PROGRAMMING IN PYTHON FOR DATA SCIENCE(CS3PP19) Final Exam: Question 3

52944

April 27, 2021

1 Initialization Code

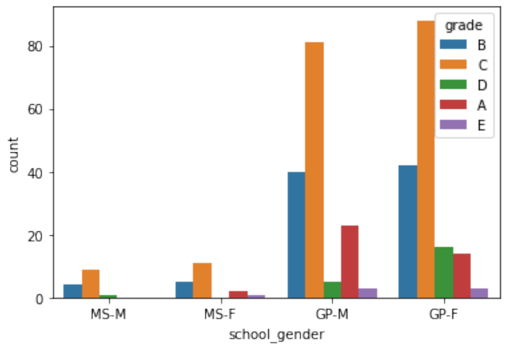
```
import pandas as pd
import seaborn as sns
data = pd.read_csv("data.csv")
data=data.dropna()
data
```

2 Q3.a.i

Trying to answer this question directly, will lead to misleading results.

```
#Creating category as cartesian product of school-gender
ms_male=data.loc[(data.school=="MS") & (data.sex=="M")].reset_index()
ms_female=data.loc[(data.school=="MS") & (data.sex=="F")].reset_index()
gp_male=data.loc[(data.school=="GP") & (data.sex=="M")].reset_index()
gp_female=data.loc[(data.school=="GP") & (data.sex=="F")].reset_index()
#Creating the new categorical variable
ms_male["school_gender"]="MS-M"
ms_female["school_gender"]="MS-F"
gp_male["school_gender"]="GP-M"
gp_female["school_gender"]="GP-F"
#Concatenate the datasets
data=pd.concat([ms_male,ms_female,gp_male,gp_female])
#plot
sns.countplot(x="school_gender",hue='grade',data=data)
```

Figure 1: Distribution of grades by school and gender



Thats because the dataset is severily undersampled, on the school categorical data, for the value MS. running a count query reveals the issue

```
"""
This dataset has issue. MS School is undersampled.
"""
data.groupby("school").school.count()
```

GP	MS
315	35

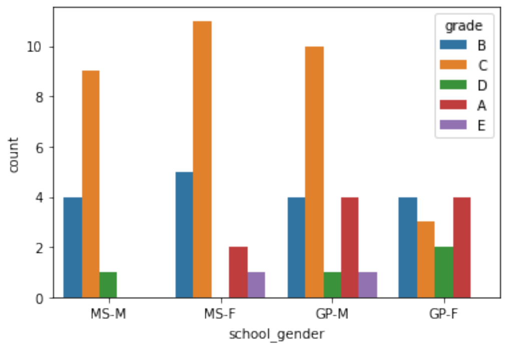
We can bypass this by sampling the oversampled class, in this case, GP class is oversampled with 315 samples.

```
ms=data.loc[data.school=="MS"]
gp=data.loc[data.school=="GP"]
sampled_gp=gp.sample(35)

sampled_balanced_data=pd.concat([ms,sampled_gp]).reset_index()
data=sampled_balanced_data
```

Now, we have a set of equal classes.

Figure 2: Distribution of grades by school and gender, sampled

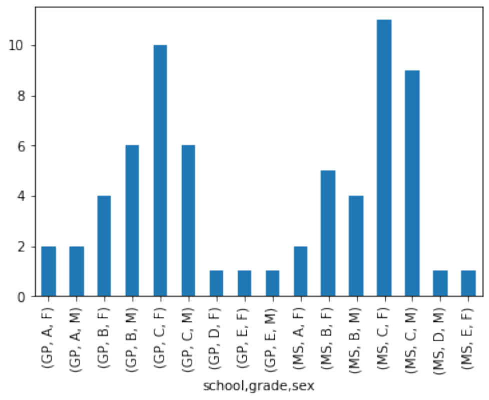


Of course this is not a perfect solution, as we loose prestision over the oversampled class. This techique should always used with care. Please note that the above task could also be achieved with the following statement

```
x=data.groupby(['school','grade','sex']).sex.count().plot.bar()
```

with the following end-result

Figure 3: Distribution of grades by school and gender, sampled



3 Q3.a.ii

```
#select rows for creating sudo-category
grade_ab=data.loc[(data.grade=="A") | (data.grade=="B")]
grade_cd=data.loc[(data.grade=="C") | (data.grade=="D")]
grade_e=data.loc[(data.grade=="E")]
#create sudo category
grade_ab["grade_category"]="AB"
grade_cd["grade_category"]="CD"
grade_e["grade_category"]="E"
#concat the dataset again
data=pd.concat([grade_ab,grade_cd,grade_e])
#find the mean . grouby the sudo category
average_study_time=data.groupby("grade_category").studytime.mean()
```

A-B	C-D	E
2.428571	1.863636	1.000000

This is misleading though, as the categories A-B and E are undersampled. We can verify this by typing...

```
data.groupby("grade_category").grade_category.count()
```

A-B	C-D	E
21	44	1

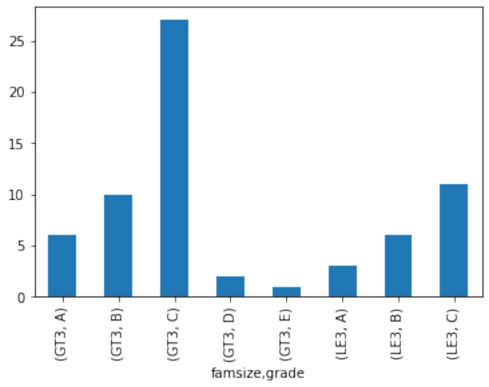
Therefore, we need to discard E from our analysis, as nothing can be done for this category(we cant sample every category to 1 point only!) Depending on our needs for accuracy, we may need to downsample the category C-D, but for the purposes of this ECD, as well as the query given, i consider this unessesary.

4 Q3.a.iii

Lets create the nessesary plots, for the family size vs grade, this can be done easily...

```
data.groupby(['famsize','grade']).count()["index"].plot.bar()
```

Figure 4: Family size vs Grades



parental education though, is sligly more complicated, as i need to define what 'parential education' is . I did that with the following metric

$$Pedu = \frac{Medu + Fedu}{8}$$

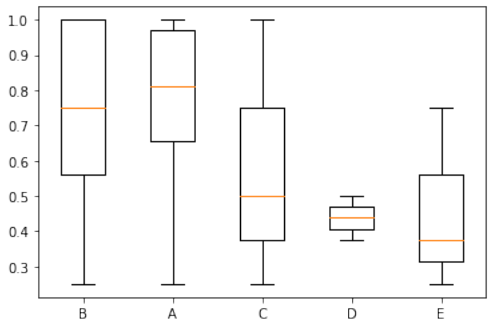
Lets then apply this row transformation to our dataset

```
data['Pedu'] = data.apply(lambda x: ((x["Medu"]+ x["Fedu"])/8,axis=1)
```

From now on, creating a boxplot is straightforward

```
datapoints = []
labels=data['grade'].unique()
for c in labels:
        datapoints.append(data.loc[data.grade==c,'Pedu'])
plt.boxplot(datapoints,labels=labels)
display
```

Figure 5: Parental education vs Grades



As we can see clearly in the respective boxplot and barchart, the parental education influences the grades of the children.

5 Q3.a.iv

In order to answer this question, we need to perform the following tasks

- Convert the grade to a **ordered** numeric variable
- Generate the corellation matrix of the data
- Pick the 'grade' column

Lets perform the first step

```
mappings={"grade":{"A":5,"B":4,"C":3,"D":2,"E":1}}
data=data.replace(mappings)
```

Here, we perform a mapping, notice the ordering of the grades. 5 (A) is better than 1 (E). Now we should generate the corellation matrix, and pick the grade column

```
bad_grades = data.corr()["grade"].sort_values()[:3]
good_grades = data.corr()["grade"].sort_values()[-5:-1] #do not pick grade variable
```

traveltime	-0.230319
Walc	-0.221261
absenses	-0.217952

Table 1: 3 best features related to bad grades

studytime	0.216986
Fedu	0.349496
Medu	0.393533
Pedu	0.419606

Table 2: 3 best features related to bad grades

The Pedu metric is included, as a proof of a quality of the metric for the question Q3.a.ii