Solving the inner optimization problem

Projected Gradient Descent: We use a first-order method with iterates
\[ \delta^{(k+1)} = \delta^{(k)} - \alpha^{(k)} \nabla_x J(\delta^{(k)}) \]
where \( P \) is a projection operator such that the constraint, \( |\delta| \leq r \), is satisfied.

Trust Region Subproblem: We use a second-order Taylor approximation of our loss function and solve
\[ \delta^{(k)} = -\left( \nabla_x J(\delta^{(k)}) + \lambda I \right)^{-1} \nabla_x J(\delta^{(k)}) \]
where \( \lambda > 0 \) is chosen such that the constraint is satisfied.

Error in trust region subproblem solution

For binary classification, we use a logistic regression loss function with a sigmoid activation function \( \sigma(z) = \frac{1}{1 + e^{-z}} \), so we have the following inner optimization problem:
\[
\max_{|\delta| \leq r} \left( -y \ln \sigma(w^T(x + \delta x) + b) - (1 - y) \ln (1 - \sigma(w^T(x + \delta x) + b)) \right)
\]
When we use a second-order approximation (RHS) instead of the true loss function (LHS), we end up solving two slightly different problems with differing terms
\[
\sigma(w^T(x + \delta x) + b) \approx \sigma(w^T x + b) + \sigma'(w^T x + b) - \sigma'(w^T x + b) \delta x
\]
Taylor expanding the LHS, we can obtain the following:
\[
\sigma(w^T(x + \delta x) + b) \approx \sigma(w^T x + b) + \sigma'(w^T x + b) \delta x + \frac{1}{2} \sigma''(w^T x + b) \delta x^2 + \ldots
\]
The error of solving the trust region subproblem comes from truncating the higher-order terms in the Taylor expansion. The error from truncating the quadratic term depends on the magnitude \( \|\theta\| \) \( \leq 0.1 \) and the \( \delta x \) for which we solved.

Adversarial training

Our research investigates whether implementing robust optimization - so that two nearby points are more likely to be classified similarly - would improve fairness. Robust optimization takes into consideration at a radius \( r \) around a data point (see fig. 1).

Hiring (synthetic data)

- \( Y \): should be hired or not hired
  - \( Y \): predicted to be hired or not hired
  - \( s \): sensitive attribute \( A \) or \( B \)
  - unfairness: Bs shifted up and right \( r \)
  - As shifted down and left \( -r \)

All red individuals in the blue region who would be hired in error are Bs, while all blue individuals in the red region who would be incorrectly not hired are As.

Which method is fairer?

- Using the trust region subproblem method significantly improves efficiency: computing second-order information using HessQuik outperforms first-order PGD across all perturbation radii on three different data sets.
- Robustness can improve fairness, but potentially at the cost of accuracy. Fairness improves as the perturbation radius increases but accuracy decreases in both training and testing data as the radius increases.
- If using robust training with a certain radius improves fairness, it appears to improve fairness by larger margins compared to random perturbation; solving the optimization problem well is worthwhile.

Conclusions

- Thank you to Dr. Elizabeth Newman, our mentor, for her guidance and support.
- This work is supported in part by the US NSF award DMS-2051019.

Acknowledgements

References


Fairness metrics

Let \( Y \) be the classifier’s prediction, \( Y \) be the true class, and \( s \) be a binary sensitive attribute. How do we measure the fairness of a binary classifier with respect to \( s \)?

- Independence: \( \text{P}(Y = 1 | s = 0) = \text{P}(Y = 1 | s = 1) \)
- Separation: \( \text{P}(Y = 1 | Y = 1, s = 0) = \text{P}(Y = 1 | Y = 1, s = 1) \)
- Sufficiency: \( \text{P}(Y = 1 | Y = 1, s = 0) + \text{P}(Y = 1 | Y = 1, s = 1) \)

Which method is faster?

- Unlike in PGD, where we may have to do many gradient computations, with the second-order method, we at worst solve a system of linear equations and use a bisection method. The trust region method has proven to be the more efficient method.