CS 281 Homework 1

Instructors: Carlos Guestrin

{khkim,lxuechen}@cs.stanford.edu

Available: 03/31/2022; Due: 23:59 PST, 04/12/2022

Introduction

In this assignment you will explore evaluating various fairness metrics on the Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) dataset. The COMPAS algorithm takes in a wide range of 137 features including age, gender, and criminal history of an individual as inputs and outputs a COMPAS score ranging from 1 (being lowest risk) to 10 (being highest risk) which represents the risk of recidivism. COMPAS has been used by the U.S. states of New York, Wisconsin, California, Florida's Broward County, and other jurisdictions. Depending on the scores generated by this software, the judge can decide upon whether to detain the defendant prior to trial and/or when sentencing. It has been observed that the Defendants who are classified medium or high risk (scores of 5–10), are more likely to be held in prison while awaiting trial than those classified as low risk (scores of 1–4). You can read about other analyses that have been done about the COMPAS dataset from https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm

At a high-level, the goal of this assignment is to build our own predictor of recidivism risk from a smaller subset of COMPAS dataset features, evaluate various fairness metrics on our predictor, and assess the trade-offs between said metrics. Download the starter-code and data by cloning the repository https://github.com/stanfordaiethics/hw1_public. Everything you need to complete homework 1 is contained in this repository. Once finished, you may run `make_submission.sh` to generate `hw1.zip` which should be submitted on Gradescope. More detailed instructions are on the repository's README.

Warm-ups

Problem 1: Data Pre-processing [5 points]

In this exercise, you will see that real-world data contains a fair amount of labeling biases and data corruption which must be removed during the data pre-processing phase. In the provided starter code, the COMPAS dataset is loaded for you in a variable named `raw_data` which is a `pandas.DataFrame` object. (Learn more about pandas library https://bit.ly/3wTKBbQ) Each row of `raw_data` contains various feature values for a single person. The name of the features can be found in `raw_data.keys()`. Some examples of these features include `name, age, days_b_screening_arrest`, and etc. From `raw_data` remove the rows with the following feature values:

- If the charge date, i.e `days_b_screening_arrest`, of a defendants COMPAS scored crime was not within ± 30 days from when the person was arrested, we assume we do not have the right offense. Thus only rows with $|$`days_b_screening_arrest`$| \leq 30$ should be kept.
- The rows with the recidivist flag `is_recid` $== -1$ corresponds to rows where we could not find a COMPAS case and should be discarded.
- In a similar vein, ordinary traffic offenses, i.e rows with a `c_charge_degree` value of `O`, will not result in jail time and should be removed.

- Remove rows with no proper `score_text`, i.e `N/A`

The filtered dataset should be stored in a variable named `df` and still be a `pandas.DataFrame` object. Report the number of remaining rows in your filtered dataset `df`.

Problem 2: Data Visualization [10 points]
Let us get a sense of what the COMPAS scores look like to obtain an intuition for what biases our learned predictor might also have. The COMPAS scores range from 1 to 10 and are stored in the column `decile_scores`. Visualize, using histograms, the `decile_scores` for the demographics of each class label in the protected features `race` and `sex`. Attach histograms to submission and comment on what biases are present in the COMPAS scores. Comment on the what the histograms reveal. (*Hint: you should have 8 histograms since there are six class labels for race and two class labels for sex*)

Learning a Predictor

Problem 3: Logistic Regression [15 points] Let $x = (x_1, ..., x_n) \in \mathbb{R}^n$ represent a feature vector, e.g x_1 might represent gender. A logistic regression model $f_\theta : \mathbb{R}^n \to (0, 1)$, for a binary classification problem, is a parametrized function of the form:

$$f_\theta(x) = \frac{1}{1 + e^{\theta_0 + \sum_{i=1}^{n} \theta_i x_i}}$$

where $\theta = (\theta_0, ..., \theta_n)$ are the coefficients representing the impact of each feature x_i on the outputs of the model. As the output of the logistic regression model is constrained to lie in the interval $(0, 1)$, it may be interpreted as an estimated probability for the label of x being 1. (Assuming the binary labels are either 0 or 1) We will now build our own predictor of the risk of recidivism. Our predictor will take as input the following features:

- `race`: Race of the individual
- `sex`: Biological sex of the individual
- `age_cat`: Age of the individual categorized into "Less than 25", "Between 25-45", and "Greater than 45"
- `c_charge_degree`: Degree of the crime categorized into "Minor (M)" and "Felony (F)"
- `priors_count`: Number of prior crimes which range from 0 - 38.

And predict:

- `two_year_recid`: A binary flag representing whether a new misdemeanor or felony offense has occured within two years of the COMPAS score administration date.

In summary, our predictor f_θ maps (`race`, `sex`, `age_cat`, `c_charge_degree`, `priors_count`) $\mapsto$ `two_year_recid`.

(a). [5 points] We must first process the input features from our filtered dataset `df` to be in a form that can be input to our logistic regression model. From your filtered COMPAS dataset `df`, extract the columns corresponding to the above features and convert them into one-hot vectors using `pandas.get_dummies`, with the exception of the `priors_count` feature which should use the original values as there are too many different classes. The resulting feature columns should be concatenated along the column axis and stored in a variable named `features`. The target vector that our model will be predicting has been processed for you and stored in the variable `target`. (*Hint: your features vector should have 14 columns*)

(b). [5 points] Your next task is to split `(features, target)` into a training and test set where the training set should consist of 75% of your data. Use `sklearn.model_selection.train_test_split` to perform the split. As we've fixed the random seed, you should always get the same train/test split of your data.

(c). [5 points] Fit a Logistic Regression model on the *training set*. Report the prediction accuracy on *both the training and test data* measured as the fraction of predicted labels that were correct.

(Hint: use `sklearn.linear_model.LogisticRegression` function with the arguments `solver='lbfgs'` and `max_iter=10000` for your model and the `.score()` method to measure prediction accuracy)

Evaluating Fairness Metrics

We first introduce some notation before the next sections. Once again let $x \in \mathbb{R}^n$ be an feature vector which represents the visible attributes of an individual. The features can be further partitioned into protected features x_p and and unprotected features x_u such that $x = (x_p, x_u)$. For ease of exposition, we often imagine the protected features indicate an individual's race or gender, but they might also represent other sensitive attributes. For each individual, we are tasked to predict a binary target $y \in \{0, 1\}$. In our COMPAS dataset, y corresponds to `two_year_recid`, the label of recidivism. Importantly, y is not known to the decision maker, e.g a judge, who at the time of the decision has access only to information encoded in the visible features x. Third, we define random variables X and Y that take on values $X = x$ and $Y = y$ for an individual drawn randomly from the population of interest (e.g. the population of defendants for whom pretrial decisions must be made). We use X_p and X_u to denote the projections of x onto its protected and unprotected components. Fourth, we define the true risk function $r(x) = \Pr(Y = 1|X = x)$. Finally, a decision algorithm, or a decision rule, is a function $d : \mathbb{R}^n \to \{0, 1\}$, where $d(x) = 0$ means that action a_0 is taken and so fourth.

For many risk assessment algorithms, such as COMPAS, instead of simply outputting a decision a_0 or a_1, we produce a risk score $s(x)$ that may be viewed as an approximation of the true risk $r(x)$. Note that $s(x)$ may only be loosely related to the true risk, and $s(x)$ may not even lie in the interval $[0, 1]$ (e.g., $s(x) \in \{1, 2, ..., 10\}$ may represent a risk decile). To go from risk scores to decisions, it is common to simply threshold the score, setting $d(x) = 1$ if and only if $s(x) \geq t$ for some fixed threshold $t \in \mathbb{R}$.

Problem 4: Demographic Parity [10 points] Our logistic regression model $f_\theta(x)$ may be viewed as a risk score $s(x)$, as it is indeed an estimate of the true risk $r(x)$ of recidivism. Since $f_\theta(x) \in (0, 1)$, the risk score can be thought of as an estimated probability of recidivism. Consider a decision rule $d(x) = 1$ if $f_\theta(x) \geq 0.5$, and $d(x) = 0$ otherwise, e.g if the estimated probability of recidivism is higher than 0.5, the judge detains the defendant. *Demographic parity* is achieved for a protected feature p when the proportion of positive decisions across the entire demographic is the same as the proportion of positive decisions among people with protected feature p:

$$\Pr(d(X) = 1|X_p) = \Pr(d(X) = 1).$$

First compute $\Pr(d(X) = 1)$ which is the ratio of positive decisions, i.e $d(x) = 1$, when using the above decision rule on the test set. Next compute $\Pr(d(X) = 1|X_p)$ for the protected features $p =$`African-American`, `Caucasian`, `Male`, and `Female`. This amounts to computing the ratio of positive decisions on a subset of rows of the test set where the protected feature column value is $x_p = 1$. Is demographic parity achieved? If not, what demographics have a higher proportion of positive decisions? Comment on your findings.

Problem 5: Pitfalls of Demographic Parity [5 points] What are some approaches that can be used to regularize our risk predictor f_θ such that demographic parity is achieved? What are potential pitfalls of using such approaches to correct the learned model f_θ to satisfy demographic parity?

Problem 6: Calibration Fairness [15 points] Calibration fairness is achieved for a protected attribute p when

$$\Pr(Y = 1|s(X), X_p) = \Pr(Y = 1|s(X)).$$

For our COMPAS dataset, calibration fairness means that among defendants with a given risk score, the proportion who would re-offend if released is the same across protected feature groups.

Consider the same decision rule as in Problem 4 except with a risk score $s(x) = \text{discretize}(f_\theta(x))$ where $\text{discretize}(v) : \mathbb{R} \to \{0, 1, 2, 3, 4\}$ is a dicretization function that assigns $d(v) = n$ if $0.2 \cdot n \leq v \leq 0.2 \cdot (n + 1)$. Compute empirical estimates of both $\Pr(Y = 1|s(X), X_p)$ and $\Pr(Y = 1|s(X))$ on the *test set* for the protected attributes $p =$`African-American`, `Caucasian`, `Male`, and `Female`. Is calibration fairness achieved? Comment on your findings.

Problem 7: Post-hoc Calibration Fairness [15 points] We will now use post-hoc calibration methods, namely Platt's method, to hopefully improve the calibration fairness of our predictor. Fit a calibrated predictor to the *training set* using `sklearn.calibration.CalibratedClassifierCV` with the default arguments except for `base_estimator` which should be set to the logistic regression model from before and the cross-validation split `cv` which should be set to 3. Redo the computations in Problem 6 for the same protected features. (*Hint: the sklearn documentation page for CalibratedClassifierCV can be quite helpful*)

Problem 8. Effect of Post-hoc Calibration on Demographic Parity [10 points] Redo the computations from Problem 4 with the calibrated predictor. Explain how these values changed from Problem 7.

Problem 9: Optimal Decision Threshold [15 points] Let $b_{00} \geq 0$ be the benefit of taking action a_0 when $y = 0$, and let $b_{11} \geq 0$ be the benefit of taking action a_1 when $y = 1$. Likewise denote by $c_{01} \geq 0$ the cost of taking action a_0 when $y = 1$ and denote by $c_{10} \geq 0$ the cost of taking action a_1 when $y = 0$. Suppose one follows the simple decision rule of $d(x) = 1$ if $r(x) \geq t$ and $d(x) = 0$ otherwise. Further assume that the benefits and cost values are constant for all x. What is the optimal decision threshold t which guarantees that the expected utility of this decision strategy is non-negative? (*Hint: first consider the expected utility of each action separately*)