In [106]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [107]:
```
mydata=pd.read_csv("xAPI-Edu-Data.csv")
```

In [108]:
```
mydata.head()
```

Out[108]:

emester	Relation	raisedhands	VisITedResources	AnnouncementsView	Discussion	ParentAnswering
F	Father	15	16	2	20	
F	Father	20	20	3	25	
F	Father	10	7	0	30	
F	Father	30	25	5	35	
F	Father	40	50	12	50	

In [109]:
```
mydata.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 480 entries, 0 to 479
Data columns (total 17 columns):
 #   Column                    Non-Null Count  Dtype 
---  ------                    --------------  ----- 
 0   gender                    480 non-null    object
 1   NationalITy               480 non-null    object
 2   PlaceofBirth              480 non-null    object
 3   StageID                   480 non-null    object
 4   GradeID                   480 non-null    object
 5   SectionID                 480 non-null    object
 6   Topic                     480 non-null    object
 7   Semester                  480 non-null    object
 8   Relation                  480 non-null    object
 9   raisedhands               480 non-null    int64 
 10  VisITedResources          480 non-null    int64 
 11  AnnouncementsView         480 non-null    int64 
 12  Discussion                480 non-null    int64 
 13  ParentAnsweringSurvey     480 non-null    object
 14  ParentschoolSatisfaction  480 non-null    object
 15  StudentAbsenceDays        480 non-null    object
 16  Class                     480 non-null    object
dtypes: int64(4), object(13)
memory usage: 63.9+ KB
```

In [110]:
```
mydata.columns
```

Out[110]:
```
Index(['gender', 'NationalITy', 'PlaceofBirth', 'StageID', 'GradeID',
       'SectionID', 'Topic', 'Semester', 'Relation', 'raisedhands',
       'VisITedResources', 'AnnouncementsView', 'Discussion',
       'ParentAnsweringSurvey', 'ParentschoolSatisfaction',
       'StudentAbsenceDays', 'Class'],
      dtype='object')
```

In [161]:
```
col_to_use=['gender','StageID', 'GradeID',
       'SectionID', 'Topic', 'Semester', 'Relation', 'raisedhands',
       'VisITedResources', 'AnnouncementsView', 'Discussion',
       'ParentAnsweringSurvey',
       'StudentAbsenceDays', 'Class']
```

In [162]:
```
mydata=mydata[col_to_use]
```

In [163]:
```
mydata.describe()
```

Out[163]:

	gender	StageID	GradeID	SectionID	Topic	Semester	Relation	raised
count	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000	480.000000	480.0
mean	0.635417	1.345833	2.906250	0.472917	5.256250	0.489583	0.410417	46.7
std	0.481815	0.603732	2.464267	0.612411	3.388388	0.500413	0.492423	30.7
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
25%	0.000000	1.000000	0.000000	0.000000	3.000000	0.000000	0.000000	15.7
50%	1.000000	1.000000	4.000000	0.000000	5.000000	0.000000	0.000000	50.0
75%	1.000000	2.000000	5.000000	1.000000	7.000000	1.000000	1.000000	75.0
max	1.000000	2.000000	9.000000	2.000000	11.000000	1.000000	1.000000	100.0

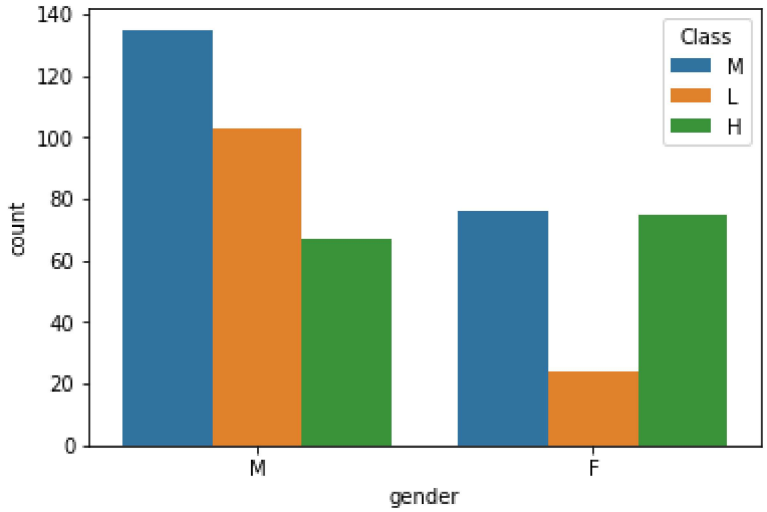
In [164]:
```
mydata.isnull().sum()
```

Out[164]:
```
gender                    0
StageID                   0
GradeID                   0
SectionID                 0
Topic                     0
Semester                  0
Relation                  0
raisedhands               0
VisITedResources          0
AnnouncementsView         0
Discussion                0
ParentAnsweringSurvey     0
StudentAbsenceDays        0
Class                     0
dtype: int64
```

Visualization of Data

In [115]:
```
sns.countplot(x="gender",data=mydata,hue="Class",order=['M','F'])
```

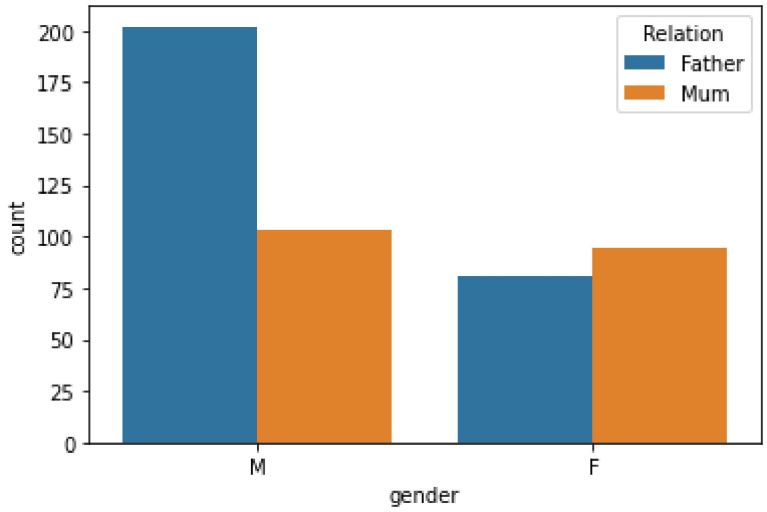
Out[115]:
```
<AxesSubplot:xlabel='gender', ylabel='count'>
```



In [116]:
```
# by seeing the above graph we can conclude that we have more engagement of males
```

In [117]:
```
sns.countplot(x="gender",data=mydata,hue="Relation",order=['M','F'])
```

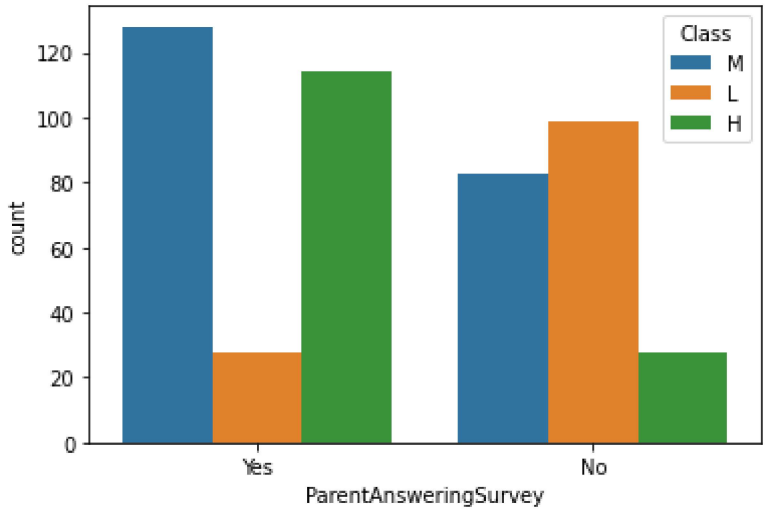
Out[117]:
```
<AxesSubplot:xlabel='gender', ylabel='count'>
```



In [118]:
```
# by seeing the above graph we can conclude children have good relation with thei
```

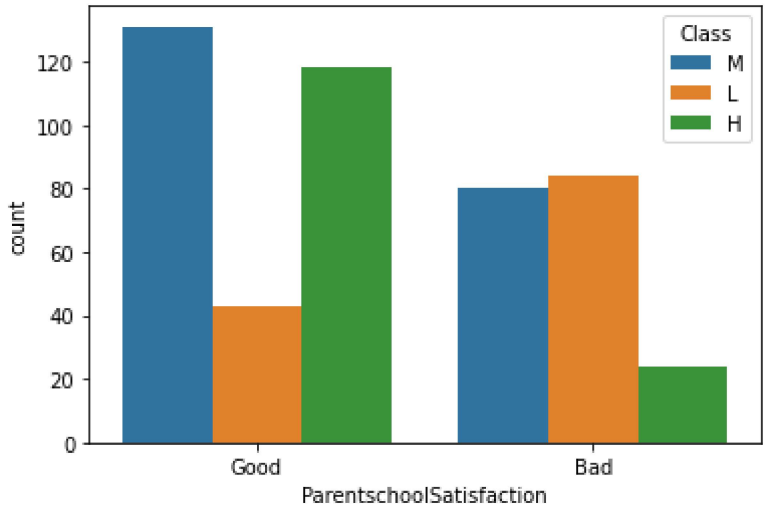
In [119]:
```
sns.countplot(x="ParentAnsweringSurvey",data=mydata,hue="Class")
```

Out[119]:
```
<AxesSubplot:xlabel='ParentAnsweringSurvey', ylabel='count'>
```



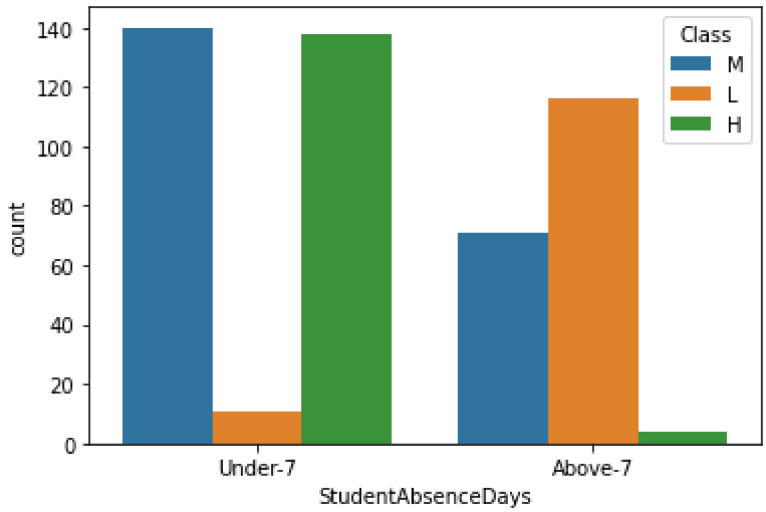
In [120]:
```
sns.countplot(x="ParentschoolSatisfaction",data=mydata,hue="Class")
```

Out[120]:
```
<AxesSubplot:xlabel='ParentschoolSatisfaction', ylabel='count'>
```

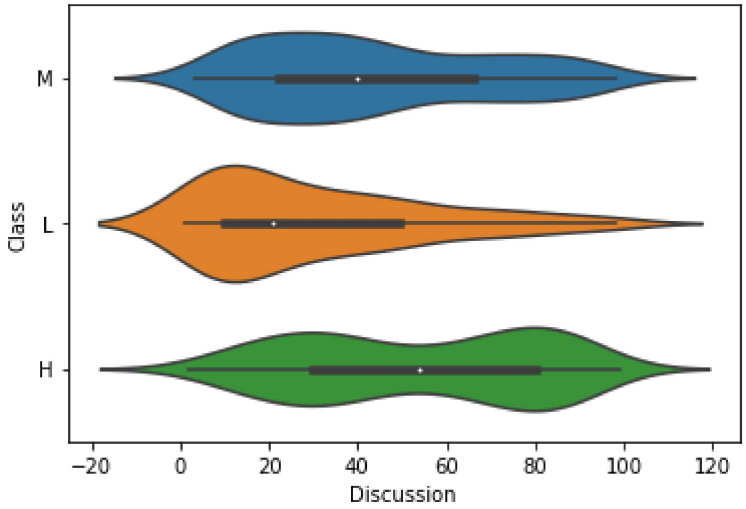


In [121]:
```
sns.countplot(x="StudentAbsenceDays",data=mydata,hue="Class")
```

Out[121]:
```
<AxesSubplot:xlabel='StudentAbsenceDays', ylabel='count'>
```

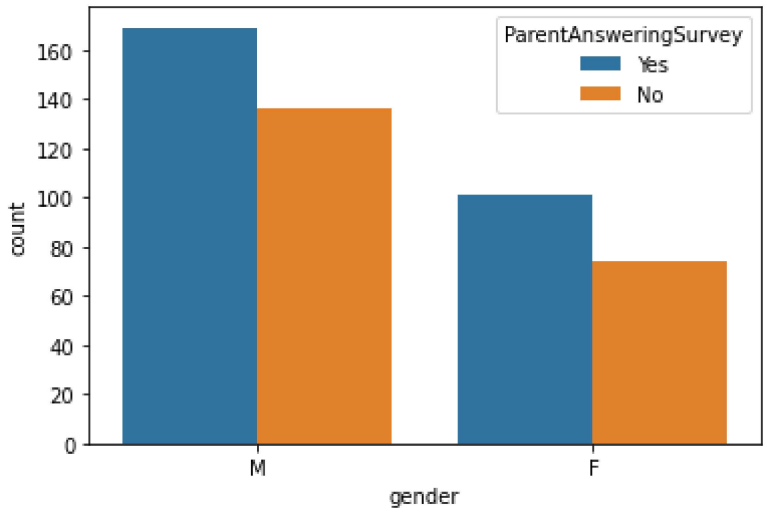


In [122]:
```
sns.violinplot(x="Discussion",y="Class",data=mydata);
```



In [123]:
```
sns.countplot(x="gender",hue="ParentAnsweringSurvey",data=mydata,order=['M','F'])
```

Out[123]:
```
<AxesSubplot:xlabel='gender', ylabel='count'>
```



In [124]:
```
# by seeing the above graphs we can conclude  that during survey we have more act
```

Label Encoder

In [148]:
```
from sklearn.preprocessing import LabelEncoder
```

In [149]:
```
LE=LabelEncoder()
```

In [150]:
```
mydata["gender"]=LE.fit_transform(mydata.gender)
```

In [151]:
```
mydata["StageID"]=LE.fit_transform(mydata.StageID)
```

In [152]:
```
mydata["GradeOD"]=LE.fit_transform(mydata.GradeID)
```

In [153]:
```
mydata["SectionID"]=LE.fit_transform(mydata.SectionID)
```

In [154]:
```
mydata["Topic"]=LE.fit_transform(mydata.Topic)
```

In [155]:
```
mydata["Semester"]=LE.fit_transform(mydata.Semester)
```

In [156]:
```
mydata["Relation"]=LE.fit_transform(mydata.Relation)
```

In [158]:
```
mydata["ParentAnsweringSurvey"]=LE.fit_transform(mydata.ParentAnsweringSurvey)
```

In [ ]:
```

```

```
mydata["StudentAbsenceDays"]=LE.fit_transform(mydata.StudentAbsenceDays)
mydata["Class"]=LE.fit_transform(mydata.Class)
mydata["GradeID"]=LE.fit_transform(mydata.GradeID)
```

correlation

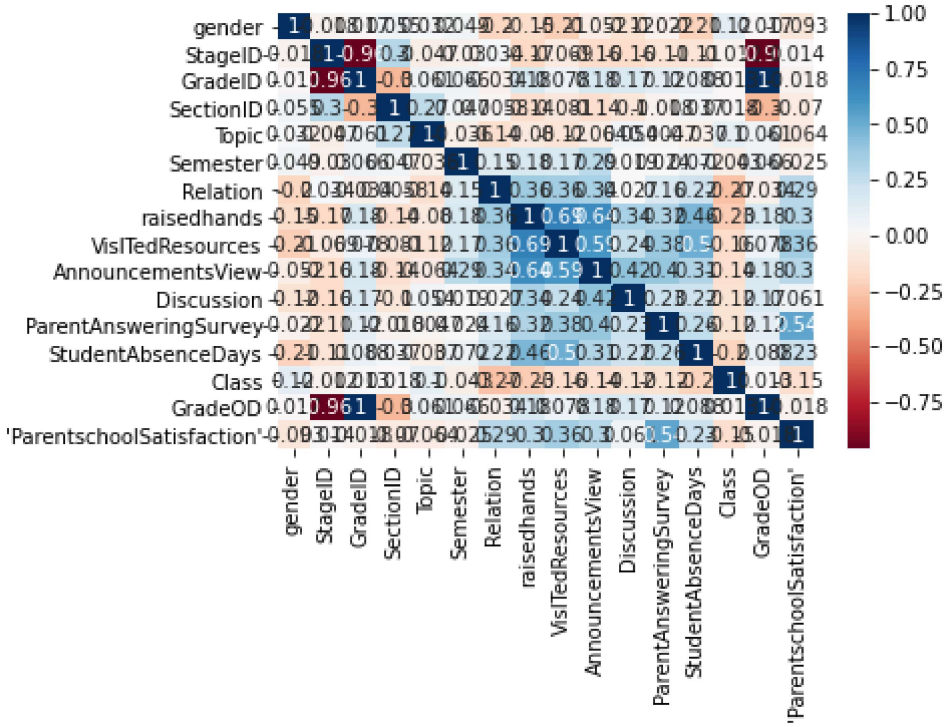
In [136]:
```
mydata_corr=mydata.corr()
mydata_corr
```

Out[136]:

	gender	StageID	GradeID	SectionID	Topic	Semester	Relatio
gender	1.000000	-0.017793	0.016869	0.054907	0.031769	0.049156	-0.19514
StageID	-0.017793	1.000000	-0.961835	0.296416	-0.047493	-0.029512	0.03420
GradeID	0.016869	-0.961835	1.000000	-0.303949	0.061389	0.066079	-0.03360
SectionID	0.054907	0.296416	-0.303949	1.000000	0.267445	0.046763	0.00578
Topic	0.031769	-0.047493	0.061389	0.267445	1.000000	-0.035975	-0.13948
Semester	0.049156	-0.029512	0.066079	0.046763	-0.035975	1.000000	0.14870
Relation	-0.195142	0.034205	-0.033602	0.005783	-0.139487	0.148705	1.00000
raisedhands	-0.149978	-0.172751	0.182621	-0.143862	-0.080418	0.178358	0.36423
VisITedResources	-0.210932	-0.068621	0.078262	-0.080909	-0.118144	0.173219	0.36024
AnnouncementsView	-0.052139	-0.163666	0.183033	-0.144955	-0.063856	0.287066	0.33950
Discussion	-0.124703	-0.161406	0.168462	-0.102538	0.054064	0.019083	0.02672
ParentAnsweringSurvey	-0.022359	-0.114025	0.118246	-0.018449	0.004730	0.023628	0.16381
StudentAbsenceDays	-0.209011	-0.112536	0.088342	0.037062	-0.036537	0.072462	0.21968
Class	0.123675	-0.011696	0.013483	0.017597	0.103610	-0.043287	-0.27211
GradeOD	0.016869	-0.961835	1.000000	-0.303949	0.061389	0.066079	-0.03360
'ParentschoolSatisfaction'	-0.093478	0.014272	-0.018421	-0.070405	-0.064087	-0.025258	0.28769

In [137]:
```
sns.heatmap(mydata_corr,annot=True,cmap="RdBu")
```

Out[137]: <AxesSubplot:>



In [138]:
```
# all the red values are highly negatively correlated whereas all the blue values
```

spliting variables

In [165]:
```
y_dep=mydata.Class
```

In [166]:
```
x_ind=mydata.drop("Class",axis=1)
```

In [167]:
```
from sklearn.model_selection import train_test_split
```

In [168]:
```
x_train,x_test,y_train,y_test=train_test_split(x_ind,y_dep,test_size=0.2,random_s
```

In [169]:
```
from sklearn.linear_model import LogisticRegression
```

Type *Markdown* and LaTeX: α^2

In [170]:
```
model1=LogisticRegression()
```

In [171]:
```
model1.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\_logistic.py:76
3: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-
learn.org/stable/modules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regressi
on)
  n_iter_i = _check_optimize_result(
```

Out[171]:
```
LogisticRegression()
```

In [172]:
```
y_pred=model1.predict(x_test)
```

In [173]:
```
y_pred
```

Out[173]:
```
array([0, 2, 0, 2, 0, 0, 1, 1, 2, 2, 1, 1, 0, 2, 1, 2, 0, 0, 2, 0, 0, 0,
       2, 1, 2, 2, 0, 1, 0, 2, 1, 0, 2, 2, 0, 1, 1, 1, 2, 1, 1, 0, 0, 0,
       2, 2, 0, 1, 0, 1, 1, 2, 0, 2, 2, 2, 0, 1, 1, 2, 0, 1, 2, 2, 2, 1,
       1, 2, 2, 0, 0, 1, 2, 2, 2, 0, 2, 1, 2, 2, 2, 2, 1, 1, 1, 2, 1, 0,
       0, 0, 1, 2, 2, 0, 0, 0])
```

confusion metrics

In [174]:
```
from sklearn.metrics import confusion_matrix,accuracy_score
```

In [175]:
```
confusion_matrix(y_test,y_pred)
```

Out[175]:
```
array([[19,  0, 15],
       [ 0, 21,  2],
       [12,  7, 20]], dtype=int64)
```

In [176]:
```
accuracy_score(y_test,y_pred)
```

Out[176]:
```
0.625
```

In [177]:
```
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve
```

In [178]:
```
logis_roc_auc=roc_auc_score(y_test,y_pred)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-178-52aa18c53738> in <module>
----> 1 logis_roc_auc=roc_auc_score(y_test,y_pred)

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py in inner
_f(*args, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64
     65             # extra_args > 0

C:\ProgramData\Anaconda3\lib\site-packages\sklearn\metrics\_ranking.py in roc_a
uc_score(y_true, y_score, average, sample_weight, max_fpr, multi_class, labels)
    534                                 "instead".format(max_fpr))
    535         if multi_class == 'raise':
--> 536             raise ValueError("multi_class must be in ('ovo', 'ovr')")
    537         return _multiclass_roc_auc_score(y_true, y_score, labels,
    538                                          multi_class, average, sample_w
eight)

ValueError: multi_class must be in ('ovo', 'ovr')
```

In [ ]:
```
# i dont understand this sir why i am getting error
# afther this we have to do ROC cruve
# then prediction, accuracy_score
#classification_report from sklearn
```