Reducing Gender Bias in Credit Card Approval Predictions

Petros Markopoulos pm10@williams.edu
Computer Science

1. Introduction

It sounds reasonable at first that decisions made by algorithms would not suffer from the subjectivity and biases that human decision-makers often possess. It has been found, however, that that is not the case, and, in fact, machine learning algorithms often perpetuate the biases that already exists in the world, because they are reflected in the data that the models are trained on (Angwin et al., 2016; Dustin, 2018).

Access to credit is important for people's quality of life, and one way many people use credit is through credit cards. Getting a credit card is not only helpful for consumer spending, but it is often the way people start building a credit history which is instrumental in getting a mortgage or even many jobs. Like any other form of credit, one has to apply to get a credit card and be evaluated for their creditworthiness. Given that nowadays people apply for credit cards online and get an approval or rejection within minutes, it is certain that some automated process is used to evaluate the applications, and machine learning models seem like good candidates for doing that. Thus, it is important to ensure that the models that determine people's access to credit do so fairly. This has not always been the case (e.g. race and gender have often affected and still affect access to credit), so deliberate action is needed to train unbiased models.

In this work, I attempt to tackle this problem and train machine learning models on a cleaned version of University of California Irvine's (UCI) Credit Approval Dataset (Dua and Graff, 2017) available on Kaggle (Cortinhas, 2022) to predict whether an applicant would be approved for credit. The dataset includes a column for applicant gender, which is a sensitive feature that should not be used in predictions. My hypothesis is that we can achieve a good tradeoff between accuracy and bias (not inductive bias, but rather gender bias). In other words, after training a model with the only goal being to maximize accuracy in predicting approvals to establish a baseline, the goal is to create a model with accuracy close to the first model, but which would have lower bias. Using an adversarial training technique inspired by Beutel et al. (2017), I trained a model which achieved good accuracy, but only moderately lower bias than my baseline.

2. Preliminaries

2.1 Neural Networks

Neural networks are structured in layers of neurons Rosenblatt (1958). A neuron receives inputs and applies a nonlinear "activation" function to their sum to produce its output. The

output of a neuron on layer ℓ is weighed by some parameter and fed to all the neurons in layer $\ell + 1$ as an input. The first layer receives the inputs X, and the last produces the outputs. Suitable activation functions are the sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$ and $ReLU(x) = \max(0, x)$.

Thus, the network produces a prediction for an outcome variable Y as a nonlinear combination of the learned neuron input weight parameters θ and the input features X. If some features are not numerical (e.g. categorical) they need to be transformed into some format that is amenable to machine learning. The transformation I used is OneHot encoding, where each unique value of the categorical attribute is transformed as its own feature, with a value of 1 if the data point had that value and 0 otherwise.

The figure below shows a sample neural network.

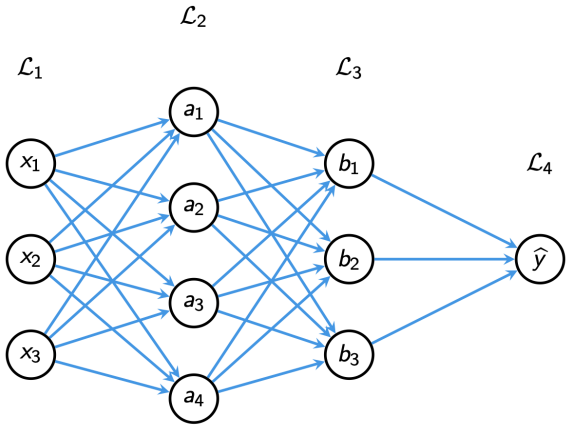

Figure 1: A neural network with 4 layers (from class slides)

Suppose the network above uses σ as the activation function; then the computation from input to output is as follows:

$$A = \sigma(X[\theta]^{\mathcal{L}_1 \to \mathcal{L}_2})$$
$$B = \sigma(A[\theta]^{\mathcal{L}_2 \to \mathcal{L}_3})$$
$$\hat{y} = \sigma(B[\theta]^{\mathcal{L}_3 \to \mathcal{L}_4})$$

The parameters θ (corresponding to the weights on the edges in the figure) can be learned by minimizing a suitable loss function L using gradient descent. A common loss function for binary classification tasks, and the one that was used in this project, is the negative-log-likelihood $L(\theta) = -(\sum_i y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i))$. Additionally, instead of gradient descent, I used adam (Kingma and Ba, 2014).

As the number of parameters in a neural network becomes large, the network becomes prone to overfitting; i.e. adjusting the parameters to perform really well on the training set, but failing to generalize to the validation and testing sets. Thus it becomes useful to employ regularization techniques. I chose to apply the L2 regularization penalty Hoerl and Kennard (2000) in conjunction with the loss function, so rather than minimizing $L(\theta)$ the training aims to minimize $L(\theta) + \lambda \sum_i \theta_i^2$. λ is a hyper-parameter, and I tuned it by

training multiple models with different values for λ and picking the best-performing one after validation.

2.2 Adversarial Learning

Another technique employed in this project is adversarial learning, pioneered by Goodfellow et al. (2014). In adversarial learning, we train two models rather than one; I'll refer to them as the model and the adversary. The adversary expects input of the same form as the model's output, and tries to predict whether the input came from the model or not. The model is trained to fool the adversary.

Beutel et al. (2017) give a variation on the concept. Rather than training two networks, one can train a single network with two neurons (heads) in the output layer. One of the outputs gives predictions for some output, while the other (adversarial) gives predictions for a sensitive feature. The goal is to train the network in such a way that the first head achieves good accuracy, while the adversarial head performs poorly.

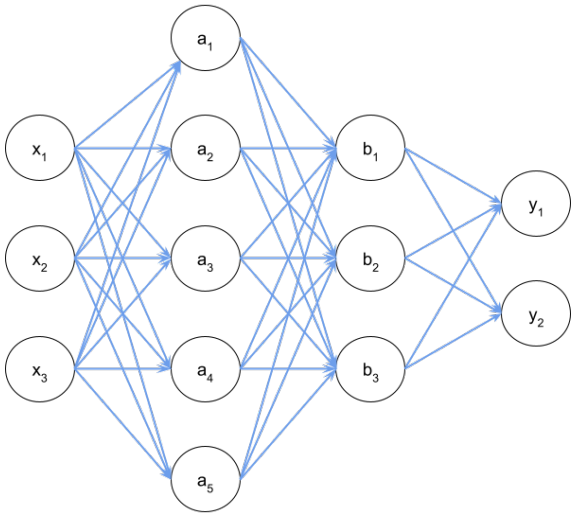

Figure 2: A neural network with two output neurons. The network can be trained to give accurate predictions from y_1 but inaccurate predictions from y_2

2.3 Fairness

The measures we use to determine fairness are based on the ideas of *demographic parity* and *equality of opportunity* (Hardt et al., 2016). We have the following definitions:

$$ProbTrue_g = \frac{(\#\text{of positive predictions where Gender} = g)}{\#\text{of data points where Gender} = g}$$
$$ParityGap = |ProbTrue_m - ProbTrue_f|$$

$$ProbCorrect_{1,g} = \frac{(\#\text{ of true positive predictions where Gender} = g)}{(\#\text{ of positive data points})}$$
$$ProbCorrect_{0,g} = \frac{(\#\text{ of true negative predictions where Gender} = g)}{(\#\text{ of negative data points})}$$
$$EqualityGap_y = |ProbCorrect_{y,m} - ProbCorrect_{y,f}|$$

Intuitively, $ParityGap$ represents the disparity of the rates at which positive results are given for the two genders, $EqualityGap_1$ gives the difference in the rate at which the two genders are classified correctly when they were approved, and, similarly, $EqualityGap_0$ gives the same metric but for the cases when they were not approved. The lower these metrics, the more fair the model is. Each of the above metrics takes value in the range $[0, 1]$ with 0 indicating complete equality.

3. Data

As mentioned earlier, the dataset used is a cleaned-up version of the Credit Approval Dataset from the UCI Machine Learning Repository (Dua and Graff, 2017). It contains 15 attributes (Gender, Age, Debt, Married, BankCustomer, Industry, Ethnicity, YearsEmployed, PriorDefault, Employed, CreditScore, DriversLicense, Citizen, ZipCode, Income) as well as the outcome of the application (in the Approved column). There are 690 data points.

Industry, Ethnicity, and Citizen are all categorical, standing for the industry of one's employment, their ethnicity, and how they came to be a citizen respectively. There are 14 values in the Industry column, 5 in Ethnicity, and 3 in Citizen. To use the features in the model I employed OneHot encoding. There were 170 distinct values (i.e. 1 value for every 4.06 data points) in the ZipCode column, and thus I did not use it in the model, because that could lead to overfitting. After this pre-processing, there are 33 columns in the data usable in the model. (see Figures 3, 4)

The data is equally split divided into training, validation, and test sets utilizing the `train_test_split` method from `sklearn.model_selection` and then standardized so that the training data has mean 0 and standard deviation 1.

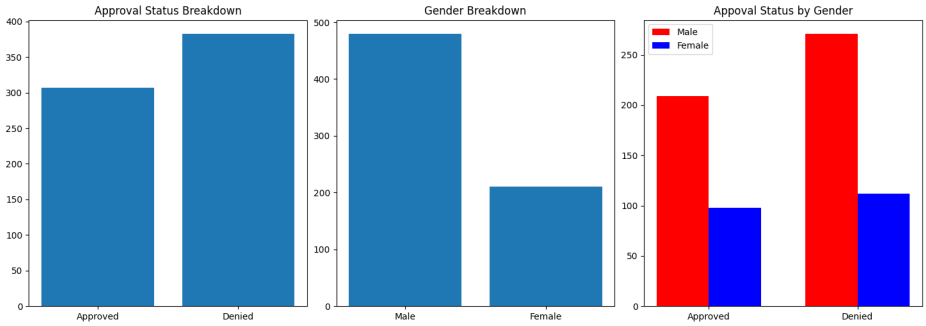

Figure 3: Breakdown of the data set in terms of approval status and gender

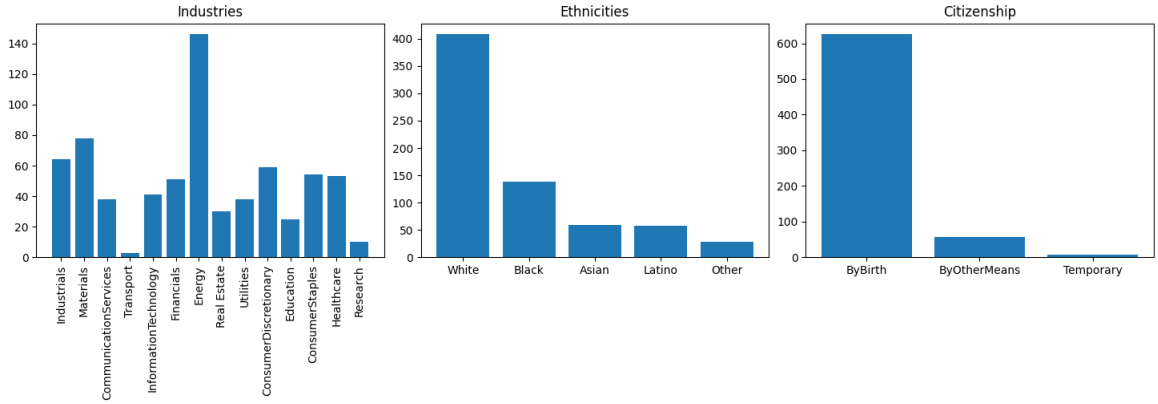

Figure 4: Breakdown of the data by category (from left to right: by industry, ethnicity, and method of acquiring citizenship)

4. Training And Validation Of Models

4.1 Baseline

In total, I trained 67 models utilizing the `MLPClassifier` class from `sklearn`: a simple logistic regression, 30 models with one very wide hidden layer, and 36 models with three hidden layers. All the models used the adam optimizer mentioned earlier and the activation function for the hidden layers was always $ReLU$. The logistic regression model had no regularization and the learning rate for its training was 0.001. The rest of the models were trained with $L2$ regularization and learning rate 0.01 and were trained until convergence. The regularization hyper-parameters tested were all integer values between 4 and 9 (inclusive). For the shallow models, the width of the hidden layer was one of $128, 256, 512, 600, 700$. In the deep models the first hidden layer had width $256, 512$, or 600, the second had width 256, or 512 and the third layer had a fixed width of 128.

The baseline model was selected solely based on accuracy, and only the selected model was evaluated for fairness. The above models were all first trained, on the training set, and the one with the highest prediction accuracy on the validation set was selected as the baseline. This was a shallow model with a 128 neuron hidden layer and the regularization parameter set to 7. The model achieved 82.9% accuracy on the test set, and it had $ParityGap = 0.057$, $EqualityGap_0 = 0.108$, and $EqualityGap_1 = 0.039$.

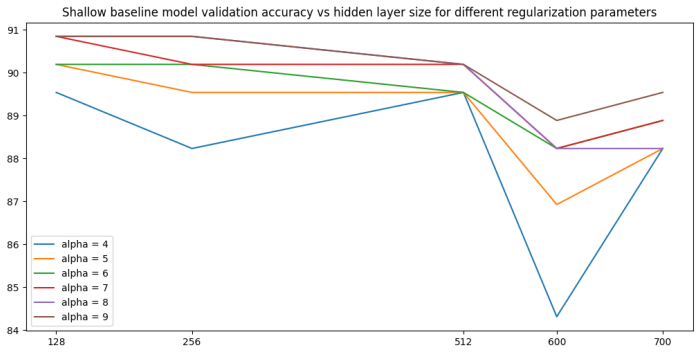

Figure 5: Validation Accuracy of Shallow Models

4.2 De-biased Model

I used adversarial training inspired by Beutel et al.. In that paper, adversarial training is achieved through the use of negative gradients. Rather than the gradient of the loss function for the adversarial head, gradient descent is given the negative gradient, in order to move the model away from the minimum of the loss function. In this project I used a simpler idea: for every training data point, I created an identical data point with the gender attribute changed. The models were trained on this duplicated training data set.

I built 34 models using `tensorflow.keras` since it allows the flexibility necessary to have multiple outputs from a network. I fit them and used the validation data set to tune the L2 regularization parameter, and the architecture of the network. The possible network architectures were identical to the baseline models described in the previous section (except for the input and output layers, since the former lacked the input for Gender and the latter had two output neurons rather than one), the activation function for the hidden layers was $ReLU$, and the optimizer was adam. The L2 regularization parameters took integer values between 0 and 2 (inclusive).

I used two different schemes to pick the best model: the first did not utilize the fairness metrics directly, while the second did. Ultimately the two methods yield the same final model.

The first method was to take the model which maximized the expression

$$Accuracy_{approval} - |0.5 - Accuracy_{gender}|$$

Since the models are trained to predict credit approvals well but gender badly, this is a reasonable measure of success (as, ideally the accuracy for the gender predictions would be close to 50%).

The other method was to select the model which maximized the expression

$$Accuracy_{approval} - ParityGap - \frac{1}{2} \cdot EqualityGap_0 - \frac{1}{2} \cdot EqualityGap_1$$

Because high values in the fairness metrics indicate high gender bias, they are subtracted in order to select a model that balances high accuracy with low parity gap and equality gaps.

The best model turned out to be the one with no hidden layers at all. It achieved 82.89% accuracy with $ParityGap = 0.032$, $EqualityGap_0 = 0.085$, and $EqualityGap_1 = 0.01$

5. Results

Recall that the fairness metrics in the baseline were

$$ParityGap = 0.057 \qquad EqualityGap_0 = 0.108 \qquad EqualityGap_1 = 0.039$$

These are already close to the desired value of 0, which means that the baseline model was not that biased to begin with. Comparing them to the fairness metrics of the de-biased model, which were

$$ParityGap = 0.032 \qquad EqualityGap_0 = 0.085 \qquad EqualityGap_1 = 0.01$$

, we see a small decrease of about 0.02 in each of them.

While, in absolute terms, the reduction is small, relative to the baseline metrics it represents improvement on the order of more than 20% for each one. I think then that the small magnitude of the improvement is quite possibly less attributable to the adversarial training technique being ineffective and more to the fact that even the baseline model was not very biased (likely because the data set includes features like Income, Credit Score, Prior default etc. that are much better predictors of credit approval than Gender). It is difficult to reduce the effect of a feature if it is already minimally utilized in predictions.

In terms of accuracy the results are very positive. The baseline model achieved 82.9% accuracy on the test data set and the de-biased model achieved 82.89% accuracy on its predictions of credit application approval (the prediction for the gender is only used for training purposes and hence it is irrelevant here). The adversarial training, then, came at negligible cost to the accuracy of the model, so the trade off with fairness remains very good.

It is worth noting, however, that the de-biased model has different architecture from the baseline. If simplifying or making the model more complex is necessary for achieving this trade-off this raises questions on how this could affect underfitting or overfitting in other problems where the underlying distribution is more complex or simpler respectively. It would be valuable to examine this problem with the network architecture held constant.

6. Ablation Study

Since the improvements from adversarial training were small, I tried to reproduce them using a simple model which just did not receive the Gender feature as an input. I trained exactly the same models as described in Section 4.1, with the only difference being that each network had one less neuron in the input layer.

Since these models do not predict the gender label, the only way to take fairness into account is to use the fairness metrics directly. Thus, to select the best architecture and hyperparameter settings I chose the model that minimized the second expression from

Section 4.2:

$$Accuracy - ParityGap - \frac{1}{2} \cdot EqualityGap_0 - \frac{1}{2} \cdot EqualityGap_1$$

The selected model had a single hidden layer with 600 neurons and the $L2$ regularization hyperparameter was set to 5. It achieved 82.89% accuracy on the test data set and the fairness metrics were $ParityGap = 0.057, EqualityGap_0 = 0.108$, and $EqualityGap_1 = 0.039$, which is exactly the same as the performance of the baseline model.

This is not too surprising since, given the fairness metrics of the baseline, it seemed like the use of the Gender feature for predictions was low. What this indicates is that it must have been close to zero. It also indicates, however, that the adversarial training was potentially able to reduce the indirect impact of Gender on predictions (through correlation with other features), since the simple removal of the feature was not able to reproduce even the modest improvement that adversarial training achieved.

7. Discussion and Conclusion

This experiment has yielded mixed results. On the positive side, it demonstrates that adversarial training such as what is described in Beutel et al. (2017) and in Section 4.2 has the potential to produce models that are fairer under the widely used metrics of demographic parity and equality of outcomes without sacrificing much in the way of accuracy. On the negative side, for this task, the benefit that this technique introduced was not very significant. This could be a result of the specific data and task, or it could reflect on the potential of this kind of training altogether.

References

Julia Angwin, Jeff Larson, Lauren Kirchner, and Surya Mattu. Machine bias, 5 2016. URL https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.

Alex Beutel, Jilin Chen, Zhe Zhao, and Ed H. Chi. Data decisions and theoretical implications when adversarially learning fair representations, 2017. URL https://arxiv.org/abs/1707.00075.

Samuel Cortinhas. Credit card approvals (clean data). https://www.kaggle.com/datasets/samuelcortinhas/credit-card-approval-clean-data?select=clean_dataset.csv, 4 2022.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Jeffrey Dustin. Amazon scraps secret ai recruiting tool that showed bias against women, 10 2018. URL https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. URL https://arxiv.org/abs/1406.2661.

Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning, 2016. URL https://arxiv.org/abs/1610.02413.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 42(1):80–86, 2000. ISSN 00401706. URL http://www.jstor.org/stable/1271436.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65 6:386–408, 1958.