DATA MINING HW 5

Tejas -2000902539

Question 1 A

- And b)

Q1(a) a) The class having a minus value $C_1(-1)=5$

The class having a positive value $C_2(+)=4$

$\therefore P(C_1) = 5/9 \quad P(C_2) = 4/9$

Entropy $(t) = -\sum p(j|t) \log p(j|t)$

$= -(4/9)\log_2(4/9) - (5/9)\log_2(5/9)$

$= 0.991076061$

b) For attribute a_1 the associating increments & probabilities are:

a_1	+	-
T	3	1
F	1	4

Entropy for $a_1 = 4/9\left[-3/4 \log(3/4) - (1/4)\log(1/4)\right] + 5/9\left[(-1/5)\log(1/5) - (4/5)\log(4/5)\right] = \underline{0.7616}$

Information gain for $a_1 = 0.9911 - 0.7616$
$= 0.2294$

For attribute a_2 the associating increments & probabilities are:

a_2	+	-
T	2	3
F	2	2

Question 1A

b)(continued) and c)

(2)

Entropy for $a_2 = \frac{5}{9}\left[\left(-\frac{2}{5}\right)\log\left(\frac{2}{5}\right) - \left(\frac{3}{5}\right)\log\left(\frac{3}{5}\right)\right] + \frac{4}{9}\left[\left(-\frac{2}{4}\right)\log\left(\frac{2}{4}\right) - \left(\frac{2}{4}\right)\log\left(\frac{2}{4}\right)\right] = \underline{0.9839}$

Info gain $= 0.9911 - 0.9839 = \underline{0.0072}$.

c)

Index	1	3	4	5	6	7
Split	0.5	2	3.5	4.5	5.5	6.5
	≤ >	≤ >	≤ >	≤ >	≤ >	≤ >
p(+)	0 4	1 3	1 3	2 2	2 2	3 1
p(−)	0 5	0 5	1 4	1 4	3 2	3 2

Index	8	9
Split	7.5	8.5
	≤ >	≤ >
	4 0	4 0
	4 1	5 0

$\text{Entropy}(1) = P(\le 0.5)\,\text{Entropy}(0,0) + P(>0.5)\,\text{Entropy}(4,5)$
$= 0 + \frac{9}{9}\left[-\left(\frac{4}{9}\right)\log_2\left(\frac{4}{9}\right) - \left(\frac{5}{9}\right)\log_2\left(\frac{5}{9}\right)\right]$
$= \underline{0.9911}$.

Gain $= 0.9911 - 0.9911 = \underline{0}$

$\text{Entropy}(3) = P(\le 2)\,\text{Entropy}(1,2) + P(>2) \times \text{Entropy}(3,5)$
$= \frac{1}{9}\left[-\frac{1}{1}\log\left(\frac{1}{1}\right) - \frac{0}{1}\log\left(\frac{0}{1}\right)\right] + \frac{8}{9}\left[-\frac{3}{8}\log\left(\frac{3}{8}\right) - \frac{5}{8}\log\left(\frac{5}{8}\right)\right] = \underline{0.8484}$

Question 1A

c) (continued)

(3)

Q1 c)

gain (3) = 0.9911 − 0.8484 = 0.142

Entropy (4) = P(≤3.5) Entropy (1,1) + P(>3.5) Entropy (3,4)

$= 2/9 \left[-1/2 \log(1/2) - (1/2)\log(1/2)\right] + 7/9 \left[-3/7 \log(3/7) - 4/7 \log(4/7)\right] = 0.9885$

gain (4) = 0.9911 − 0.9885 = 0.0026

Entropy (5) = P(≤4.5) Entropy (2,1) + P(>4.5) Entropy (2,4)

$= 3/9 \left[-2/3 \log(2/3) - 1/3 \log(1/3)\right] + 6/9 \left[-2/6 \log(2/6) - 4/6 \log(4/6)\right] = 0.9183$

gain (5) = 0.9911 − 0.9183 = 0.0728

Entropy (6) = P(≤6.5) Entropy (2,3) + P(>5.5) Entropy (2,2)

$= 5/9 \left[-(2/5)\log(2/5) - (3/5)\log(3/5)\right] + 4/5 \left[(-2/4)\log(2/4) - (2/4)\log(2/4)\right] = 0.9839$

gain (6) = 0.9911 − 0.9839 = 0.0072

Entropy (7) = P(≤6.5) Entropy (3,3) + P(>6.5) Entropy (1,2)

$= 6/9 \left[-(3/6)\log(3/6) - (3/6)\log(3/6)\right] + 3/9 \left[(-1/3)\log(1/3) - (2/3)\log(2/3)\right] = 0.9728$

Question 1

c) (continued) and d)

(4)

Gain (7) = 0.9911 − 0.9728 = 0.0183.

Entropy (8) = P(≤7.5) Entropy (4,4) + P(>7.5) Entropy (0,1)

$= 8/9 \left[-(4/8)\log(4/8) - (4/8)\log(4/8) \right] - 1/9 \left[-(0/1) \times \log(0/0) - (1/1)\log(1/1) \right] = 0.8889$

Gain (8) = 0.9911 − 0.8889 = 0.1022

d) From the Calculations above we got,

Gain (a_1) = 0.2296

Gain (a_2) = 0.0072

Best of a_3 Gain (a_3) = 0.1428

Ans From the above gain values the Gain (a_1) has high value. Hence, a_1 provides the best split among the three.

Ans: Gain (a_1)

Question 1

e)

Q1

e) Attribute — Error Rate

Attribute	Error Rate
a_1	2/9
a_2	4/9.

$\therefore$ a_1 gives best split according to the rate

Question 1 A f) and Question 1 B a)

f). For the attribute a_1 the gain index is given by

$$4/9\left[1-(3/4)^2-(1/4)^2\right]+\frac{5}{9}\left[1-\left(\frac{1}{5}\right)^2-\left(\frac{4}{5}\right)^2\right]$$

$= 0.3444$

For the attribute a_2 the gain index is given by.

$$\frac{5}{9}\left[1-\left(\frac{2}{5}\right)^2-\left(\frac{3}{5}\right)^2\right]+\frac{4}{9}\left[1-\left(\frac{2}{4}\right)^2-\left(\frac{2}{4}\right)^2\right]$$

From above we can conclude that a_1 will give a better split.

Q1 (b) a) The contingency tables after splitting a attributes A & B are:

A	T	F
+	4	0
−	3	3

B	T	F
+	3	1
−	1	5

Overall Entropy Before Splitting is

$E_{overall} = -0.4 \log(0.4) - 0.6 = 0.9710$

The information gain after splitting A:

$E_{A=T} = -\frac{4}{7}\log\frac{4}{7} - \frac{3}{7}\log\frac{3}{7} = 0.9852$

~~The information gain~~

$E_{A=F} = -\frac{3}{3}\log\frac{3}{3} - \frac{0}{3}\log\frac{0}{3} = 0$

Question 1 B) a) and b)

(6)

$Gain = E_{overall} - \frac{7}{10} E_{A=T} - \frac{3}{10} E_{A=F} = 0.2813$

Information gain after splitting B :

$E_{B=T} = -3/4 \log 3/4 - 1/4 \log 1/4 = 0.8113$

$E_{B=F} = -1/6 \log 1/6 - 5/6 \log 5/6 = 0.650$

$Gain = E_{overall} - 4/10\, E_{B=T} - \frac{6}{10} E_{B=F} = \cancel{0.21088}$

$= 0.21565$

∴ A has high information gain

∴ choose A to split the nodes.

(b) Overall gain before splitting

$G_{overall} = 1 - (0.4)^2 - (0.6)^2 = 0.48$

The info gain after splitting A is :

$G_{A=T} = 1 - (4/7)^2 - (3/7)^2 = 0.4898$

$G_{A=F} = 1 - (3/3)^2 - (0/3)^2 = 0$

$Gain = G_{overall} - 7/10\, G_{A=T} - 3/10\, G_{A=F} = 0.1371$

the gain in gini after splitting on B :-

(7)

(8)

Q1 (b) (b) $g_{B=T} = 1-(1/4)^2-(3/4)^2 = \underline{\underline{0.3750}}$

$g_{B=F} = 1-(1/6)^2-(5/6)^2 = \underline{\underline{0.2778}}$

$Gain = Gain_{overall} - (4/10)\, g_{B=T} - 6/10\, g_{B=F} = \underline{\underline{0.1633}}$

Attribute B is choose to split the node.

Question 1 B) part c)

Q1

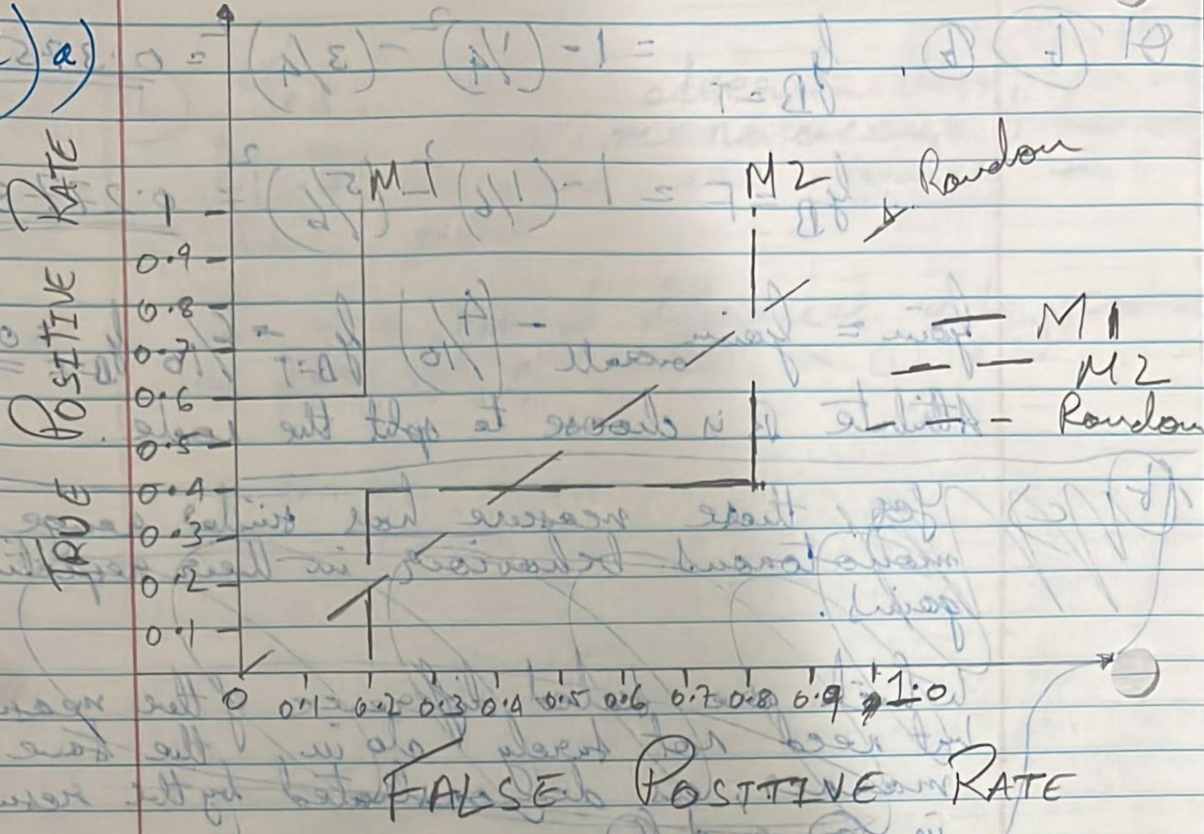
b) (c) Yes, We observe that range and monotonous behaviour of the measures are <u>similar</u> in their gains.

The are scaled differences of the measures as proved by results in (1) & (2)

Question 1 C) part a) and b)

8

C) a)



b) When $t = 0.5$, the Confusion Matrix for M1 is shown below:

Actual	+	—
+	3	2
—	1	4

Precision $= 3/4 = 75\%$

Recall $= 3/5 = 60\%$

F-measure $= (2 \times 0.75 \times 0.6) / (0.75 + 0.6)$

$= \underline{0.667}$

Question 1 C) part c) and d)

(9)

c) When $t=0.5$, the Confusion matrix for M2 is shown:

Actual	+	−
+	1	4
−	1	4

Precision $= 1/2 = 50\%$

Recall $= 1/5 = 20\%$

F-measure $= (2 \times 0.5 \times 0.2) / (0.5 + 0.2) = \underline{0.2857}$

Based on F-measure M1 is still better than M2. This result is consistent with the ROC plot.

d) When $t=0.1$, then Confusion Matrix for M1 is shown

Actual	+	−
+	5	0
−	4	1

Precision $= 5/9 = 55.55\%$

Recall $= 5/5 = 100\%$

F-measure $= (2 \times 0.5555 \times 1) / (0.5555 + 1)$
$= 0.715$

According to F-measure, $t=0.1$ is better than $\underline{t=0.5}$.

Question 1 C) part D)

From ROC curve, we need ~~to~~ to find which threshold is better.
We use the table for that.
We notice that when $t = 0.1$, $FPR = 0.8$ & $TPR = 1$:

when $t = 0.5$, $FPR = 0.2$ & $TPR = 0.6$.
We favor $t = 0.5$, because we choose $(0.2, 0.6)$ to point $(0, 1)$

Since F-measure & ROC are different Ways of evaluating the performance of classification. The choices of thresholds based on them are inconsistent.

We can prove this by calculating ~~the~~ Area Under ROC Curve.

For $t = 0.5$, area $= 0.6(1-0.2) = 0.48$
$t = 0.1$, area $= 1 \times (0.2) = 0.2$

$\therefore$ Area $(t = 0.5) >$ Area $(t = 0.1)$

$\therefore$ We prefer $t = 0.5$

QUESTION 2

```
In [120]: import pandas as pd
          import numpy as np
          from sklearn import preprocessing
          from sklearn.model_selection import train_test_split
          from sklearn import linear_model
          import random
```

```
In [121]: collect_1 = pd.read_csv('auto-mpg.csv')
```

```
In [122]: collect_1.dtypes
```

```
Out[122]: mpg             float64
          cylinders         int64
          displacement    float64
          horsepower       object
          weight            int64
          acceleration    float64
          model year        int64
          origin            int64
          car name         object
          dtype: object
```

```
In [123]: collect_1.isnull().sum() # We get all null values number and get to see all columns at the same time
```

```
Out[123]: mpg             0
          cylinders       0
          displacement    0
          horsepower      0
          weight          0
          acceleration    0
          model year      0
          origin          0
          car name        0
          dtype: int64
```

```
In [124]: collect_1 = collect_1.replace('?', 0)
          collect_1['horsepower'] = collect_1['horsepower'].astype(float, errors = 'raise')
          collect_1.loc[collect_1['horsepower']==0]

          # Done to change horsepower dtype to string
```

Out[124]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
32	25.0	4	98.0	0.0	2046	19.0	71	1	ford pinto
126	21.0	6	200.0	0.0	2875	17.0	74	1	ford maverick
330	40.9	4	85.0	0.0	1835	17.3	80	2	renault lecar deluxe
336	23.6	4	140.0	0.0	2905	14.3	80	1	ford mustang cobra
354	34.5	4	100.0	0.0	2320	15.8	81	2	renault 18i
374	23.0	4	151.0	0.0	3035	20.5	82	1	amc concord dl

```
In [125]: # L2 norm used here

          dfa = collect_1.drop('car name',axis=1)
          x=dfa.drop('mpg',axis=1) # Independent variable
          y= dfa['mpg'] # Dependent variable
          X_train,X_test,y_train,y_test = train_test_split(x,y,test_size=0.333)

          # In this step we split the dataset into X train, Xtest, Y train , Y test.
          #This will help us partition data for training and testing purposes. In ration 2/3 for training and 1/3 for testing
```

In [126]:
```
X_train
```

Out[126]:

	cylinders	displacement	horsepower	weight	acceleration	model year	origin
114	4	98.0	90.0	2265	15.5	73	2
141	4	98.0	83.0	2219	16.5	74	2
209	4	120.0	88.0	3270	21.9	76	2
324	4	85.0	65.0	2110	19.2	80	3
334	3	70.0	100.0	2420	12.5	80	3
...	...	...	...	...	...	...	...
319	4	120.0	75.0	2542	17.5	80	3
243	3	80.0	110.0	2720	13.5	77	3
54	4	72.0	69.0	1613	18.0	71	3
363	6	231.0	110.0	3415	15.8	81	1
50	4	116.0	90.0	2123	14.0	71	2

265 rows × 7 columns

In [127]:
```
from sklearn.preprocessing import Normalizer      # Inbuilt
nm = Normalizer(norm = 'l2')
X_train = nm.fit_transform(X_train)
X_test = nm.transform(X_test)

reg1 = linear_model.LinearRegression()
reg1.fit(X_train,y_train)
```

Out[127]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [128]:
```
reg1.score(X_train,y_train)
```

Out[128]:
```
0.8106892652374971
```

In [129]:
```

dfb = pd.DataFrame(reg1.coef_).T
dfb = dfb.rename(columns={0: "cylinders", 1: "displacement",2:"horsepower",3:"weight",4:"acceleration",5:"model year",6:"origin"
dfb # to casually check result of perations
```

Out[129]:

	cylinders	displacement	horsepower	weight	acceleration	model year	origin
0	-2463.243733	34.63221	-121.107388	103.756931	23.256692	897.716582	684.560212

In [130]:
```
coeff1 = reg1.coef_
coeff1
```

Out[130]:
```
array([-2463.24373271,    34.63221008,  -121.10738755,   103.75693147,
          23.25669247,   897.71658161,   684.56021162])
```

In [131]:
```
from sklearn.metrics import mean_squared_error
y_pred_test = reg1.predict(X_test)
MeanSqError = mean_squared_error(y_test,y_pred_test) # Evaluating the RMSE values for the test Predictions
print('Mean Squared Error : ',MeanSqError)
```

```
Mean Squared Error :  11.147965363734231
```

In [132]:
```
np.random.seed(42)
linregCV = linear_model.RidgeCV(alphas=[0.001,0.01,0.01,10.0,100.0,1000.0,1000.0,1000.0],cv=5)
linregCV.fit(X_train,y_train)
linregCV.alpha_
```

Out[132]:
```
0.001
```

In [133]:
```

linridgeCV = linear_model.Ridge(alpha=0.01,normalize=True)
linridgeCV.fit(X_train,y_train)
linridgeCV.score(X_train,y_train)
```

Out[133]:
```
0.8100414202727876
```

In [134]:
```
coeff2 = linridgeCV.coef_
coeff2
```

Out[134]:
```
array([-1862.05870361,    -2.78485176,  -121.26904217,  -192.53626176,
         155.07161376,   815.58110412,   723.30396269])
```

In [135]:
```
y_predCV = linridgeCV.predict(X_test)
y_predCV
```

Out[135]:
```
array([18.82089475, 22.20142451, 32.93433985, 29.12992204, 34.40439588,
       16.15686523, 35.98570326, 24.6702725 , 13.05066821, 28.27914723,
       17.69527381, 14.3653996 , 19.06268785, 29.77670724, 21.55701675,
       21.20108778, 26.68353037, 19.53842324, 14.85249273, 18.96391254,
       13.30009791, 32.37638235, 26.10020453, 20.17293198, 12.9026615 ,
       25.68679137, 12.87679285, 22.91169824, 20.75546507, 33.11613554,
       25.09313258, 22.71353991, 36.22569511, 26.30480056, 27.25699015,
       14.65323276, 34.2190685 , 21.03763447, 12.97466391, 31.00770584,
       27.87744128, 34.60909857, 24.61774644, 28.62980162, 20.40806859,
       21.68366457, 14.46112081, 26.6164932 , 25.52804388, 16.70094299,
       35.55451661, 20.5130999 , 31.8536226 , 25.43043032, 15.77139429,
       33.66915676, 20.36376985, 15.1305317 , 30.61352133, 33.6969068 ,
       23.49827639, 14.26827474, 36.37211475, 16.97022666, 20.09284987,
       27.92379774, 11.56753636, 22.75384736, 39.29044505, 22.77231677,
       12.13499068, 18.49387136, 35.21185391, 23.62652728, 34.1774882 ,
       14.59192115, 25.62018657, 24.9363117 , 26.00934943, 12.17015936,
       35.6276298 , 12.15009186, 14.95751601, 26.64127025, 13.72828508,
       21.69794357, 22.34852071, 15.12149455, 15.82788143, 29.72718369,
       26.62172169, 28.76264226, 32.91914692, 25.50804076, 12.06586386,
       18.98985721, 33.37058142, 19.99687327, 31.54169836, 24.38350852,
       25.48352886, 12.43446771, 19.708025  , 28.26014989, 16.12542537,
       35.62819476, 20.17006565, 20.73600666, 19.76856376, 34.18021998,
       25.89770793, 15.3012844 , 27.53332277, 24.94111891, 25.52104823,
       24.74357526, 30.21335818, 13.30410736, 26.2322031 , 18.91942104,
       27.81109364, 27.91793933, 14.42363599, 17.59737897, 14.39051371,
       12.14968458, 12.02573463, 20.47229663, 26.63786421, 27.54586735,
       26.09670944, 24.56849746, 28.87659916])
```

In [136]:
```python
MeanSqError = np.square(np.subtract(y_test,y_predCV)).mean()
MeanSqError
```
Out[136]:
```
10.078905193059327
```
In [137]:
```python
np.random.seed(42)
lassoCV=linear_model.LassoCV(alphas=[0.0001,0.001,0.01,0.1,1],max_iter=10000,cv=10)
lassoCV.fit(X_train,y_train)
y_predlasso=lassoCV.predict(X_test)
```
In [138]:
```python
lassoCV.alpha_
```
Out[138]:
```
0.0001
```
In [139]:
```python
LCV = linear_model.Lasso(alpha=0.001,normalize=True)
LCV.fit(X_train,y_train)
LCV.score(X_train,y_train)
```
Out[139]:
```
0.8104857786484225
```
In [140]:
```python
coeff3 = LCV.coef_
coeff3
```
Out[140]:
```
array([-1868.66908674,     5.90717546,  -122.53958782,   -65.47712536,
           0.        ,   871.95693234,   496.50973391])
```
In [141]:
```python
y_predlasso=lassoCV.predict(X_test)
```
In [142]:
```python
MeanSqError=np.square(np.subtract(y_test,y_predlasso)).mean()
```
In [143]:
```python
coeff1
```
Out[143]:
```
array([-2463.24373271,    34.63221008,  -121.10738755,   103.75693147,
          23.25669247,   897.71658161,   684.56021162])
```
In [144]:
```python
coeff2
```
Out[144]:
```
array([-1862.05870361,    -2.78485176,  -121.26904217,  -192.53626176,
         155.07161376,   815.58110412,   723.30396269])
```
In [145]:
```python
coeff3
```
Out[145]:
```
array([-1868.66908674,     5.90717546,  -122.53958782,   -65.47712536,
           0.        ,   871.95693234,   496.50973391])
```

Q 2

(e)

Coefficients value is reduced by Regularization therefore it directly effects the importance of an attribute. Linear Regression and Ridge reduce value of coefficient that are not important, but LassoCV converts coefficients that are not valueable to 0.

Since Cross Validaton reduces the Mean Square Error [MSE] , therefore MSE is a guide which indicates best result. Hence LassoCV gives the best result going by MSE followed by RidgeCV and Linear Regression

Question 3)

The authors take cognizance of the pervasiveness of AI in our life. This leads to various societal implications such as bias in AI decisions. The authors have extrapolated their understanding of the situation by surveying all research papers in this field. The authors are of the opinion that bias is as "old as human civilization" and that its difficult to tackle. But AI may amplify the entire bias as it has no mind of its own, it completely relies on the coders, designers, dataset fed etc. The bias in data can manifest due to sensitive features and casual influences, representativeness of data and data modalities.

It is imperative to tackle bias at every stage in order to combat it effectively. The authors have framed the idea of how to tackle bias by telling us about how to understand the bias, mitigate the bias, accounting for bias and the legal frameworks that are present, how they can be improved.

In Socio-technical cause of bias the authors tell us that the bias is present as many of the datasets are made by humans. As human society is already biased (sexism, institution bias, representation bias) all this bias might have crept in and must be getting reflected in the data. Hence when we use algorithms in institutions they just carry this bias forward. This was seen in real as researchers found Google ads showing less high paying posts to women, despite them being capable of better, this is due to existing gender misrepresentation and sexism. Authors also tell us that correlated features can also lead to bias they give us a example of how us districts are linked to racial representation, hence using district as a measure for loans,insurance etc indirectly puts them at a disadvantage. Authors also comment on how over and under representation in data leads to bias and how different types of data like language, images also have bias in them. They also comment on how ambiguous the meaning of fairness is.

The authors then tell us the survey results and observation on how to mitigate bias. They give us 3 approaches

- Pre-processing approach: This aims at making a balanced dataset which when used for any type of learning by an algorithm can ensure unbiased results.
- In-processing approaches: Here many researchers have tried to tackle bias by incorporating it into the algorithm through regularization, constraints, latent target variable. There are approaches for unsupervised learning like fair-PCA which has equal representation of every group in the cluster.
- Post-processing approaches: After the predictions by the model, we take action required by changing the test values or predicted values.Some times we also change their internals.

Legal issues: The authors inform us on how if we perform pre- and in- processing approach there might be an Intellectual property issue. Authors also comment on how GDPR enforces users consent before their personal data can be used for analytics. This might affect model alteration.

Accounting for bias tells us about how to handle AI bias by not codifying the solution but by rather using ML algorithms and complex data. Authors extrapolate on two facets:

- Proactively: There are many ways of collecting data. Each will have its own bias and we must be aware of it, like crowdsourcing data which itself might have bias present, mitigating bias in it is hard. There are set ways told by many researchers to mitigate bias at this stage but they were later found to be biased themselves.

- Retroactively: Tracking how an algorithm makes decision is hard as it uses many layers of neural network. This approach deals with how to make an AI explainable to anybody, this has lead to many terms like explainable AI.

Sometimes bias is useful for some insights like for using analytics to aid in discovery of new drugs, treatments etc. Laws are limited scope on antidiscrimination law, hence law has a lot to catch up.

In my opinion since AI depends a lot on Human created data, features and expectations. It will have a tilted result. Mitigating bias involves real world effects like ending sexism, ending racial discrimination. We can use scientific ways to Mitigate bias but not many companies will adopt it till its cost effective as buying data itself is an expensive affair. This process also involves educating Data scientists on how to deal with bias sensitively. This can ensure that future researchers handle data well and also create new unbiased algorithms. Governments can play a pivotal role in spurring this massive change by making laws to end prevalent biases, addressing them at root level, creating laws that act on biased output of AI and urges companies to do better. It is difficult to remove bias completely, I strongly feel we currently are on the right path and we can end a lot of the bias that is present.