PS08

June 3, 2023

1 PS 08

1.1 Name: Xinyu Chang

---

[1]:
```
# import the packages
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import confusion_matrix, accuracy_score, f1_score,␣
↪recall_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
```

1.2 1 Is COMPAS fair?

1.2.1 1.1 Load and prepare

1. Load the COMPAS data, and perform the basic checks.

[2]:
```
compas = pd.read_csv("compas-score-data.csv.bz2", sep='\t')
compas
```

[2]:

	age	c_charge_degree	race	age_cat	sex	priors_count	decile_score	two_year_recid
0	69	F	Other	Greater than 45	Male	0	1	0
1	34	F	African-American	25 - 45	Male	0	3	1
2	24	F	African-American	Less than 25	Male			
3	44	M	Other	25 - 45	Male			
4	41	F	Caucasian	25 - 45	Male			
…	…	…	…	…	…			
6167	23	F	African-American	Less than 25	Male			
6168	23	F	African-American	Less than 25	Male			
6169	57	F	Other	Greater than 45	Male			
6170	33	M	African-American	25 - 45	Female			
6171	23	F	Hispanic	Less than 25	Female			

```
2                4              4                1
3                0              1                0
4               14              6                1
…              …              …              …
6167             0              7                0
6168             0              3                0
6169             0              1                0
6170             3              2                0
6171             2              4                1

[6172 rows x 8 columns]
```

[3]:
```
compas.shape
```

[3]:
```
(6172, 8)
```

[4]:
```
compas.isna().sum()
```

[4]:
```
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
```

[5]:
```
compas.dtypes
```

[5]:
```
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
```

[6]:
```
compas.describe()
```

[6]:
```
                age  priors_count  decile_score  two_year_recid
count  6172.000000   6172.000000   6172.000000     6172.000000
mean     34.534511      3.246436      4.418503        0.455120
std      11.730938      4.743770      2.839463        0.498022
min      18.000000      0.000000      1.000000        0.000000
```

```
25%       25.000000      0.000000      2.000000        0.000000
50%       31.000000      1.000000      4.000000        0.000000
75%       42.000000      4.000000      7.000000        1.000000
max       96.000000     38.000000     10.000000        1.000000
```

There are 6172 rows and 8 columns in the current dataset. There are no missing values in the current dataset. The data types are int64 for "age", "priors_count", "decile_score", and "two_year_recid" and the data types are object for "c_charge_degree", "race", "age_cat", and "sex". It seems reasonable for the mean, std, min, max for the numeric variables(age, priors_count, decile_score and two_year_recid).

2. Filter the data to keep only Caucasian and African-Americans.All the tasks below are about these two races only, there are just too few other offenders.

[7]:
```
new_compas = compas[compas['race'].isin(['Caucasian', 'African-American'])]
new_compas
```

[7]:
```
      age c_charge_degree             race       age_cat     sex  \
1      34               F  African-American       25 - 45    Male
2      24               F  African-American  Less than 25    Male
4      41               F         Caucasian       25 - 45    Male
6      39               M         Caucasian       25 - 45  Female
7      27               F         Caucasian       25 - 45    Male
…   …              …                 …             …     …
6165   30               M  African-American       25 - 45    Male
6166   20               F  African-American  Less than 25    Male
6167   23               F  African-American  Less than 25    Male
6168   23               F  African-American  Less than 25    Male
6170   33               M  African-American       25 - 45  Female

      priors_count  decile_score  two_year_recid
1                0             3               1
2                4             4               1
4               14             6               1
6                0             1               0
7                0             4               0
…              …           …             …
6165             0             2               1
6166             0             9               0
6167             0             7               0
6168             0             3               0
6170             3             2               0

[5278 rows x 8 columns]
```

[8]:
```
new_compas.race.value_counts()
```

[8]:
```
African-American    3175
Caucasian           2103
Name: race, dtype: int64
```

3. 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).Hint: you can do it in different ways but for technical reasons related the tasks below, the best way to do it is to create a variable "high score", that takes values 1 (decile score 5 and above) and 0 (decile score 1-4).

[9]:
```
new_compas = new_compas.copy()
new_compas['high_score'] = new_compas['decile_score'].apply(lambda x: 1 if x >=␣
 ↪5 else 0)
new_compas
```

[9]:

	age	c_charge_degree	race	age_cat	sex	priors_count	decile_score	two_year_recid	high_score
1	34	F	African-American	25 - 45	Male	0	3	1	0
2	24	F	African-American	Less than 25	Male	4	4	1	0
4	41	F	Caucasian	25 - 45	Male	14	6	1	1
6	39	M	Caucasian	25 - 45	Female	0	1	0	0
7	27	F	Caucasian	25 - 45	Male	0	4	0	0
…	…	…	…	…	…	…	…	…	…
6165	30	M	African-American	25 - 45	Male	0	2	1	0
6166	20	F	African-American	Less than 25	Male	0	9	0	1
6167	23	F	African-American	Less than 25	Male	0	7	0	1
6168	23	F	African-American	Less than 25	Male	0	3	0	0
6170	33	M	African-American	25 - 45	Female	3	2	0	0

[5278 rows x 9 columns]

[10]:
```
new_compas.high_score.value_counts()
```

[10]:
```
0    2753
1    2525
```

```
Name: high_score, dtype: int64
```

4. Now analyze the offenders across this new risk category:

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

[11]:
```
new_compas.groupby('high_score')['two_year_recid'].mean()
```

[11]:
```
high_score
0    0.320015
1    0.634455
Name: two_year_recid, dtype: float64
```

The recidivism rate (percentage of offenders who re-commit the crime) for lowrisk individuals is 32% and the recidivism rate (percentage of offenders who re-commit the crime) for high-risk individuals is 63.4%.

(b) What are the recidivism rates for African-Americans and Caucasians? Hint: 39% for Caucasians.

[12]:
```
new_compas.groupby('race')['two_year_recid'].mean()
```

[12]:
```
race
African-American    0.52315
Caucasian           0.39087
Name: two_year_recid, dtype: float64
```

The recidivism rates for African-Americans is 52.3% and the recidivism rates for Caucasians is 39%.

5.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". Note: you do not have to predict anything here. COMPAS has made the prediction for you, this is the high risk variable you created in 3. See the referred articles about the controversy around COMPAS methodology.

[13]:
```
cm = confusion_matrix(new_compas['two_year_recid'], new_compas['high_score'])
cm
```

[13]:
```
array([[1872,  923],
       [ 881, 1602]])
```

In this case, the classifier is the COMPAS system, and the confusion matrix is comparing the COMPAS predictions for recidivism (high risk/low risk) with the actual two-year recidivism data. We can interpret the numbers in the confusion matrix as follows:

True Negatives (TN): These are the cases where COMPAS correctly predicted that the individual would not recidivate within two years (1872 cases).Predict the non-positives and actual result is non-positives.

False Positives (FP): These are the cases where COMPAS incorrectly predicted that the individual would recidivate within two years, but they did not (923 cases).Predict the positives and actual result is non-positives.

False Negatives (FN): These are the cases where COMPAS incorrectly predicted that the individual would not recidivate within two years, but they did (881 cases).Predict the non-positives and actual result is positives.

True Positives (TP): These are the cases where COMPAS correctly predicted that the individual would recidivate within two years (1602 cases).Predict the positives and actual result is positives.

6.Discuss the CM. What is accuracy? What percentage of low-risk individuals are wrongly classified as high risk? What about the way around? We did not talk about FPR and FNR in class, but you can consult Lecture Notes, section 6.1.1 Confusion matrix and related concepts.

[14]:
```
# accuracy
accuracy = accuracy_score(new_compas['two_year_recid'],␣
↪new_compas['high_score'])
accuracy
```

[14]: 0.6582038651004168

[15]:
```
# F score
F_score = f1_score(new_compas['two_year_recid'], new_compas['high_score'])
F_score
```

[15]: 0.639776357827476

[16]:
```
# percentage of low-risk individuals are wrongly classified as high risk
FPR = 923 / (923 + 1872)
FPR
```

[16]: 0.3302325581395349

[17]:
```
# percentage of high-risk individuals are wrongly classified as low risk
FNR = 881 / (881 + 1602)
FNR
```

[17]: 0.35481272654047524

The accuracy is 65.8%. The percentage of low-risk individuals are wrongly classified as high risk is 33%(FPR). The percentage of high-risk individuals are wrongly classified as low risk is 35%(FNR).

Would you feel comfortable having a judge to use COMPAS to inform sentencing guidelines? I feel not comfortable having a judge to use COMPAS to inform sentencing guidelines. Given the data and the study by Kleinberg et al., it seems that the use of COMPAS or similar predictive tools could potentially improve outcomes in the criminal justice system. However, this does not come without concerns. In the context of recidivism prediction, a high FPR means that

a significant number of individuals who did not recidivate were predicted to do so, potentially leading to unnecessary interventions or restrictions. A high FNR means that a significant number of individuals who did recidivate were not predicted to do so, indicating missed opportunities for interventions that might have prevented recidivism. There is a higher percentage of people who are predicted not recidivate but recidivate than the percentage of people are predicted recidivate but not recidivate. Falsely predicted the person who recidivate will be more harmful than falsely predicted the person who will not recidivate. The error rates in classification, and the potential for these errors to result in unfair outcomes, is a significant issue. Additionally, there are broader concerns about transparency and potential biases in the algorithm. So, while there could be benefits, it's crucial that such tools are used with a clear understanding of their limitations and with safeguards in place to prevent unfair treatment.

What do you think, how well can judges perform the same task without COMPAS's help? Are they better or worse? I think it is hard to say how well can judges perform the same task without COMPAS's help, and are they better or worse. On one hand, the study by Kleinberg et al. suggests that machine learning tools like COMPAS might potentially outperform human judges in certain respects, for instance, in minimizing jail populations without increasing crime rates. On the other hand, the study also notes that judges might consider a broader set of variables than what the algorithm focuses on although judeges may also have some personal bias, indicating that human decision-making may have advantages in capturing complex, multifaceted considerations.

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? In my personal perspective, I think the error/misclassification should be lower than 15 percent. I think we should hold algorithms to a higher standard of accuracy because they lack the ability to consider the unique circumstances of each case in the way a human judge can. In terms of the problem whether the acceptable error rate should be the same for human judges and for algorithms, in my personal perspective, because algorithms are designed by humans and trained on human-generated data, they can reflect human biases and should be held to the same standard.

1.2.2 1.2 Analysis by race

Now we perform the fairness analysis by race. Does the model treat Caucasians and AfricanAmericans in a similar fashion?

1. Compute the recidivism rate separately for high-risk and low risk African-Americans and Caucasians. Hint: High risk AA = 65%.

[18]:
```
new_compas.groupby(['race', 'high_score'])['two_year_recid'].mean()
```

[18]:
```
race              high_score
African-American  0             0.351412
                  1             0.649535
Caucasian         0             0.289979
                  1             0.594828
Name: two_year_recid, dtype: float64
```

The recidivism rate for high-risk for African-Americans and Caucasians are 65% and 59% respectively. The recidivism rate for low-risk for African-Americans and Caucasians are 35% and 29% respectively.

2. Comment the results in the previous point. How similar are the rates for the the lowrisk 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? The recidivism rates provided indicate that for both high-risk and low-risk groups, African-Americans have a slightly higher recidivism rate than Caucasians.

High-risk group: African-Americans (65%), Caucasians (59%)
Low-risk group: African-Americans (35%), Caucasians (29%)

The difference between the recidivism rates of African-Americans and Caucasians in the high-risk group is 6 percentage points (65% - 59%). In the low-risk group, the difference is also 6 percentage points (35% - 29%). I think it is unfair for the COMPAS. From a purely numerical perspective, the rates for both high-risk and low-risk groups are not identical, but the difference in both cases is the same (6 percentage points). However, when it comes to the question of racial disparity and fairness, it's important to consider these rates in the context of the larger social, historical, and systemic factors that can influence recidivism. The COMPAS system is predicting more African-Americans to be high-risk compared to Caucasians, and these groups actually have a similar risk of recidivism, this could potentially be a sign of racial bias in the system. It implies the Caucasians is favored. is could be seen as a violation of the concept of group fairness, which requires that similar groups (in terms of the outcome of interest) are treated similarly by the system. On the other hand, the concept of individual fairness requires that similar individuals are treated similarly. If African-Americans who are similar to Caucasians in terms of their risk factors for recidivism are being more frequently or harshly labeled as high-risk by COMPAS, this could be seen as a violation of individual fairness.

3. Now repeat your confusion matrix calculation and analysis from 1.1.5. But this time do it separately for African-Americans and for Caucasians:

[19]:
```
aa = new_compas[new_compas['race'] == 'African-American']
aa
```

[19]:

	age	c_charge_degree	race	age_cat	sex
1	34	F	African-American	25 - 45	Male
2	24	F	African-American	Less than 25	Male
8	23	M	African-American	Less than 25	Male
10	41	F	African-American	25 - 45	Male
12	31	F	African-American	25 - 45	Male
…	…	…	…	…	…
6165	30	M	African-American	25 - 45	Male
6166	20	F	African-American	Less than 25	Male
6167	23	F	African-American	Less than 25	Male
6168	23	F	African-American	Less than 25	Male
6170	33	M	African-American	25 - 45	Female

```
      priors_count  decile_score  two_year_recid  high_score
1                0             3               1           0
2                4             4               1           0
8                3             6               1           1
10               0             4               0           0
12               7             3               1           0
…              …             …             …           …
6165             0             2               1           0
6166             0             9               0           1
6167             0             7               0           1
6168             0             3               0           0
6170             3             2               0           0

[3175 rows x 9 columns]
```

[20]:
```
c = new_compas[new_compas['race'] == 'Caucasian']
c
```

[20]:
```
      age c_charge_degree      race          age_cat     sex  priors_count  \
4      41               F  Caucasian          25 - 45    Male            14
6      39               M  Caucasian          25 - 45  Female             0
7      27               F  Caucasian          25 - 45    Male             0
9      37               M  Caucasian          25 - 45  Female             0
11     47               F  Caucasian  Greater than 45  Female             1
…   …              …        …               …     …           …
6148   36               M  Caucasian          25 - 45    Male             0
6151   32               F  Caucasian          25 - 45    Male             0
6153   30               M  Caucasian          25 - 45  Female             2
6158   23               F  Caucasian     Less than 25    Male             0
6164   21               M  Caucasian     Less than 25    Male             0

      decile_score  two_year_recid  high_score
4                6               1           1
6                1               0           0
7                4               0           0
9                1               0           0
11               1               1           0
…              …             …         …
6148             1               0           0
6151             2               0           0
6153             1               1           0
6158             8               0           1
6164             6               1           1

[2103 rows x 9 columns]
```

[21]:
```
cm_aa = confusion_matrix(aa['two_year_recid'], aa['high_score'])
cm_aa
```

[21]:
```
array([[ 873,  641],
       [ 473, 1188]])
```

[22]:
```
cm_c = confusion_matrix(c['two_year_recid'], c['high_score'])
cm_c
```

[22]:
```
array([[999, 282],
       [408, 414]])
```

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

[23]:
```
accuracy_aa = accuracy_score(aa['two_year_recid'], aa['high_score'])
accuracy_aa
```

[23]:
```
0.6491338582677165
```

[24]:
```
accuracy_c = accuracy_score(c['two_year_recid'], c['high_score'])
accuracy_c
```

[24]:
```
0.6718972895863052
```

The accuracy of the COMPAS classification for African-Americans and for Caucasians are 64.9% and 67.2% respectively.

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

[25]:
```
FPR_aa = 641 / (641 + 873)
FPR_c = 282 / (282 + 999)
FPR_aa, FPR_c
```

[25]:
```
(0.4233817701453104, 0.22014051522248243)
```

The false positive rates (false recidivism rates) FPR for African-Americans and for Caucasians are 42.3% and 22% respectively.

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

Hint: FPR for Caucasians is 0.22, FNR for African-Americans is 0.28

[26]:
```
FNR_aa = 473 / (473 + 1188)
FNR_c = 408 / (408 + 414)
FNR_aa, FNR_c
```

[26]:
```
(0.2847682119205298, 0.49635036496350365)
```

The false negative rates (false no-recidivism rates) FNR for African-Americans and for Caucasians are 28.5% and 49.6% respectively.

4. 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. Accuracy refers to the proportion of true results (both true positives and true negatives) in the total number of cases examined. In this case, COMPAS has a slightly higher accuracy for Caucasians (67.2%) than for African-Americans (64.9%). This difference, while not large, suggests that COMPAS's predictions are slightly more likely to be correct for Caucasians. The false positive rate (FPR) refers to the proportion of negative events that are incorrectly classified as positive. Here, the false positive rate is much higher for African-Americans (42.3%) than for Caucasians (22%). This means that African-Americans are much more likely to be incorrectly labeled as high risk (recidivist) by COMPAS compared to Caucasians. The false negative rate (FNR) refers to the proportion of positive events that are incorrectly classified as negative. In this case, the false negative rate is higher for Caucasians (49.6%) than for African-Americans (28.5%). This means that Caucasians are more likely to be incorrectly labeled as low risk (non-recidivist) by COMPAS.

The different rates of false positives and false negatives can have significant real-world impacts. A high false positive rate for African-Americans could lead to unjust outcomes, such as unnecessary incarceration or harsher sentences. Conversely, a high false negative rate for Caucasians could mean that individuals who pose a higher risk of reoffending are not being appropriately identified and managed.

Given these disparities in false positive and false negative rates, we can argue that the COMPAS algorithm is not fair, even though its overall accuracy is similar for both groups. This is because the algorithm's errors disproportionately impact different racial groups in different ways, which can contribute to systemic biases and inequalities. However, it's also worth noting that the COMPAS algorithm is not solely responsible for these disparities. The algorithm is trained on historical data, which can reflect and perpetuate existing biases in the criminal justice system. Therefore, addressing these issues may require broader systemic changes, in addition to improving the fairness of predictive algorithms like COMPAS.

5. Does your answer in 4 align with your answer in 2? Explain! Hint: This is not a trick question. If you read the first two recommended readings, you will find that people disagree how you define fairness. Your answer will not be graded on which side you take, but on your justification. Yes, my answer to this question aligns with the answer provided in question 2. The key point in both answers is that there are observed disparities in the COMPAS algorithm's predictions across different racial groups, but these disparities alone do not necessarily mean the algorithm is unfair. In question 2, I mentioned that the COMPAS system shows higher recidivism rates for African-Americans compared to Caucasians in both low-risk and high-risk groups. However, I also pointed out that these differences could be due to a wide range of factors, including socioeconomic factors, systemic discrimination, or differences in policing practices, among others. Similarly, in this question, I noted that the COMPAS algorithm shows different false positive and false negative rates for African-Americans and Caucasians, but this does not automatically mean the algorithm is unfair. The algorithm's errors disproportionately impact different racial groups, but these errors may be reflecting existing biases in the criminal

justice system rather than biases inherent in the algorithm itself. In both cases, the key issue is not just the presence of disparities, but the reasons behind these disparities and their real-world impacts. As I mentioned in both answers, determining whether the COMPAS algorithm is "fair" requires a more nuanced understanding of these issues. The recommended readings by Kleinberg et al. discuss the complexities of defining fairness in algorithms. They highlight that there can be inherent trade-offs in trying to make an algorithm fair according to different definitions of fairness. For example, an algorithm might be fair in terms of its overall accuracy across different groups, but unfair in terms of its false positive or false negative rates. This is a reflection of the fact that fairness is a multifaceted concept, and achieving fairness according to one definition can sometimes lead to unfairness according to another definition. In conclusion, the question of whether the COMPAS algorithm is fair cannot be definitively answered based solely on the observed disparities. It requires a deeper understanding of the causes and impacts of these disparities, as well as a thoughtful consideration of what we mean by "fairness" in this context.

We also need to consider the concept of fairness in algorithmic decision-making often revolves around two main paradigms: individual fairness and group fairness. Individual fairness requires that similar individuals are treated similarly. In the context of the COMPAS algorithm, this would mean that two individuals with similar criminal histories and other relevant characteristics should have similar recidivism risk scores, regardless of their race. Group fairness, on the other hand, requires that certain outcomes are balanced across different demographic groups. In this context, it could mean that the rates of false positives and false negatives in recidivism predictions should be approximately equal for African-Americans and Caucasians. The contradiction between individual and group fairness can be seen in the results I presented above. The COMPAS algorithm may be relatively fair from an individual fairness perspective, as it has similar accuracy rates for African-Americans and Caucasians. This suggests that it treats similar individuals similarly, at least in terms of its overall predictive accuracy. However, the algorithm appears to be less fair from a group fairness perspective, as it has significantly different false positive and false negative rates for African-Americans and Caucasians. This means that the errors it makes are not equally distributed across these two groups, which can lead to systemic biases and unfair outcomes.This contradiction underscores the inherent complexities and trade-offs in achieving algorithmic fairness. Depending on how we define fairness, the same algorithm can be seen as both fair and unfair. It also highlights the need for a careful and nuanced approach to developing and evaluating predictive algorithms like COMPAS, with a focus on both individual and group fairness.

1.3 2 Can you beat COMPAS?

1.3.1 2.1 Create the model

Create such a model. We want to avoid explicit race and gender bias, hence you do not want to include gender and race in order to avoid it. Finally, let's analyze the performance of the model by cross-validation.More detailed tasks are here:

1.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! The F-score is a performance metric that considers both precision (P) and recall (R), which makes it a balanced measure when I care equally about both of these aspects. In the context of predicting recidivism, precision would be the proportion of individuals who actually reoffended among all those predicted to reoffend, while recall would be the proportion of individuals who reoffended and were correctly predicted to do so out of all

individuals who actually reoffended. The F1-score is the harmonic mean of precision and recall, and it tends towards the smaller value of the two. This means that a good F1-score requires both good precision and good recall. On the other hand, the Accuracy (A) might not be a good measure in this case if the classes are imbalanced (i.e., if there are many more non-recidivists than recidivists). Accuracy could be high just by predicting the majority class. The False Positive Rate (FPR) and False Negative Rate (FNR) are useful for examining specific types of errors. For instance, if it's much worse to incorrectly predict that someone will reoffend when they won't (false positive), I would want a low FPR. If it's much worse to incorrectly predict that someone won't reoffend when they will (false negative), I would want a low FNR. By focusing on the F1-score, I am implicitly assuming that these two types of errors are equally bad, which might not be the case.

2.you should not use variable decile score that originates from COMPAS model. Why? The decile score is a direct output of the COMPAS model, which is proprietary and has been criticized for its lack of transparency, potential bias. The exact formula used to calculate the decile score is proprietary information owned by Northpointe, the company that created COMPAS. This means it's not possible to fully understand how the score is calculated, which factors are considered, and how much weight is given to each factor. This lack of transparency can make it difficult to validate the accuracy, fairness, or bias of the score. Including it as a feature in my model would introduce that same lack of transparency and potential bias. I want my model to be independent and to rely on its own features. Relying heavily on the decile score might result in overlooking other important factors in a person's situation or background that aren't captured by the COMPAS algorithm.

3.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.

[27]:
```
new_df = new_compas.drop(columns=['race', 'sex', 'decile_score'])
new_df = pd.get_dummies(new_df)
new_df
```

[27]:

	age	priors_count	two_year_recid	high_score	c_charge_degree_F	\
1	34	0	1	0	1	
2	24	4	1	0	1	
4	41	14	1	1	1	
6	39	0	0	0	0	
7	27	0	0	0	1	
…	…	…	…	…	…	
6165	30	0	1	0	0	
6166	20	0	0	1	1	
6167	23	0	0	1	1	
6168	23	0	0	0	1	
6170	33	3	0	0	0	

```
      c_charge_degree_M  age_cat_25 - 45  age_cat_Greater than 45  \
1                     0                1                        0
2                     0                0                        0
4                     0                1                        0
6                     1                1                        0
7                     0                1                        0
…                   …                …                        …
6165                  1                1                        0
6166                  0                0                        0
6167                  0                0                        0
6168                  0                0                        0
6170                  1                1                        0

      age_cat_Less than 25
1                        0
2                        1
4                        0
6                        0
7                        0
…                      …
6165                     0
6166                     1
6167                     1
6168                     1
6170                     0

[5278 rows x 9 columns]
```

[28]:
```
X = new_df.drop(columns='two_year_recid')
# Best model to save it
Xbest = X
y = new_df['two_year_recid']
```

[29]:
```
m_best = LogisticRegression(max_iter=5000)
_ = m_best.fit(X,y)
```

[30]:
```
cv = cross_val_score(m_best, X, y, cv=10, scoring="f1").mean()
cv
```

[30]:
```
0.6414062288345379
```

4. 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?

[31]:
```
ks = range(1, 100, 4)
for k in ks:
    m = KNeighborsClassifier(k)
    cv = cross_val_score(m, X, y, cv=10, scoring="f1").mean()
    print("K=", k, "and F score(10 fold cv)", cv)
```

```
K= 1 and F score(10 fold cv) 0.5662547032588329
K= 5 and F score(10 fold cv) 0.595021716471606
K= 9 and F score(10 fold cv) 0.6126383225082133
K= 13 and F score(10 fold cv) 0.6255880944898505
K= 17 and F score(10 fold cv) 0.6249634502676934
K= 21 and F score(10 fold cv) 0.6261101316216522
K= 25 and F score(10 fold cv) 0.6347230588177404
K= 29 and F score(10 fold cv) 0.6376736377345582
K= 33 and F score(10 fold cv) 0.6328562599223322
K= 37 and F score(10 fold cv) 0.6329048965149129
K= 41 and F score(10 fold cv) 0.6363734081136888
K= 45 and F score(10 fold cv) 0.6366159119253659
K= 49 and F score(10 fold cv) 0.6370140303640658
K= 53 and F score(10 fold cv) 0.6341151428384012
K= 57 and F score(10 fold cv) 0.6345006442054597
K= 61 and F score(10 fold cv) 0.6332443767480986
K= 65 and F score(10 fold cv) 0.6331095883942304
K= 69 and F score(10 fold cv) 0.6369255483620962
K= 73 and F score(10 fold cv) 0.6306892166460489
K= 77 and F score(10 fold cv) 0.6323030062134353
K= 81 and F score(10 fold cv) 0.6296598857831027
K= 85 and F score(10 fold cv) 0.6311187872564172
K= 89 and F score(10 fold cv) 0.6300770814664682
K= 93 and F score(10 fold cv) 0.6328645554052287
K= 97 and F score(10 fold cv) 0.6317525460687279
```

[32]:
```
layers = range(1, 100, 4)
for layer in layers:
    m = DecisionTreeClassifier(max_depth=layer)
    cv = cross_val_score(m, X, y, cv=10, scoring="f1").mean()
    print("Layer =", layer, "and F score(10 fold cv)", cv)
```

```
Layer = 1 and F score(10 fold cv) 0.6398977065560275
Layer = 5 and F score(10 fold cv) 0.63138920554349
Layer = 9 and F score(10 fold cv) 0.614816262385073
Layer = 13 and F score(10 fold cv) 0.593428740856074
Layer = 17 and F score(10 fold cv) 0.5807679194167163
Layer = 21 and F score(10 fold cv) 0.5794976063642338
Layer = 25 and F score(10 fold cv) 0.5797625301931094
Layer = 29 and F score(10 fold cv) 0.5809157523344426
Layer = 33 and F score(10 fold cv) 0.580129653186626
```

```
Layer = 37 and F score(10 fold cv) 0.5811809996026168
Layer = 41 and F score(10 fold cv) 0.5810310154660768
Layer = 45 and F score(10 fold cv) 0.581368633920249
Layer = 49 and F score(10 fold cv) 0.5802476159062684
Layer = 53 and F score(10 fold cv) 0.5812519510568308
Layer = 57 and F score(10 fold cv) 0.5810071099955472
Layer = 61 and F score(10 fold cv) 0.5819611388135171
Layer = 65 and F score(10 fold cv) 0.5807050748592404
Layer = 69 and F score(10 fold cv) 0.5806476857603524
Layer = 73 and F score(10 fold cv) 0.5801750003733932
Layer = 77 and F score(10 fold cv) 0.5808397111869149
Layer = 81 and F score(10 fold cv) 0.5818768575311288
Layer = 85 and F score(10 fold cv) 0.5809122757209054
Layer = 89 and F score(10 fold cv) 0.5799537238240808
Layer = 93 and F score(10 fold cv) 0.5798304797174996
Layer = 97 and F score(10 fold cv) 0.5809201998608903
```

[33]:
```
new_df['age2'] = pd.cut(new_df['age'],bins=[0,20,40,60,80],
                                labels=["0-20","20-40","40-60","60-80"])
```

[34]:
```
new_df = pd.get_dummies(new_df)
new_df
```

[34]:

	age	priors_count	two_year_recid	high_score	c_charge_degree_F	\
1	34	0	1	0	1	
2	24	4	1	0	1	
4	41	14	1	1	1	
6	39	0	0	0	0	
7	27	0	0	0	1	
…	…	…	…	…	…	
6165	30	0	1	0	0	
6166	20	0	0	1	1	
6167	23	0	0	1	1	
6168	23	0	0	0	1	
6170	33	3	0	0	0	

	c_charge_degree_M	age_cat_25 - 45	age_cat_Greater than 45	\
1	0	1	0	
2	0	0	0	
4	0	1	0	
6	1	1	0	
7	0	1	0	
…	…	…	…	
6165	1	1	0	
6166	0	0	0	
6167	0	0	0	
6168	0	0	0	

```
6170                  1                 1                          0

      age_cat_Less than 25  age2_0-20  age2_20-40  age2_40-60  age2_60-80
1                        0          0           1           0           0
2                        1          0           1           0           0
4                        0          0           0           1           0
6                        0          0           1           0           0
7                        0          0           1           0           0
…                      …          …           …           …           …
6165                     0          0           1           0           0
6166                     1          1           0           0           0
6167                     1          0           1           0           0
6168                     1          0           1           0           0
6170                     0          0           1           0           0

[5278 rows x 13 columns]
```

[35]:
```
X = new_df.drop(columns='two_year_recid')
y = new_df['two_year_recid']
```

[36]:
```
ks = range(1, 100, 4)
for k in ks:
    m = KNeighborsClassifier(k)
    cv = cross_val_score(m, X, y, cv=10, scoring="f1").mean()
    print("K=", k, "and F score(10 fold cv)", cv)
```

```
K= 1 and F score(10 fold cv) 0.566700448868809
K= 5 and F score(10 fold cv) 0.5937619745900877
K= 9 and F score(10 fold cv) 0.6128427190081109
K= 13 and F score(10 fold cv) 0.6261518583489329
K= 17 and F score(10 fold cv) 0.6239904771457854
K= 21 and F score(10 fold cv) 0.6264829273593937
K= 25 and F score(10 fold cv) 0.6335216049541755
K= 29 and F score(10 fold cv) 0.6365741032402531
K= 33 and F score(10 fold cv) 0.632765256222257
K= 37 and F score(10 fold cv) 0.6317203147744092
K= 41 and F score(10 fold cv) 0.6349913472653548
K= 45 and F score(10 fold cv) 0.636740235947862
K= 49 and F score(10 fold cv) 0.636260436938554
K= 53 and F score(10 fold cv) 0.6330374402303988
K= 57 and F score(10 fold cv) 0.6344815077723621
K= 61 and F score(10 fold cv) 0.6317320712478389
K= 65 and F score(10 fold cv) 0.6316803512587806
K= 69 and F score(10 fold cv) 0.6342862104187091
K= 73 and F score(10 fold cv) 0.6314376116008296
K= 77 and F score(10 fold cv) 0.6304667714583602
K= 81 and F score(10 fold cv) 0.628048780686284
```

```
K= 85 and F score(10 fold cv) 0.6312857093479068
K= 89 and F score(10 fold cv) 0.6293872141457968
K= 93 and F score(10 fold cv) 0.6305747858507671
K= 97 and F score(10 fold cv) 0.6308059958739577
```

[37]:
```
layers = range(1, 100, 4)
for layer in layers:
    m = DecisionTreeClassifier(max_depth=layer)
    cv = cross_val_score(m, X, y, cv=10, scoring="f1").mean()
    print("Layer =", layer, "and F score(10 fold cv)", cv)
```

```
Layer = 1 and F score(10 fold cv) 0.6398977065560275
Layer = 5 and F score(10 fold cv) 0.63138920554349
Layer = 9 and F score(10 fold cv) 0.614217808635628
Layer = 13 and F score(10 fold cv) 0.5923697599017614
Layer = 17 and F score(10 fold cv) 0.5795221889864605
Layer = 21 and F score(10 fold cv) 0.5805778193337429
Layer = 25 and F score(10 fold cv) 0.5810845232183992
Layer = 29 and F score(10 fold cv) 0.5829374849259948
Layer = 33 and F score(10 fold cv) 0.5814870251450841
Layer = 37 and F score(10 fold cv) 0.5804138870164666
Layer = 41 and F score(10 fold cv) 0.5808736880617889
Layer = 45 and F score(10 fold cv) 0.5833761318706888
Layer = 49 and F score(10 fold cv) 0.5792397006433938
Layer = 53 and F score(10 fold cv) 0.581512064470935
Layer = 57 and F score(10 fold cv) 0.5807945151201441
Layer = 61 and F score(10 fold cv) 0.5809470292967145
Layer = 65 and F score(10 fold cv) 0.5817312212266443
Layer = 69 and F score(10 fold cv) 0.5818874982503021
Layer = 73 and F score(10 fold cv) 0.5818597926652844
Layer = 77 and F score(10 fold cv) 0.5828329079848473
Layer = 81 and F score(10 fold cv) 0.5810440273149599
Layer = 85 and F score(10 fold cv) 0.5810719561471815
Layer = 89 and F score(10 fold cv) 0.57905783461521
Layer = 93 and F score(10 fold cv) 0.5816986438824693
Layer = 97 and F score(10 fold cv) 0.5813298621366162
```

I tried the decision tree model and KNN model for this question. I also do feature engineering, e.g. create a different set of age groups, include variables like age^2.

When I didn't include the "age^2" in the model, the best model performance I got is:
KNN model: K= 29 and F score(10 fold cv) is 0.6376736377345582
Decision tree model: Layer = 1 and F score(10 fold cv) is 0.6398977065560275

When I include the "age^2" in the model, the best model performance I got is:
KNN model: K= 45 and F score(10 fold cv) is 0.636740235947862
Decision tree model: Layer = 1 and F score(10 fold cv) is 0.6398977065560275

So the best F score(10 fold cv) I got is 0.6414062288345379 of the logistic regression model, and the F score of the COMPAS model is 0.6398(see above 1.1.6).

Compared with the F sore I got in the COMPAS model, my model is 0.0016 higher than that of in the COMPAS model, so my model is better than the COMPAS model.

1.3.2 2.2 Is your model more fair?

Finally, is your best model any better than COMPAS in terms of fairness? Let's use your model to predict recidivism for everyone and see if your FPR and FNR for African-Americans and Caucasians are now similar. Let's ignore the testing-training split below and just do all predictions and training on all data.

1.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?

[38]:
```
m = LogisticRegression(max_iter=5000)
_ = m.fit(Xbest,y)
predicted_recid = m.predict(Xbest)
new_compas["predicted_recid"]= predicted_recid
```

[39]:
```
recidivism_rates = new_compas.groupby(['race',␣
↪'predicted_recid'])['two_year_recid'].mean()
print(recidivism_rates)
```

```
race              predicted_recid
African-American  0                 0.338677
                  1                 0.687723
Caucasian         0                 0.299016
                  1                 0.633218
Name: two_year_recid, dtype: float64
```

From the COMPAS model:

For African-Americans with a high_score of 0 (low risk), the recidivism rate is 35.14%.
For African-Americans with a high_score of 1 (high risk), the recidivism rate is 64.95%.
For Caucasians with a high_score of 0 (low risk), the recidivism rate is 28.99%.
For Caucasians with a high_score of 1 (high risk), the recidivism rate is 59.48%.

From my model:

For African-Americans with a high_score of 0 (low risk), the recidivism rate is 33.87%.
For African-Americans with a high_score of 1 (high risk), the recidivism rate is 68.77%.
For Caucasians with a high_score of 0 (low risk), the recidivism rate is 29.9%.
For Caucasians with a high_score of 1 (high risk), the recidivism rate is 63.32%.

Group Fairness: Group fairness looks at the equality of outcomes across different demographic groups. In this case, group fairness would mean that the recidivism rates for low-risk and high-risk scores should be similar for both African-Americans and Caucasians in each model. In the COMPAS model, the recidivism rate for low-risk African-Americans is 35.14%, and for low-risk Caucasians, it's 28.99%. The difference is 6.15 percentage points. For high-risk scores, the recidivism rate is 64.95% for African-Americans and 59.48% for Caucasians, a difference of 5.47 percentage points. In my model, the recidivism rate for low-risk African-Americans is 33.87%, and for low-risk Caucasians,

it's 29.9%, a difference of 3.97 percentage points. For high-risk scores, the recidivism rate is 68.77% for African-Americans and 63.32% for Caucasians, a difference of 5.45 percentage points. From the perspective of group fairness, my model shows a smaller disparity in recidivism rates for low-risk scores between the two races compared to the COMPAS model, which could indicate that my model is more fair in this aspect. The disparity for high-risk scores is nearly the same in both models.

Individual Fairness: Individual fairness suggests that similar individuals should receive similar outcomes. This would mean that two individuals with similar criminal histories and other relevant characteristics should receive similar risk scores, regardless of their race. Since individual fairness focuses on the treatment of similar individuals, it's harder to evaluate from the provided aggregate data. However, based on the data we do have, both models are consistent in that individuals labeled as high-risk have a higher rate of recidivism than those labeled as low-risk. The larger gap in recidivism rates between low-risk and high-risk individuals in my model could suggest that it is more effective at identifying individuals who are truly at a higher risk of recidivism. However, this would need to be confirmed with more detailed, individual-level data.

In summary, my model appears to show slightly better group fairness for low-risk scores, while both models are similar in terms of high-risk scores. Individual fairness is harder to evaluate, but both models show consistent patterns.

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

[40]:
```
aa = new_compas[new_compas['race'] == 'African-American']
aa
```

[40]:

	age	c_charge_degree	race	age_cat	sex	\
1	34	F	African-American	25 - 45	Male	
2	24	F	African-American	Less than 25	Male	
8	23	M	African-American	Less than 25	Male	
10	41	F	African-American	25 - 45	Male	
12	31	F	African-American	25 - 45	Male	
…	…	…	…	…	…	
6165	30	M	African-American	25 - 45	Male	
6166	20	F	African-American	Less than 25	Male	
6167	23	F	African-American	Less than 25	Male	
6168	23	F	African-American	Less than 25	Male	
6170	33	M	African-American	25 - 45	Female	

	priors_count	decile_score	two_year_recid	high_score	predicted_recid
1	0	3	1	0	0
2	4	4	1	0	1
8	3	6	1	1	1
10	0	4	0	0	0
12	7	3	1	0	1
…	…	…	…	…	…
6165	0	2	1	0	0
6166	0	9	0	1	1
6167	0	7	0	1	1

```
6168              0              3               0           0                0
6170              3              2               0           0                0

[3175 rows x 10 columns]
```

[41]:
```
c = new_compas[new_compas['race'] == 'Caucasian']
c
```

[41]:
```
      age c_charge_degree       race          age_cat     sex  priors_count  \
4      41               F  Caucasian          25 - 45    Male            14
6      39               M  Caucasian          25 - 45  Female             0
7      27               F  Caucasian          25 - 45    Male             0
9      37               M  Caucasian          25 - 45  Female             0
11     47               F  Caucasian  Greater than 45  Female             1
…   …              …         …                …     …           …
6148   36               M  Caucasian          25 - 45    Male             0
6151   32               F  Caucasian          25 - 45    Male             0
6153   30               M  Caucasian          25 - 45  Female             2
6158   23               F  Caucasian     Less than 25    Male             0
6164   21               M  Caucasian     Less than 25    Male             0

      decile_score  two_year_recid  high_score  predicted_recid
4                6               1           1                1
6                1               0           0                0
7                4               0           0                0
9                1               0           0                0
11               1               1           0                0
…            …              …         …              …
6148             1               0           0                0
6151             2               0           0                0
6153             1               1           0                0
6158             8               0           1                1
6164             6               1           1                1

[2103 rows x 10 columns]
```

[42]:
```
cm_aa = confusion_matrix(aa['two_year_recid'], aa['predicted_recid'])
cm_aa
```

[42]:
```
array([[ 990,  524],
       [ 507, 1154]])
```

[43]:
```
cm_c = confusion_matrix(c['two_year_recid'], c['predicted_recid'])
cm_c
```

[43]:
```
array([[1069,  212],
       [ 456,  366]])
```

[44]:
```
accuracy_aa = accuracy_score(aa['two_year_recid'], aa['predicted_recid'])
accuracy_aa
```

[44]:
```
0.6752755905511811
```

[45]:
```
accuracy_c = accuracy_score(c['two_year_recid'], c['predicted_recid'])
accuracy_c
```

[45]:
```
0.6823585354255824
```

[46]:
```
tn_aa, fp_aa, fn_aa, tp_aa = cm_aa.ravel()
tn_c, fp_c, fn_c, tp_c = cm_c.ravel()
```

[47]:
```
FPR_aa = fp_aa / (fp_aa + tn_aa)
FPR_c = fp_c / (fp_c + tn_c)
FPR_aa, FPR_c
```

[47]:
```
(0.34610303830911493, 0.16549570647931303)
```

[48]:
```
FNR_aa = fn_aa / (fn_aa + tp_aa)
FNR_c = fn_c / (fn_c + tp_c)
FNR_aa, FNR_c
```

[48]:
```
(0.30523780854906685, 0.5547445255474452)
```

For the COMPAS model:
The FPR for African-Americans is 42.3% and for Caucasians is 22%.
The FNR for African-Americans is 28.5% and for Caucasians is 49.6%.

For my model:
The FPR for African-Americans is 34.61% and for Caucasians is 16.55%.
The FNR for African-Americans is 30.52% and for Caucasians is 55.47%.

Group Fairness: From a group fairness perspective, we would want the FPR and FNR to be similar for both races in each model.

In the COMPAS model, the difference in FPR between African-Americans and Caucasians is 20.3 percentage points (42.3% - 22%), and the difference in FNR is 21.1 percentage points (49.6% - 28.5%).

In my model, the difference in FPR between African-Americans and Caucasians is 18.06 percentage points (34.61% - 16.55%), and the difference in FNR is 24.95 percentage points (55.47% - 30.52%).

From a group fairness perspective, my model appears to be more fair in terms of FPR, as the disparity between the races is smaller than in the COMPAS model. However, the disparity in FNR is slightly larger in my model compared to the COMPAS model.

Individual Fairness: From the perspective of individual fairness, which aims to ensure similar individuals are treated similarly, the evaluation is more nuanced:

While my model has lower FPRs, suggesting it is better at avoiding wrongful high risk predictions for individuals, it has a higher FNR for African-Americans and Caucasians. This means my model may be more likely to underestimate the risk of recidivism for African-Americans and Caucasians, potentially overlooking individuals who may benefit from interventions. We do not have treatment information here, but we can see that given the score AA-s have higher re-offence probabilities. So There are good arguments for not treating similar individuals in a similar manner. From a individual fairness perspective, compas model appears to be more fair.

3.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. Group Fairness: My model seems to perform better than COMPAS in terms of group fairness. This is indicated by the smaller disparity in recidivism rates between different racial groups (African-Americans and Caucasians) in my model compared to COMPAS. In particular, my model has a lower False Positive Rate (FPR) for both African-Americans and Caucasians and a smaller disparity in FPR between these groups compared to COMPAS. This is a positive aspect of group fairness as it suggests that my model is less likely to unfairly label individuals as high risk.

Individual Fairness: The results are more nuanced from the perspective of individual fairness. While my model generally maintains a consistent trend in that individuals labeled as high risk have a higher rate of recidivism than those labeled as low risk, there's a potential concern that my model might be underestimating the risk of recidivism for African-Americans and Caucasians (as indicated by a higher False Negative Rate, FNR, for African-Americans and Caucasians in my model compared to COMPAS). This could mean that individuals who could benefit from interventions are being overlooked, which could be a challenge in terms of individual fairness.

In conclusion, my model seems to be more fair than COMPAS in terms of group fairness, but faces some challenges in terms of individual fairness.

1.4 Finally

1.4.1 I spent 14 hours in this problem set. Thanks for your help this quarter.