based on team feedback
Informaticist	(If clinical data is involved) Assembles the training data set. The Informaticist is responsible for ensuring the data quality meets the needs defined, data is extracted accurately, coded properly, documented, and that there are no biases in sampling.
Subject Matter Expert	Depending on the use case for the model being examined, the subject matter expert is well-versed in how the ML model being examined should function and can easily spot problems. If this model is for the treatment or diagnosis of a patient, it might be a medical doctor. If this model predicts who is likely to pay a bill, it might be someone from the Accounts Receivable team.
IT / Cybersecurity	Reviews the solution for compliance with the proper technology standards (including security). Implements and integrates the solution on a computer system (if relevant).
Data Steward / Ombudsman	Reviews the data being used and the goal of the ML model for ethical, legal, and compliance. Ensures data is only being used for proper purposes and only shared with those who have a need to know.
Business Owner / Program Mgr.	The program manager whose daily work is affected by the AI solution. They are an integral part of understanding the problem, evaluation the solution, and change management for putting the solution in.

To ensure long-term sustainability, our broad Measure Report can (where relevant) allow users to see both potential retrospective bias that may be due to sampling problems, as well as prospective bias where the model's predictions do not appear fair across protected classes. In some cases, it may not be clear where the bias is coming from, in which case we recommend reviewing the more detailed visualizations that display results for the individual protected classes.

Generalizability Plan

Because we wanted to maximize the value offered by our tool and because we know bias is not unique to one industry, our tool will assess and mitigate bias in any machine learning model where protected classes are used. We have tested it with datasets from fraud detection, law enforcement, genomics, banking, and hospital readmission. It can be used for any medical discipline that creates predictions (of disease progression, diagnosis, treatment, and more) if there is ground truth value to compare to an actual value. We did not include out of the box support for unstructured data (e.g., text, imaging) as we felt that it was not straightforward to include given the challenge results for an input data file.

As mentioned in previous sections, we purposefully designed our solution to be unencumbered of licenses and runnable on almost any modern computer. To view the output of the measurement report, one only needs a web browser which is included in all modern smartphones and personal computers. We believe the goals of this challenge are not met if only those organizations with access to exotic hardware and paying costly license fees are allowed to measure and mitigate bias.

We also strive to make this “human friendly”, producing a report that is easy to read for those who are not experts in machine learning. In some cases, the measurement step of the process may be all that is required to notice a problem with an ML model, and that is perfectly acceptable to our team.

Patients and Patient Advocacy Group Information Sharing

The information from the report can be shared with the caregiver, physicians, and the public through online published reports (it is HTML already!), and with any patient whose case involves an ML model. Full transparency and visibility can help patients be better educated about prior limitations in their diagnosis history for them to be proactive in resolving potential issues with their circumstance.

We also recommend the user to provide transparency in all reporting and evaluation with patient advocacy groups to aid them in the identification of healthcare barriers to specific patient populations and identify systematic regions where they are high levels of disparity. This is an opportunity to engage stakeholders who can benefit from collaborating in this process, and feedback will further advance the evolution of the technology.

Implementation Requirements

Any implementation plan must consider the problem being solved, the relative costs and benefits of the entire solution being proposed, and the potential impact of the ML model. As our solution is a general-purpose, model agnostic tool for the monitoring of any binary classification problem, the implementation plan should be tailored to business, not our supporting bias measurement and mitigation procedure.

As an example, the implementation plan for an ML model that helps with appointment scheduling for cosmetic procedures will be very different from that for a model that predicts a deadly disease (high cost of false positive) and whose treatment is potentially harmful to a patient (high cost of a false negative).

For implementation of a machine learning solution, we recommend following the industry-standard CRISP-DM process shown at **Figure** . Please see our section on sustainability for a discussion of the human resources (roles) necessary to implement and sustain this solution. At a high-level, the process involves:

1. **Business understanding** – A subject matter expert and project champion should work together to document the problem being solved and what solutions might be.

2. **Data understanding** – a subject matter expert (e.g., doctor), data scientist, informaticist (encouraged if needed), and IT owner of the relevant data sources should analyze what data is available to support the problem and how the quality is. At this point, the team may realize they need to pause and collect more or better data. A data steward should also examine the project for compliance with
3. **Data preparation** – the data scientist and informaticist collaborate to ensure the data is of high quality, consistent, meets the stated specifications, and is in a format suitable for use as a training data set from an ML model
4. **Modeling** – the data scientist trains and tests a model until they have one that may meet the objectives laid out by the champion and charter.
5. **Evaluation** – the data scientist runs our measurement tool and uses other standard techniques for assessing the model performance. Here, they collaborate with the subject matter expert, data steward, and IT personnel to see if the solution is understandable, feasible, and is not unreasonably biased. If the model seems to be useful but has more bias than desired, they should run our mitigation tool to ensure a fairer outcome in the process.
6. **Deployment** – If all parties are satisfied with the results, the team should work to deploy or put the solution into production usage. Alternatively, if the solution seems promising but is not totally ready, a pivot may be required, and the process should go back to step 2 for a new iteration.

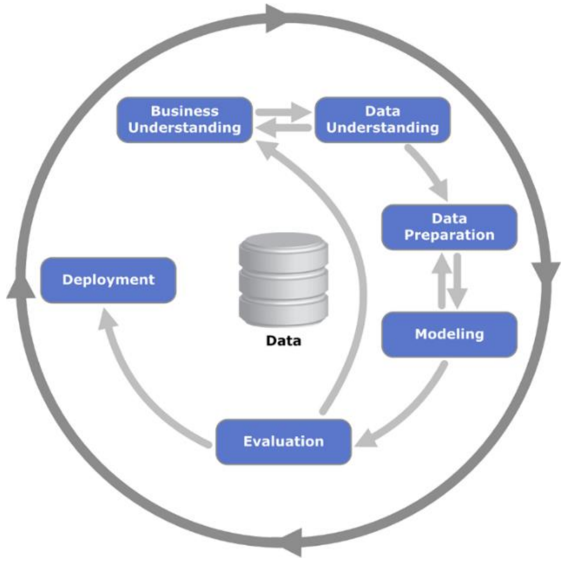

Figure 4 - We recommend CRISP-DM to guide implementation.

Scaling to other locations or centralizing implementation should mirror the above process but involve more stakeholder engagement and cost benefit analyses to ensure an effective implementation and ongoing operations while meeting compliance and security requirements. While scaling to different teams and areas, they should be re-evaluating the performance and viability of the solution, while continuing to assess the bias of the model along with any other technical, security, operational, or business changes required. This review is necessary, because a single machine learning model may not be efficient or effective for all use cases, developments, systems, groups, or geographic areas, causing one to overlook areas of bias or reducing the impact of the tool's ability to detect bias. This process, although lengthy, will ensure support by all parties, improve the ultimate patient outcome, and reduce unintended bias that may develop.

There are potential limitations of directly combatting social and algorithmic bias before full field testing, as it may limit performance by being stuck in "local minima" or sub-optimal outcomes. This tool was not developed with any information around the AI it is meant to supplement and, outside the mitigation process with one model we create, does not communicate the bias found back to the original model. This means that this tool is useful in helping the user make the determination as to when the AI is mature enough for production testing, where we believe it is most important to determine overall outcomes and capabilities.

Lessons Learned

We have participated in other government challenges, but we found this project one of the most interesting because it fuses technology and machine learning with issues of social justice, fairness, and equity. We have previous experience with some of the metrics used for bias assessment here, but also learned a lot by watching the various webinars. It was refreshing to see how much work has already been done.

Through working on this project, meeting with potential stakeholders (e.g., medical practitioner colleagues), and reviewing the results, we learned:

- Everyone wants to make sure that we have unbiased models, but it is often impossible to eliminate all perceived bias in a process that involves humans. We can only report on different forms we see and let experts decide which is or is not an issue.
- Bias means many things to different people, and it is difficult to have a unitary measure of it. We originally hoped to have a single measure we optimized for, but ended up realizing that this would miss or conceal certain problems, leaving users with incomplete information on what was happening.
- Medical professionals are familiar with concepts like “false positives” from diagnostic testing but thinking about what many of these measures like demographic parity mean is **not** something most members of the public are used to. We spent a lot of time explaining them to our internal reviewers who felt it was not intuitive and ended up embedding explanations into our final product to make this usable by people besides data scientists.

In the medium term, we plan to take our tool for this challenge and enhance it in the following ways to make it more useful in improving health equity to a larger group:

- Expand beyond the requirements of the challenge and let the bias measurement also assess training datasets. It is often easier and more productive to point out the training data is biased vs. the model.
- Include running of our tool as part of our existing ML training pipeline, so that it is an integral part of any model we do.
- Create an optional graphical user interface for the tools and ways to distribute as an executable (not a .py file), so less technical audiences can use it and scale even easier. While Python is great for data science and machine learning, it is not the most popular tool for building interactive desktop applications.
- Test and write-up material like blog posts or papers showing how our tool could be used in a variety of clinical and non-clinical settings.
- Look into ways the input dataset can be modified to also work with imaging. We have done previous work in this area as featured here.