**19CSE304: Foundations of Data Science**

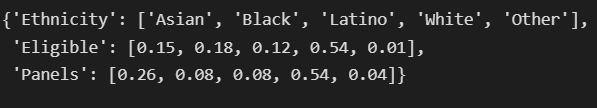
**Assignment - 3**

**Name: Laxman K R**

**Roll: CB.EN.U4CSE20234**

**jury = {"Ethnicity":["Asian","Black","Latino","White","Other"],"Eligible":[0.15,0.18,0.12,0.54,0.01],"Panels":[0.26,0.08,0.08,0.54,0.04]}**

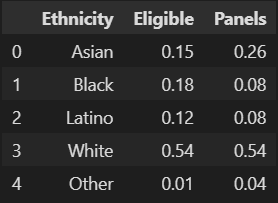
**jury**

****

Storing the data in a 2d-list

Alameda\_df = pd.DataFrame(jury)

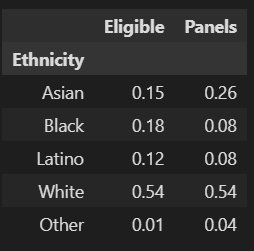
Alameda\_df



Saving the list as Dataframe

Alameda\_df\_1 = Alameda\_df.set\_index('Ethnicity')

Alameda\_df\_1

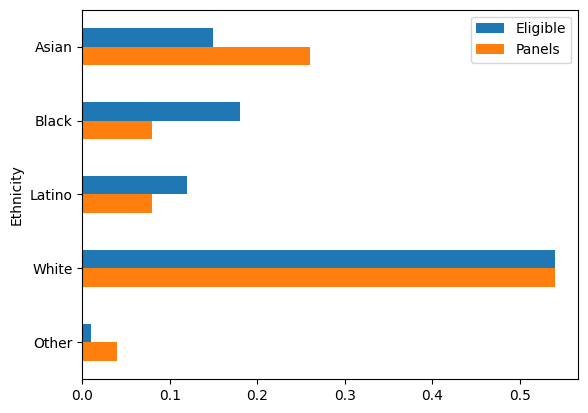


Setting Index for better visualization.

Alameda\_df\_1.plot.barh()

plt.ylabel('Ethnicity')

plt.gca().invert\_yaxis()



The difference between Eligible and panels

Alameda\_df\_1['jury\_with\_diffs'] = Alameda\_df\_1['Panels']-Alameda\_df\_1['Eligible']

Alameda\_df\_1



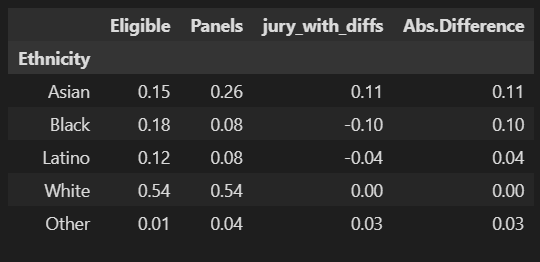
The difference between Eligible and Panels

Sum(jury\_with\_diffs)=0

Because the gain of the panel is exactly equal to the loss in the panel.

Alameda\_df\_1['Abs.Difference']=abs(Alameda\_df\_1['jury\_with\_diffs'])

Alameda\_df\_1



Calculating the Abs difference

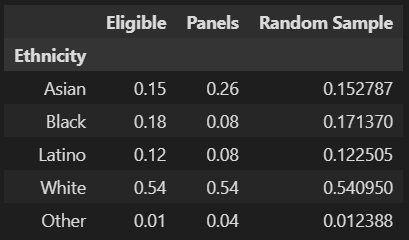
test\_statistic = Alameda\_df\_1['Abs.Difference'].sum()/2

test\_statistic

0.14

Alameda\_df\_2['Random Sample'] = npr.multinomial(1453,[0.15,0.18,0.12,0.54,0.01])/1453

Alameda\_df\_2

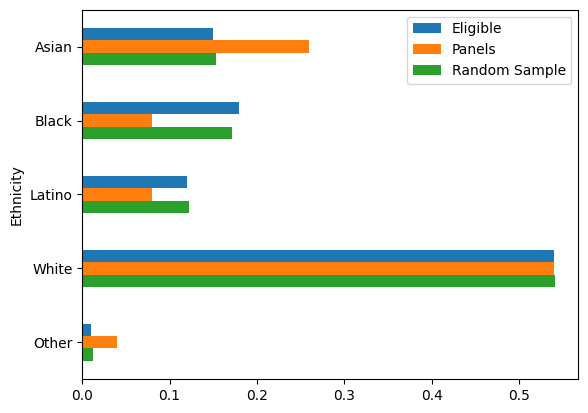


Taking Random sample with given eligibility

Alameda\_df\_2.plot.barh()

plt.ylabel('Ethnicity')

plt.gca().invert\_yaxis()



The green bars are closer in size to the blue bats than the orange bars.

TVD = (abs(Alameda\_df\_2['Eligible']-Alameda\_df\_2['Random Sample'])).sum()/2

TVD

0.00863041982105983

The difference between Eligible and Random Sample is very insignificant.

simulations = 5000

tvd\_list=[]

for i in np.arange(simulations):

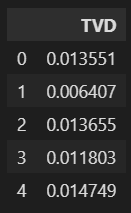
Alameda\_df\_2["Random Sample"]=(npr.multinomial(1453,[0.15, 0.18, 0.12, 0.54, 0.01]))/panel\_size

tvd\_list.append(table\_tvd(Alameda\_df\_2, 'Eligible', 'Random Sample'))

tvd\_final\_df=pd.DataFrame(tvd\_list)

tvd\_final\_df.rename(columns={0:"TVD"},inplace=True) # renaming column

tvd\_final\_df.head()



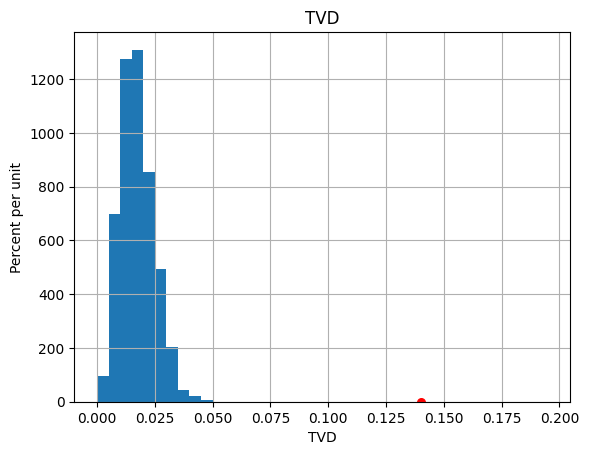
Ran 5000 simulations containing a list of 5000\*5 random entries.

tvd\_final\_df.hist(bins=np.arange(0,0.2,0.005))

plt.ylabel('Percent per unit')

plt.xlabel('TVD')

plt.scatter(observed\_stat, 0, color='red', s=30)



This percentage vs TVD shows the majority of them have a very low TVD value.

Since there is enough evidence of the Alternate hypothesis, we reject the Null hypothesis.