Name: Adel Hamad Alkhamisy

Miner2 username: AA
Rank on miner=19
Accuracy=0.86
F1_score=0.70

In this project, I implemented logistic regression, decision tree, Adaboost, random forest, SVM and KNN classifiers on Adult dataset which consists of 13 features and one outcome to examine the classifiers' fairness properties and experiment with a few different mitigation options.

The strategy that I used is as follow:

Preproccess: I developed a function that preprocess the dataset. First, I split the numerical and categorical features and then replaced '?' sign and null value with the most frequent item in that column. After that I used countplot to show count of observations in each category to get clear picture about the dataset. For example, in figure 1 and 2, we can see that white male make more money that other groups. Therefore, females and other races represents are minority, so we have to implement fairness metrics. I used StandardScaler to scale the numerical features to

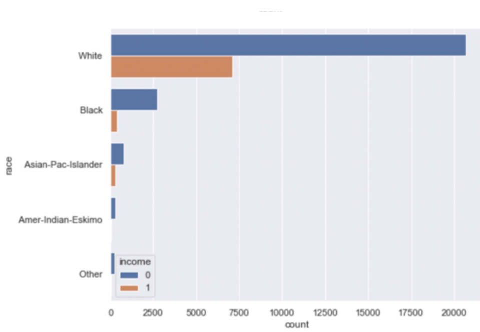

Figure 1

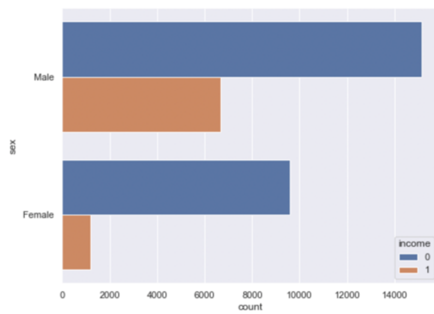

Figure 1

increase the accuracy of classifiers. Then, I encoded male and female features with 0 and 1 respectively. After that I encoded the rest of categorical features with get_dummies() to transfer them to dummies variables. I dropped the education and country features because there are not necessary.

Part 1 Classifiers:
My approach in choosing the hyper-parameter in all the following classifiers is to try and validate and record the performance metrics such as accuracy and f1_score. **First**, I check sklearn documentations to see the default values of parameters. **Second**, I start with the default values of the classifier and check the performance of my classifier. **Third**, I increase the values of parameters slightly and check the performance of the classifier again. Then, I repeat **Second and Third** until I find the **peak** performance of the classifier.

1. **Logistic regression:**
   I tried many different values in logistic regression classifier as shown in Table 1. The accuracy range is 0.82-0.85 while the f1_score range is 0.65-0.70 which considered a good performance. The highest scores are highlighted with yellow in Table 1. I used class_weight, Random UnderSampling and Random OverSampling in an effort to balance the classes in the dataset. However, class_weight and Random UnderSampling drop the accuracy and f1_score while Random OverSampling and PCA did not change the performance much.

Table 1: Logistic regression classifier results

Logistic Regression									
C	1	1	0.5	1	1	1	0.5	0.5	0.5
class_weight	-	{0:1,1:2}	{0:1,1:3}	-	-	-	-	-	-
max_iter	300	300	400	400	400	400	300	300	300
Random UnderSampling	-	-	-	-	-	Yes	-	-	-
Random OverSampling	-	-	-	-	Yes	-	-	-	-
PCA	-	-	-	-	-	-	-	30	-
Accuracy	0.85	0.84	0.82	0.85	0.85	0.83	0.85	0.85	0.85
F1_score	0.66	0.70	0.69	0.66	0.66	0.83	0.66	0.65	0.66
Miner2 Score	0.85	0.84	0.81	-	-	-	-		0.85

Best performance is highlighted with yellow

2. **Decision Tree:** I have used gini and entropy to construct tree, however, the entropy was slower than gini. I have also implemented Random UnderSampling, Random OverSampling and PCA, but the performance decreased in case of Random OverSampling and PCA and did not change in case of Random UnderSampling. To find the optimal max depth of the tree, I started with 6 and gradually increased the value. The peak performance was found at max depth 10 as shown in Table 2. The accuracy range is 0.82-0.86 which is good.

Table 2: Decision tree classifier results

Decision Tree							
max_depth	6	6	6	10	20	30	10
criterion	gini	gini	gini	entropy	entropy	gini	Gini
Random UnderSampling	-	Yes	-	-	-	-	-
Random OverSampling	-	-	Yes	-	-	-	-
PCA	-	-	-	30	-	-	-
Accuracy	0.85	0.85	0.83	0.83	0.84	0.82	0.85
F1_Score	0.65	0.65	0.84	0.63	0.66	0.62	0.66
Miner2 Score	-	-	-	-	-	-	0.86

Best performance is highlighted with yellow

3. **Adaboost**: I decided to not use Random UnderSampling, Random OverSampling and PCA in Adaboost, Random Forest, SVM and KNN because in the previous two classifiers they did not improve the performance. In Adaboost classifier, I started with n_estimators=100 and increased the number gradually until it reaches the peak at 500 as shown in Table 3. This classifier achieved the highest performance with 0.86 accuracy and 0.70 f1_score.

Table 3: Adaboost classifier results

Adaboost						
n_estimators	100	200	300	400	500	1000
Accuracy	0.86	0.87	0.87	0.87	0.87	0.86
F1_Score	0.69	0.69	0.70	0.70	0.70	0.67
Miner2 Score	-	-	0.86	0.86	-	-

Best performance is highlighted with yellow

4. **Random forest:** I used the same strategy that I adapt in Adaboost.

Table 4: Random Forest classifier results

Random Forest					
n_estimators	300	400	500	1000	2000
max_depth	6	6	10	20	30
Accuracy	0.85	0.85	0.86	0.86	0.85
F1_Score	0.62	0.62	0.66	0.68	0.67
Miner2 Score	-	0.83	-	0.86	-

Best performance is highlighted with yellow

5. **In SVM**, I try to balance female and male classes with class weight, but the performance dropped. I have tried linear and kernel SVM, and rbf kernel performed better than linear. Moreover, I have tuned the C parameter to prevent classifier from overfitting.

Table 5: SVM classifier results

SVM					
C	0.2	1	0.5	0.5	0.5
Kernel	rbf	linear	rbf	rbf	Rbf
class_weight	{0:1,1:2}	{0:1,1:3}	-	-	-
PCA	-	-	-	30	-
Accuracy	0.84	0.80	0.86	0.86	0.86
F1_score	0.70	0.68	0.67	0.67	0.67
Miner2 Score	0.84	0.80	0.86	0.86	0.86

Best performance is highlighted with yellow

6. **In KNN** I used a loop with a step of 5 to find the optimal value for K.

Table 6: KNN classifier results

KNN								
No. Of Neighbor	10	20	30	35	40	45	50	55
Accuracy	0.84	0.85	0.85	0.85	0.85	0.85	0.85	0.85
F1_Score	0.64	0.65	0.66	0.66	0.66	0.66	0.66	0.66
Miner2 Score	0.84	-	-	-	-	-	-	0.85

Best performance is highlighted with yellow

Part 2 Fairness Diagnosis: In table 7, Fairness Criteria are computed for sensitive attributes (Gender and Race). The best performance classifiers from part 1 were used and the demographic disparity, equality of opportunity, and equality of odds were calculated. Demographic disparity measures the independence of sensitive attributes from the classifier (0 means independent). According to data in Table 7, it is apparent that all classifiers are not independent of the Gender and Race attributes because the scores of Demographic Disparity are 0.17-0.19 for Gender and 0.07-0.10 for Race. The Equality of Opportunity requires that the desirable outcome C=1 should be equal across groups; however, in table 7 we can see that the desirable outcome (C=1) is not equal across groups. The Equality of Odds for Y=0 requires that the false-positive rates are equal across groups S=0 and S=1. In the case of Gender, the Equality of odds difference between groups ranged from 0.04-0.05. In the case of Race and Y=0, the Equality of odds difference is almost zero across classifiers. In conclusion according to Table 7, Gender and Race made the classifiers unfair across groups and need a fairness mitigation strategy. The likely source of biases are the ground truth function between features and observed outcome differs across groups causing the classifiers to work against the minority group, and the dataset has historical biases.

Table 7: Fairness Criteria

Sensitive Attribute	Best Classifier Accuracy/ F1_score Fairness Criteria	**Logistic regression (0.85/ 0.66)**	**Decision tree (0.85/ 0.66)**	**Adaboost (0.87/ 0.70)**	**Random Forest (0.86/ 0.68)**	**SVM (0.86/ 0.67)**	**KNN (0.85/ 0.66)**
Gender	Demographic Disparity	0.19	0.17	0.19	0.17	0.17	0.18
	Average Equality of Opportunity	0.12	0.12	0.13	0.13	0.12	0.12
	Equality of Opportunity difference	0.13	0.13	0.14	0.14	0.13	0.14
	Average Equality of Odds	0.04	0.04	0.04	0.04	0.04	0.05
	Equality of odds difference	0.05	0.04	0.04	0.04	0.04	0.05
Race	Demographic Disparity	0.09	0.07	0.09	0.09	0.08	0.10
	Average Equality of Opportunity	0.12	0.12	0.12	0.12	0.12	0.12
	Equality of Opportunity difference	0.08	0.07	0.08	0.08	0.07	0.08
	Average Equality of Odds	0.12	0.12	0.12	0.12	0.11	0.12
	Equality of odds difference	0.01	0.00	0.01	0.01	0.01	0.01

Best performance is highlighted with yellow

Part 3 Fairness Mitigation:

The best performing classifier Adaboost was implemented and the demographic disparity, equality of opportunity and equality of odds were calculated after removing sensitive attributes. The scores of demographic disparity, equality of opportunity and equality of odds were better after removing sensitive attributes Gender and Race compared to not removing them which means the fairness is increased between groups in the sensitive attributes; however, the accuracy of classifier did not change. Moreover, the scores of fairness criteria improved even more when we delete the sensitive attributes (Gender and Race) and the attributes that correlate the most with sensitive attributes (relationship and marital status) compared to remove only gender and race; however, I observed an accuracy-fairness trade-off because the accuracy and F1_Score reduced as shown in Table 8.

Table 8: Fairness Criteria After Removing Sensitive Attributes

Sensitive Attribute	**Adaboost Classifier** / Fairness Criteria		**Accuracy**	**F1_score**
After removing sensitive attributes	Gender Demographic Disparity	0.17	0.87	0.70
	Gender Average Equality of Opportunity	0.12		
	Gender Equality of Opportunity difference	0.13		
	Gender Average Equality of Odds	0.04		
	Gender Equality of odds difference	0.04		
	Race Gender Demographic Disparity	0.08		
	Race Average Equality of Opportunity	0.12		
	Race Equality of Opportunity difference	0.08		
	Race Average Equality of Odds	0.12		
	Race Equality of odds difference	0.00		
After Removing sensitive attributes and correlated features	Gender Demographic Disparity	0.09	0.84	0.61
	Gender Average Equality of Opportunity	0.10		
	Gender Average Equality of Opportunity difference	0.10		
	Gender Average Equality of Odds	0.04		
	Gender Equality of odds difference	0.00		
	Race Demographic Disparity	0.07		
	Race Average Equality of Opportunity	0.10		
	Race Average Equality of Opportunity difference	0.06		
	Race Average Equality of Odds	0.10		
	Race Equality of odds difference	0.00		