PS 8

June 5, 2023

[1]:
```
import pandas as pd
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
np.random.seed(1)
```

0.1 1. Is COMPAS fair? (50pt)

0.1.1 1.1 Load and prepare

1.1.1. (1pt) Load the COMPAS data, and perform the basic checks.

[2]:
```
compas = pd.read_csv("compas-score-data.csv.bz2", sep="\t")
print(f"Rows, Columns: {compas.shape}")
print("\nColumn Types")
print(compas.dtypes)
print("\nNumber of NaN values in each column")
print(compas.isna().sum())
print("\nSample")
print(compas.sample(3))
```

```
Rows, Columns: (6172, 8)

Column Types
age                  int64
c_charge_degree     object
race                object
age_cat             object
sex                 object
priors_count         int64
decile_score         int64
two_year_recid       int64
dtype: object

Number of NaN values in each column
age                 0
c_charge_degree     0
```

```
race                0
age_cat             0
sex                 0
priors_count        0
decile_score        0
two_year_recid      0
dtype: int64

Sample
      age c_charge_degree              race          age_cat   sex  \
3984   54               F         Caucasian  Greater than 45  Male
3179   31               F  African-American          25 - 45  Male
5484   22               F  African-American     Less than 25  Male

      priors_count  decile_score  two_year_recid
3984             2             4               1
3179             4             8               1
5484             3             6               1
```

1.1.2. (1pt) Filter the data to keep only Caucasian and African-Americans.

[3]:
```
compas = compas[(compas.race == "African-American") | (compas.race ==␣
↪"Caucasian")]
```

1.1.3. (2pt) Create a new dummy variable based off of COMPAS risk score (decile_score), which indicates if an individual was classified as low risk (score 1-4) or high risk (score 5-10).

[4]:
```
compas["highscore"] = np.where(compas.decile_score <= 4, 0, 1)
```

1.1.4. (4pt) Now analyze the offenders across this new risk category:

(a) What is the recidivism rate (percentage of offenders who re-commit the crime) for low- risk and high-risk individuals?

[5]:
```
low_risk_rate = (compas[compas.highscore == 0].two_year_recid == 1).mean()
print(f"Low Risk Recidivism Rate: {low_risk_rate * 100}%")
high_risk_rate = (compas[compas.highscore == 1].two_year_recid == 1).mean()
print(f"High Risk Recidivism Rate: {high_risk_rate * 100}%")
```

```
Low Risk Recidivism Rate: 32.00145296040683%
High Risk Recidivism Rate: 63.445544554455445%
```

(b) What are the recidivism rates for African-Americans and Caucasians?

[6]:
```
aa_risk_rate = (compas[compas.race == "African-American"].two_year_recid == 1).
↪mean()
print(f"African-American Recidivism Rate: {aa_risk_rate * 100}%")
c_risk_rate = (compas[compas.race == "Caucasian"].two_year_recid == 1).mean()
```

```
print(f"Caucasians Risk Rate: {c_risk_rate * 100}%")
```

```
African-American Recidivism Rate: 52.31496062992126%
Caucasians Risk Rate: 39.08701854493581%
```

1.1.5. (7 pt) Create a confusion matrix (CM) comparing COMPAS predictions for recidivism (low risk/high risk you created above) and the actual two-year recidivism and interpret the results. In order to be on the same page, let's call recidivists "positives".

[7]:
```
cm = pd.crosstab(compas.two_year_recid, compas.highscore)
```

Actual/Predicted	Non Recidivists	Recidivists	Total
Non Recidivists	1872	923	2795
Recidivists	881	1602	2483
	2753	2525	5278

1.1.6. (7pt) Discuss the CM. What is accuracy? What percentage of low-risk individuals are wrongly classified as high risk? What about the way around?

Would you feel comfortable having a judge to use COMPAS to inform sentencing guidelines? What do you think, how well can judges perform the same task without COMPAS's help? Are they better or worse? At what point would the error/misclassification risk be acceptable for you? Do you think the acceptable error rate should be the same for human judges and for algorithms?

[8]:
```
TN = 1872
FP = 923
FN = 881
TP = 1602
acc = (TP + TN)/5278
print(f"Accuracy: {acc}")
print(f"FPR: {FP/2795 * 100}%")
print(f"FNR: {FN/2483 * 100}%")
```

```
Accuracy: 0.6582038651004168
FPR: 33.02325581395349%
FNR: 35.48127265404752%
```

I would not fell comfortable having a judge use compas to inform sentencing guidelines, having an accuracy of 0.6582 is low, since it means the algorithm only got 65.82% of the predictions correct. Also having 33.02% in FPR and 35.48% in FNR is both too high, since it meant that it usually misclassifies around one-thirds of low-risk people or high-risk people. I think a judge performs the same tasks better without COMPAS's help. In order to use compas, I would need the accuracy to be at least 96~98% and the FPR and FNR to be both lower than 3%.

0.1.2 1.2 Analysis by race (28pt)

1.2.1. (2pt) Compute the recidivism rate separately for high-risk and low risk African-Americans and Caucasians.

[9]:
```
low_risk_aa_rate = (compas[(compas.highscore == 0) & (compas.race ==␣
 ↪"African-American")].two_year_recid == 1).mean()
print(f"Low Risk African American Recidivism Rate: {low_risk_aa_rate * 100}%")
high_risk_aa_rate = (compas[(compas.highscore == 1) & (compas.race ==␣
 ↪"African-American")].two_year_recid == 1).mean()
print(f"High Risk African American Recidivism Rate: {high_risk_aa_rate * 100}%")
low_risk_c_rate = (compas[(compas.highscore == 0) & (compas.race ==␣
 ↪"Caucasian")].two_year_recid == 1).mean()
print(f"Low Risk Caucasian Recidivism Rate: {low_risk_c_rate * 100}%")
high_risk_c_rate = (compas[(compas.highscore == 1) & (compas.race ==␣
 ↪"Caucasian")].two_year_recid == 1).mean()
print(f"High Risk Caucasian Recidivism Rate: {high_risk_c_rate * 100}%")
```

```
Low Risk African American Recidivism Rate: 35.14115898959881%
High Risk African American Recidivism Rate: 64.95352651722253%
Low Risk Caucasian Recidivism Rate: 28.997867803837952%
High Risk Caucasian Recidivism Rate: 59.48275862068966%
```

1.2.2. (6pt) Comment the results in the previous point. How similar are the rates for the the low- risk Caucasians and low-risk African Americans? For the high-risk Caucasians and high-risk African Americans? Do you see a racial disparity here? If yes, which group is it favoring? Based on these figures, do you think COMPAS is fair? Overall the recidivism rate is higher for African Americans when compared with Caucasians. For low-risks, African American Recidivism Rate is higher by around 6.14% while for high-risks, African American Recidivism Rate is higher by 5.47%. I do see a racial disparity, Compas favors Caucasians, therefore based on these figures, COMPAS is not fair as it rates a group higher in both categorizations.

1.2.3. (6pt) Now repeat your confusion matrix calculation and analysis from 1.1.5. But this time do it separately for African-Americans and for Caucasians:

[10]:
```
aa_cm = pd.crosstab(compas[compas.race == "African-American"].two_year_recid,␣
 ↪compas[compas.race == "African-American"].highscore)
c_cm = pd.crosstab(compas[compas.race == "Caucasian"].two_year_recid,␣
 ↪compas[compas.race == "Caucasian"].highscore)
```

African American

Actual/Predicted	Non Recidivists	Recidivists	Total
Non Recidivists	873	641	1514
Recidivists	473	1188	1661
	1346	1829	3175

Caucasian			
Actual/Predicted	Non Recidivists	Recidivists	Total
Non Recidivists	999	282	1281
Recidivists	408	414	822
	1407	696	2103

(a) How accurate is the COMPAS classification for African-Americans and for Caucasians?

[11]:
```
aa_TN = 873
aa_FP = 641
aa_FN = 473
aa_TP = 1188
c_TN = 999
c_FP = 282
c_FN = 408
c_TP = 414
print(f"African American Accuracy: {(aa_TP + aa_TN)/3175}")
print(f"Caucasian Accuracy: {(c_TP + c_TN)/2103}")
```

```
African American Accuracy: 0.6491338582677165
Caucasian Accuracy: 0.6718972895863052
```

(b) What are the false positive rates (false recidivism rates) FPR?

[12]:
```
print(f"African American FPR: {aa_FP/1514 * 100}%")
print(f"Caucasian FPR: {c_FP/1281 * 100}%")
print(f"Diff: {(aa_FP/1514 * 100) - (c_FP/1281 * 100)}")
```

```
African American FPR: 42.33817701453104%
Caucasian FPR: 22.01405152224824%
Diff: 20.3241254922828
```

(c) The false negative rates (false no-recidivism rates) FNR?

[13]:
```
print(f"African American FNR: {aa_FN/1661 * 100}%")
print(f"Caucasian FNR: {c_FN/822 * 100}%")
print(f"Diff: {(aa_FN/1661 * 100) - (c_FN/822 * 100)}")
```

```
African American FNR: 28.47682119205298%
Caucasian FNR: 49.63503649635037%
Diff: -21.158215304297386
```

1.2.4. (6pt) If you have done this correctly, you will find that COMPAS's percentage of correctly categorized individuals (accuracy) is fairly similar for African-Americans and Caucasians, but that false positive rates and false negative rates are different. In your opinion, is the COMPAS algorithm "fair"? Justify your answer. No COMPAS is not fair because even though the accuracy is similar, COMPAS actually misclassified more low risk African American as High Risk and misclassifies more high risk Caucasians as low risk individuals.

1.2.5. (8pt) Does your answer in 4 align with your answer in 2? Explain! My answers align but the justification is different, after seperating the races, the analysis showed that COMPAS isn't fair because of the high disparity in FPR and FNR, such that it misclassified many low risk African Americans as high risk and high risk Caucasians as low risks.

0.2 2. Can you beat COMPAS? (50pt)

2.1. Create the model (30pt)

2.1.1. (6pt) Before we start: what do you think, what is an appropriate model performance measure here? A, P, R, F or something else, such as FPR or FNR? Maybe you want to report multiple measures? Explain! I think accuracy, FPR, FNR are the three measures I should measure, because we need to see how well the model predicts the outcome, but also measure the amount of false positives and false negatives we have in order to see if our model is particular bias towards a race or gender.

2.1.2. (6pt) you should not use variable decile score that originates from COMPAS model. Why? Decile score is COMPAS's prediction, since we aim to produce a better model than COMPAS, we shouldn't use the score the COMPAS predicted in our model

2.1.3. (8pt) Now it is time to do the modeling. Create a logistic regression model that contains all explanatory variables you have in data into the model. (Some of these you have to convert to dummies). Do not include the variables discussed above, do not include race and gender in this model to avoid explicit gender/racial bias.

Use 10-fold cross-validation (CV) to compute its relevant performance measure(s) you discussed above. Some basic code for CV is in Python Notes 13.2 Cross Validation background explanations are in the ISLR book Section 5.1 Cross-Validation.

```
[14]: y = compas.two_year_recid
      X = compas.drop(columns=["two_year_recid", "sex", "race", "decile_score",␣
       ↪"highscore"])
      # X = compas.drop(columns=["two_year_recid", "sex", "race", "decile_score",␣
       ↪"highscore", "age_cat", "age"])
      X = pd.get_dummies(X, columns=["age_cat", "c_charge_degree"])
      # X = pd.get_dummies(X, columns=["c_charge_degree"])
      m = LogisticRegression(max_iter = 1000).fit(X, y)
      def convert(num):
          l = []
          for i in num:
              l.append((i - 1) * -1)
          return l
      X_inverted = X.copy()
      X_inverted[["age_cat_25 - 45", "age_cat_Greater than 45", "age_cat_Less than␣
       ↪25", "c_charge_degree_F", "c_charge_degree_M"]] = X_inverted[["age_cat_25 -␣
       ↪45", "age_cat_Greater than 45","age_cat_Less than 25","c_charge_degree_F",␣
       ↪"c_charge_degree_M"]].apply(convert)
```

```
# X_inverted[["c_charge_degree_F", "c_charge_degree_M"]] =␣
↪X_inverted[["c_charge_degree_F", "c_charge_degree_M"]].apply(convert)
y_inverted = convert(y)
m_inverted = LogisticRegression(max_iter = 1000).fit(X_inverted, y_inverted)
acc_cv = cross_val_score(m, X, y, scoring="accuracy", cv=10).mean()
FNR_cv = np.subtract(np.ones(10), cross_val_score(m, X, y, scoring="recall",␣
↪cv=10)).mean()
FPR_cv = np.subtract(np.ones(10), cross_val_score(m_inverted, X_inverted,␣
↪y_inverted, scoring="recall", cv=10)).mean()
print(f"Accuracy: {acc_cv}")
print(f"FNR: {FNR_cv}")
print(f"FPR: {FPR_cv}")
```

```
Accuracy: 0.6703291213846242
FNR: 0.3838369607462107
FPR: 0.28157962109575013
```

4. (10pt) Experiment with different models to find the best model according to your performance indicator. Try trees and k-NN, you may also include other types of models. Include/exclude different variables. You may also do feature engineering, e.g. create a different set of age groups, include variables like age2, age3, interaction effects, etc. But do not include race and gender. Report what did you try (no need to report the full results of all of your unsuccessful attempts), and your best model's performance. Did you got better results or worse results than COMPAS?

[15]:
```
mTree = DecisionTreeClassifier().fit(X, y)
mTree_inverted = DecisionTreeClassifier().fit(X_inverted, y_inverted)
acc_cv = cross_val_score(mTree, X, y, scoring="accuracy", cv=10).mean()
FNR_cv = np.subtract(np.ones(10), cross_val_score(mTree, X, y,␣
↪scoring="recall", cv=10)).mean()
FPR_cv = np.subtract(np.ones(10), cross_val_score(mTree_inverted, X_inverted,␣
↪y_inverted, scoring="recall", cv=10)).mean()
print(f"Accuracy: {acc_cv}")
print(f"FNR: {FNR_cv}")
print(f"FPR: {FPR_cv}")
```

```
Accuracy: 0.646263872117762
FNR: 0.45994299779764225
FPR: 0.3313082437275986
```

[16]:
```
mNeigh = KNeighborsClassifier().fit(X, y)
mNeigh_inverted = KNeighborsClassifier().fit(X_inverted, y_inverted)
acc_cv = cross_val_score(mNeigh, X, y, scoring="accuracy", cv=10).mean()
FNR_cv = np.subtract(np.ones(10), cross_val_score(mNeigh, X, y,␣
↪scoring="recall", cv=10)).mean()
FPR_cv = np.subtract(np.ones(10), cross_val_score(mNeigh_inverted, X_inverted,␣
↪y_inverted, scoring="recall", cv=10)).mean()
```

```
print(f"Accuracy: {acc_cv}")
print(f"FNR: {FNR_cv}")
print(f"FPR: {FPR_cv}")
```

```
Accuracy: 0.6263713989994826
FNR: 0.44662844928099493
FPR: 0.30876600102406554
```

I tried removing age and age_cat from the model, but the results turned out worse, my best results came from the logistics model having a higher accuracy of 0.6703 and worse FNR of 0.3838, but a better FPR of 0.2815.

0.2.1 2.2 Is your model more fair? (20pt)

2.2.1. (6pt) Now use your best model to predict the two-year recidivism risk, and compute the percentage of the predicted low-risk and high-risk individuals who recidivate, by race (replicate 1.2-1). Is your model more or less fair than COMPAS?

[17]:
```
compas["predictions"] = m.predict(X)
```

[18]:
```
low_risk_aa_rate = (compas[(compas.predictions == 0) & (compas.race ==␣
↪"African-American")].two_year_recid == 1).mean()
print(f"Low Risk African American Recidivism Rate: {low_risk_aa_rate * 100}%")
high_risk_aa_rate = (compas[(compas.predictions == 1) & (compas.race ==␣
↪"African-American")].two_year_recid == 1).mean()
print(f"High Risk African American Recidivism Rate: {high_risk_aa_rate * 100}%")
low_risk_c_rate = (compas[(compas.predictions == 0) & (compas.race ==␣
↪"Caucasian")].two_year_recid == 1).mean()
print(f"Low Risk Caucasian Recidivism Rate: {low_risk_c_rate * 100}%")
high_risk_c_rate = (compas[(compas.predictions == 1) & (compas.race ==␣
↪"Caucasian")].two_year_recid == 1).mean()
print(f"High Risk Caucasian Recidivism Rate: {high_risk_c_rate * 100}%")
```

```
Low Risk African American Recidivism Rate: 33.10201249132547%
High Risk African American Recidivism Rate: 68.28143021914647%
Low Risk Caucasian Recidivism Rate: 31.183510638297875%
High Risk Caucasian Recidivism Rate: 58.931552587646074%
```

The results were the same since African Americans have higher rates in both categories. However the difference in the Low Risk category was smaller, but the difference in the High Risk category was bigger.

2. (6pt) Compute FPR and FNR by race (replicate 1.2-3 the FNR/FPR question). Is your model more or less fair than COMPAS?

[19]:
```
aa_cm = pd.crosstab(compas[compas.race == "African-American"].two_year_recid,␣
↪compas[compas.race == "African-American"].predictions)
c_cm = pd.crosstab(compas[compas.race == "Caucasian"].two_year_recid,␣
↪compas[compas.race == "Caucasian"].predictions)
```

African American			
Actual/Predicted	Non Recidivists	Recidivists	Total
Non Recidivists	964	550	1514
Recidivists	477	1184	1661
	1441	1734	3175

Caucasian			
Actual/Predicted	Non Recidivists	Recidivists	Total
Non Recidivists	1035	246	1281
Recidivists	469	353	822
	1504	599	2103

[20]:
```
aa_TN = 964
aa_FP = 550
aa_FN = 477
aa_TP = 1184
c_TN = 1035
c_FP = 246
c_FN = 469
c_TP = 353
print(f"African American FPR: {aa_FP/1514 * 100}%")
print(f"Caucasian FPR: {c_FP/1281 * 100}%")
print(f"Diff: {(aa_FP/1514 * 100) - (c_FP/1281 * 100)}")
print(f"African American FNR: {aa_FN/1661 * 100}%")
print(f"Caucasian FPR: {c_FN/822 * 100}%")
print(f"Diff: {(aa_FN/1661 * 100) - (c_FN/822 * 100)}")
```

```
African American FPR: 36.327608982826945%
Caucasian FPR: 19.20374707259953%
Diff: 17.123861910227415
African American FNR: 28.717639975918118%
Caucasian FPR: 57.05596107055961%
Diff: -28.338321094641493
```

3. (8pt) Interpret your results from 2.2.1 and 2.2.2, and explain whether your model is any better (or worse) than COMPAS in terms of fairness. My model is slightly more fair in terms of the FPR rating, but more unfair in terms of FNR rating. My model is has a smaller difference in the FPR rating of the two races by around 3% but more unfair in te FNR rating by 7%. However my model is still unable to equalize the FPR ad the FNR of the two races, which means that my model is still unfair and is still misclassifying low risk African Americans as high risk and high risk caucasians as low risk. The highlight of my model is that it managed to decrease the FPR of both African American and Caucasian, and even decreased the FPR for African Americans even more.

Around 4 hours