Does a child's ADHD/ADD status and their use of the internet for learning have an effect on their parent-reported grades?

Trevor Rizzi, William Tran, Yuning Li

Author contributions

1. Trevor Rizzi worked on data cleaning/EDA and created the grade prediction classification model.
2. William Tran worked on data cleaning/EDA and assembled the final report.
3. Yuning Li worked on data cleaning/EDA

Abstract

The National Household Education Surveys (NHES) Program provides publicly accessible datasets that describe detailed profiles of American students. In this report, we analyze and interpret data provided by the NHES to determine if ADHD/ADD status and internet usage in learning affect a student's parent-reported academic performance. We then attempt to accurately predict academic performance based off of these given factors. Ultimately, we found that no significant relation could be drawn between ADHD/ADD status and internet usage in learning, and that both these factors alone are not reliable predictors for student grades.

Introduction

Background

The National Household Education Surveys (NHES) Program was developed by the National Center for Education Statistics (NCES), an agency within the U.S. Department of Education’s Institute of Education Sciences. The surveys that comprise the NHES are integral data collection tools for addressing topics that cannot be studied through institutional data collections. By collecting data directly from households, the NHES has allowed NCES to gather data on a wide range of issues, such as early childhood care and education, children’s readiness for school, the before- and after-school activities of school-age children, adult basic and work-related education, parents’ involvement in education, school choice, and homeschooling.

In this project, we look at these data to conduct research that considers the relation between a child's ADHD/ADD status and their use of the internet for learning to determine if these factors impact academic performance. Of all the columns in this dataset, what we found particularly interesting were those that involved the child's internet usage, the child's disability status, the type of school that they attend, and their parental involvement in their schoolwork.

Aims

We approached our chosen dataset based on three questions:

1. *Does a child's ADHD/ADD status have an effect on their use of technology and the internet for learning?*
2. *Does a student's ethnicity affect the type of schooling (K-12) in which they are enrolled? What does the ethnic makeup of a given school type's minority population look like?*
3. *Does parental involvement in children's school work affect their children's grades and performance in school?*

Some interesting findings emerged from our initial EDA. We found that children with higher grades receive less parental help on homework assignments, perhaps indicating that high-performing students mostly complete homework independently. We also found that between children with ADHD/ADD and those without, relative rates of internet usage for learning remain quite similar.

After our initial EDA, we decided to narrow our focus on the data towards a combination of questions 1 and 3. By looking at differences in grade performance and technology usage between children with ADHD/ADD and those without, we attempted to produce a predictive model that forecasts a child's grades based on ADHD/ADD status and internet usage. Unfortunately, with an approximate 50% prediction accuracy rate, we ultimately could not reliably forecast this information with the selected features.

Materials and methods

Datasets

The data come from The National Household Education Surveys (NHES) Program which ask many questions that inform topics central to education policy and research. They come from two surveys, Early Childhood Program Participation (ECPP) survey and the Parent and Family Involvement in Education (PFI) survey. ECPP is about children aged 6 or younger and not enrolled in kindergarten, and PFI is for those 20 or younger. These surveys were administered primarily online. In 2019, households were mailed an invitation to a web survey or a screener asking for general detail about every child living in the household. After the screener, one child per household was selected and their parent would then complete the actual web survey. An initial sample of 225,500 addresses was selected, of which 205,000 were designated for the NHES:2019. The NHES:2019 sample was a two-phase, stratified sample. In the first phase, a sample of residential addresses was selected from the MSG master address file. In the second phase, an eligible child was selected from information provided in the completed household screener.

Tables describing all our chosen variables of interest as well as details relating to our key features are below.

Variable descriptions: Name | Variable description | Type | ---|---|---| BASMID | Unique Child Identifier | Numeric HVINTSPHO | Internet access on a cell phone | Categorical HVINTCOM | Internet access ona computer | Categorical CHILDNT | How often the child uses the internet for learning | Categorical LRNCOMP | Learn on a computer | Categorical LRNTAB | Learn on a tablet | Categorical LRNCELL | Learn on a cellphone | Categorical HDADDX | Child have ADD or ADHD | Categorical SEGRADES | Child's grades| Categorical SEGRADEQ | Description of schoolwork| Categorical FHCHECKX | Frequency of checking for child's complete homework| Categorical FHHELP | Help with homework| Categorical RACEETH2 | Detailed race & ethnicity of the child| Categorical EDCPUB | If the school is public| Categorical EDCCAT | If the school is private catholic| Categorical EDCREL | If the school is private religious and not catholic| Categorical EDCPRI | Private not religious school| Categorical EDCINTK12 | Full time online k - 12| Categorical EDCHSFL | Homeschooled| Categorical

Dataset properties:

`CHLDNT` : Mode = 'Everyday'. Most children in the dataset internet for learning everyday.

`HDADDX` : Mode = 'No'. Most children in the dataset do not have ADHD or ADD : .804% of children included in the dataset have ADHD or ADD

`HVINTCOM` : 91% of respondents have a computer with internet

`HVINTCELL` : 98% of respondents have a phone with internet

`SEGRADES` : Mode = 'Mostly A's'. Most parents in the dataset claim that their children get mostly A grades

`RACEETH2` : Mode = 'White'. White is the most common race reported

`FHHELP` : Mode = 'Once a week' . Most parents claim to help their child with homework about once a week.

Methods

As the majority of the variables presented in the data are categorical, our EDA consists primarily of visualization done using bar plots and line plots. Some notable visualizations include comparisons between amount of homework help received and average grade cohort attained, frequencies of internet usage in learning separated by ADHD/ADD status, and frequencies of the different average grade cohorts. Our multiclass classification model attempts to predict the reported grade cohort of a given student depending on their ADHD/ADD status and their usage of the internet for learning, unfortunately with a relatively low (50%) prediction accuracy rate. The performance of our model is visualized using an ROC curve which, in direct contradiction with our previous results, shows a relatively strong AUC score of 0.83. This contradiction is explained in the later "Discussion" portion of the report.

Results

This section should show your results. You can include sub-header structure as suits your aims (or not).

The easiest way to compose this section is to structure it around your figures and tables. Prepare and input your figures and tables in the order you'd like them to appear, and then draft the text. Move through the figures and tables in sequence, and for each one:

- introduce the figure/table;
- describe what it shows;
- and then describe what you see (its significance).

Keep the latter brief; you'll have an opportunity to offer more nuanced/extended commentary in the discussion section.

Figure 1 below shows a line plot comparing the frequencies of each grade cohort colored and separated by the average amount of homework help a student received. We can see that students who perform better academically typically receive less homework help.

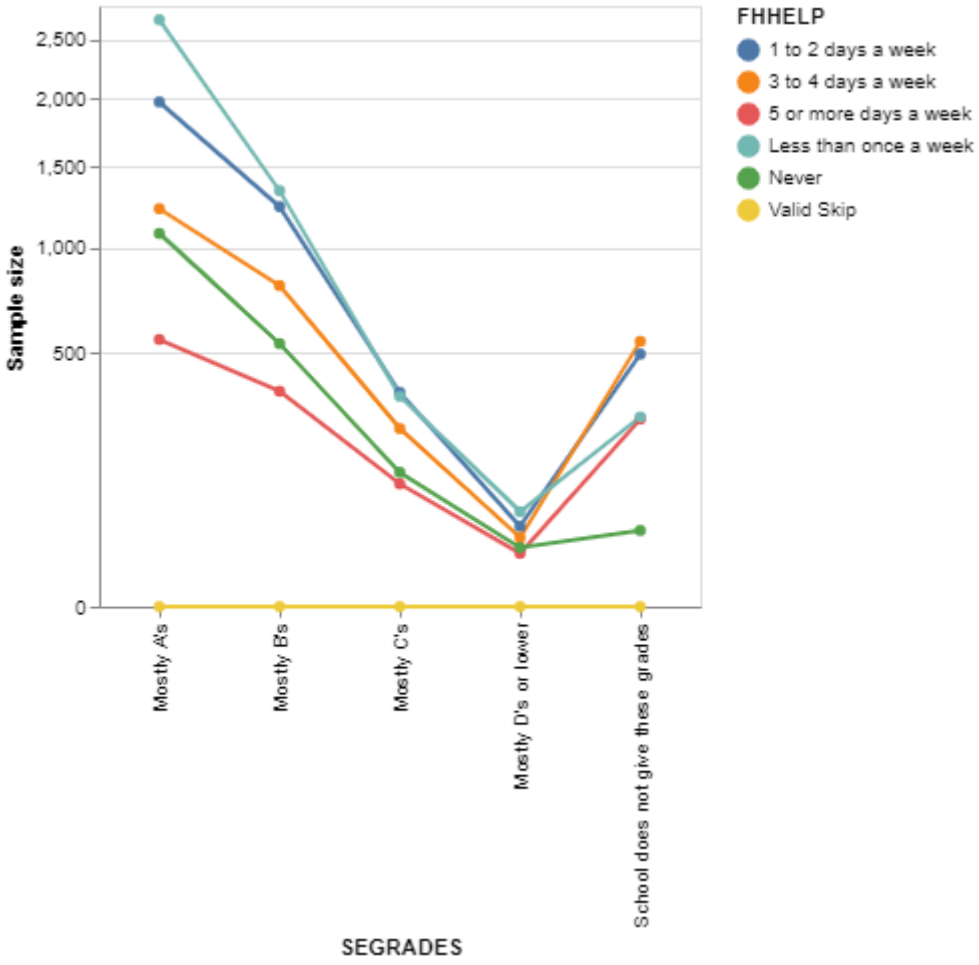


Figure 2 supports the conclusions shown in Figure 1 by depicting the proportions of amounts of homework help received per grade cohort.

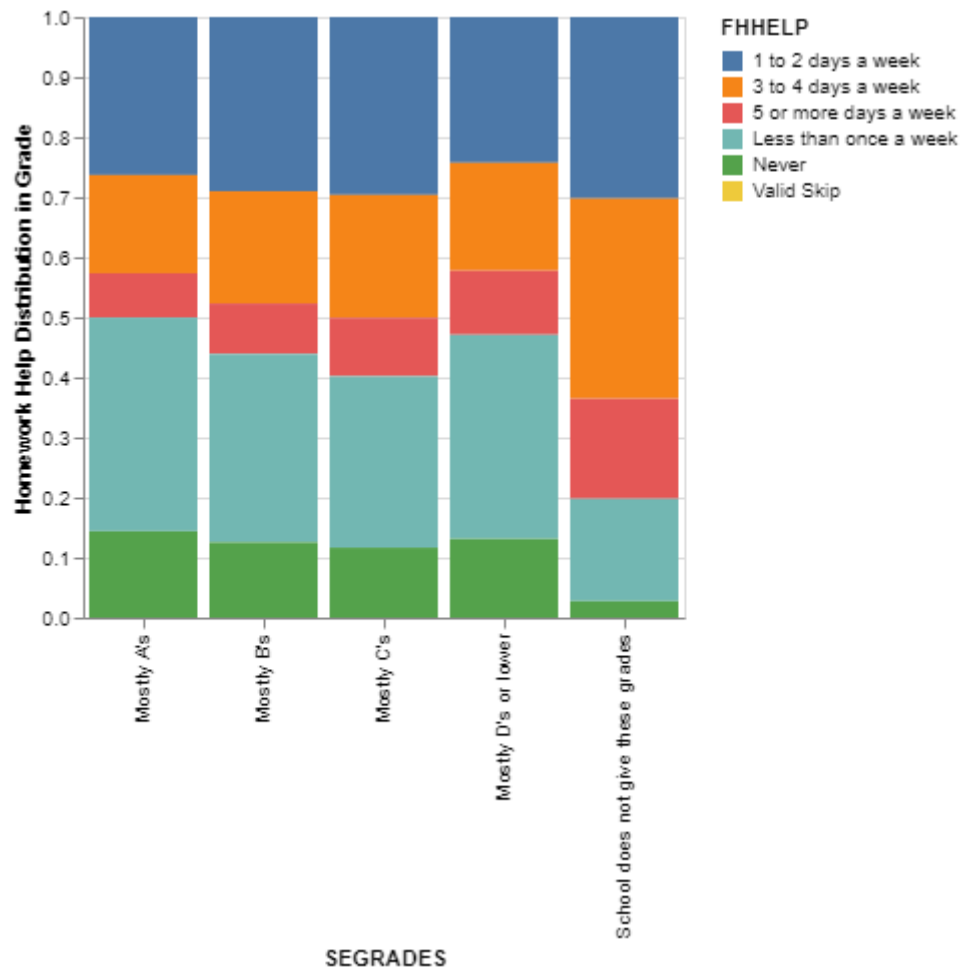


Figure 3 shows the counts of a child's internet usage grouped by their ADHD/ADD status. We can see that the graphs have a very similar distribution but at different scales. To further examine this we can scale the count with log(), shown in Figure 4.

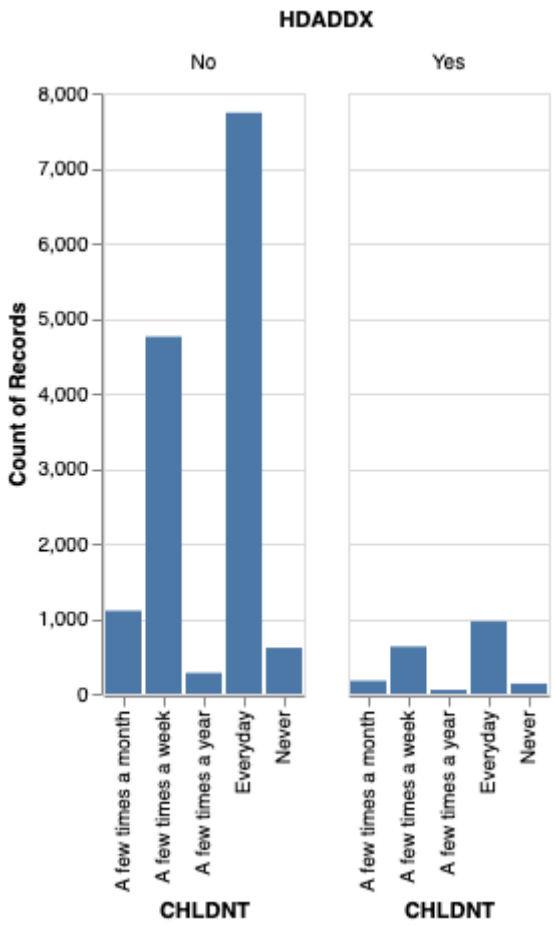
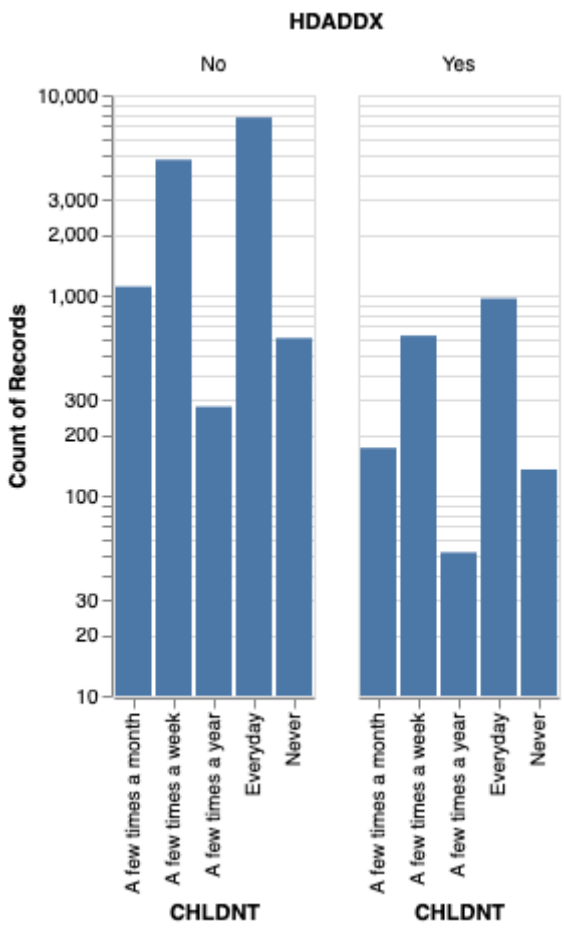
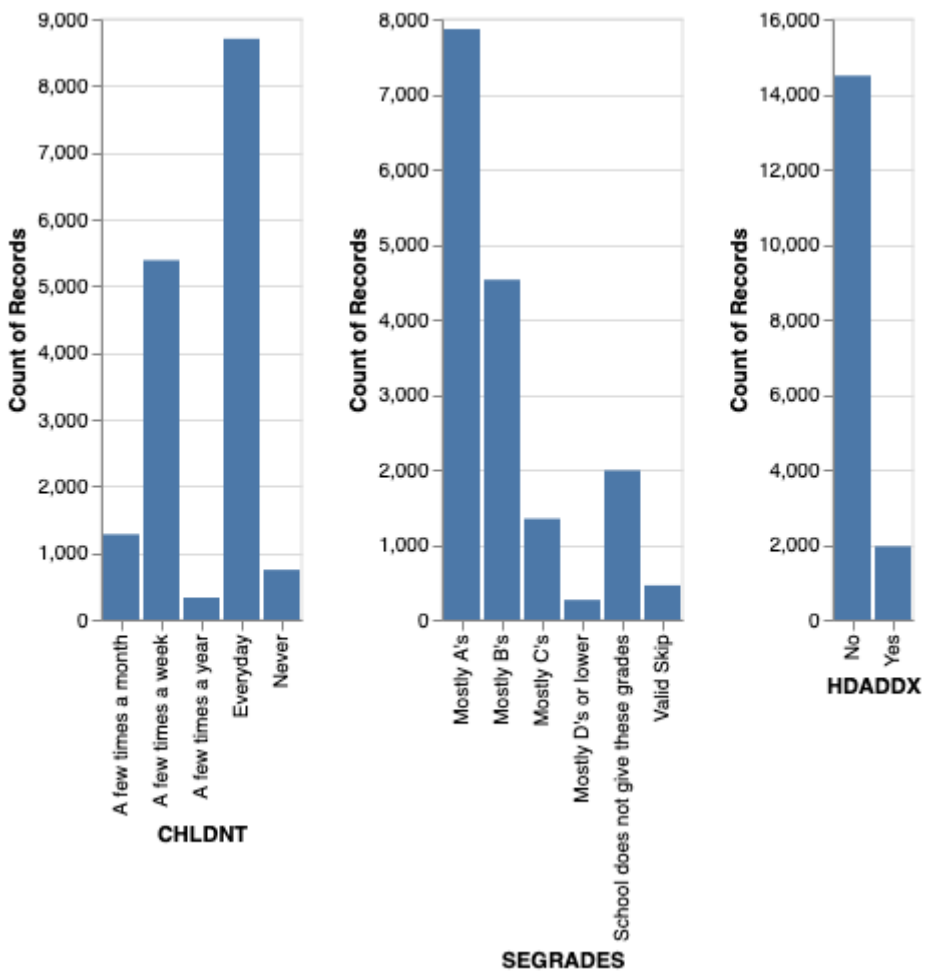


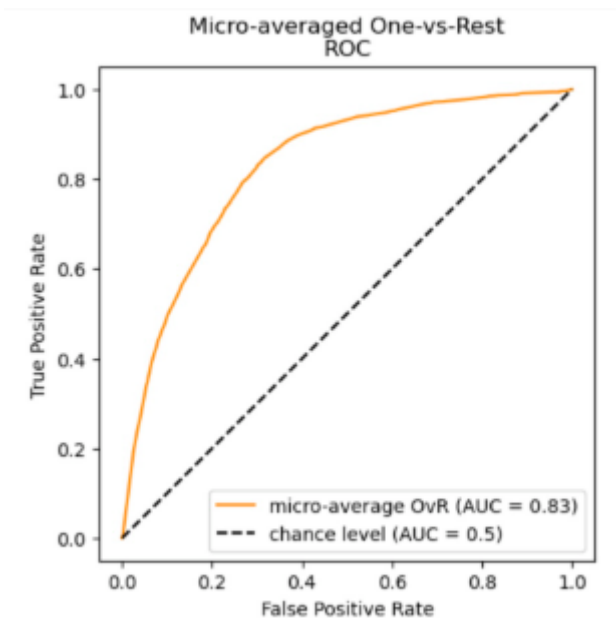
Figure 4 shows the log count of records against the child's internet usage, faceted by ADHD. It is important to note that this chart cannot be read in the same way as the previous. In the linearly scaled bar plot, we compare the heights of each bar. In the log scaled chart, we are looking at the variation of the responses at equal magnitudes. The plot still shows a similar distribution, but the variation of internet usage for non ADHD/ADD is higher than that of those with ADHD/ADD.



Finally, Figure 5 shows us the count distribution of some of our variables of primary interest. We can see that most students use the internet for learning everyday, and that most students get primarily A's.



Since this data is entirely categorical, we must use a classification model to predict the grades. Remember the students' grades are grouped into "Mostly A's", "Mostly B's", etc so we will have to perform a multiclass classification. To do this, we fit two models, a logistic regression and a random forest classifier. Performance between the two models was nearly identical with about a 50% prediction accuracy coming from both. These results are definitely not ideal and suggest poor model performance. To look at the ROC of a multi class classification model, we first choose a one v rest classifier that looks at one of the classes (the positive class) and compares them to the rest (the negative classes). To look at all of the classes as a whole, we have to use an averaging technique. In this case since we have highly imbalanced classes, we use a micro average. This gives us the overall performance of the model, but not of the individual classes. Interestingly, our AUC score was .83 which is considered good.



Discussion

In response to our main guiding question, we could not definitively draw a correlation between ADHD/ADD status or internet usage in learning and average academic performance. Internet usage in learning appears to be largely unaffected by ADHD/ADD status, and both factors alone are not able to accurately predict a given student's grade cohort.

It is important to address some possible sources of problems in our analysis. In regards to Figure 5, we should note that the grade evaluations are reported by the students' parents which could very well indicate response bias. Unsurprisingly, most of the subjects in the data do not have ADHD/ADD which could lead to potentially misfitting of the data in the future. One interesting result of our modeling is the contradiction between our poor prediction accuracy rate of about 50% and our relatively high AUC score of 0.83. A possible reason for this is the imbalance in the data. The majority of instances are included in the "Mostly A's" class leading the model to predict that class more often.

Some further topics we would have liked to explore in this dataset are questions involving student ethnicity. Our early EDA involved some brief analysis of columns relating to student ethnicity, but was ultimately dropped in order to narrow the scope of our project to generate more specific and meaningful conclusions. The data provided have detailed insights as to student ethnicity; perhaps we could compare this to features regarding socioeconomic status to shed light on some disparities between students in the US education system.

APPENDIX

Question: Does a child's ADHD/ADD status have an effect on their use of technology and the internet for learning?

In [1]:
```
# packages
import numpy as np
import pandas as pd
import altair as alt
import seaborn as sns
import sklearn.linear_model as lm
from sklearn.linear_model import LinearRegression
import warnings
from sklearn.preprocessing import add_dummy_feature

# raw data
ecpp = pd.read_csv('data/nhes_19_ecpp_v1_0.csv') #earl childhood program participation
pfi = pd.read_csv('data/nhes_19_pfi_v1_0.csv') # Parent and Family Involvement in Education

#data = pd.merge(pfi, ecpp, how = 'left' , on = 'BASMID')
data = pd.concat([ecpp,pfi], ignore_index=True)
```

In [3]:
```
data.head(3)
```

Out[3]:

	BASMID	RCNOW	RCWEEK	RCTYPE	RCAGE	RCPLACE	RCTIME	RCDAYS	RCHRS	RCCVRWK	...	F_FOGROUPX	F_FOSPRTEVX	F_
0	20191000097	2.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	...	NaN	NaN	
1	20191000098	2.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	...	NaN	NaN	
2	20191000116	2.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0	...	NaN	NaN	

3 rows × 1169 columns

In [2]:
```
data = pfi[['BASMID', 'HVINTSPHO', 'HVINTCOM', 'CHLDNT', 'LRNCOMP', 'LRNTAB', 'LRNCELL', 'HDADDX', 'SEGRADES',
            'SEREPEAT', 'SEGRADEQ', 'FHCHECKX', 'FHHELP', 'RACEETH2', 'EDCPUB', 'EDCCAT', 'EDCREL', 'EDCPRI', 'EDCI
```

In [86]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16446 entries, 0 to 16445
Data columns (total 20 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   BASMID     16446 non-null  int64
 1   HVINTSPHO  16446 non-null  int64
 2   HVINTCOM   16446 non-null  int64
 3   CHLDNT     16446 non-null  int64
 4   LRNCOMP    16446 non-null  int64
 5   LRNTAB     16446 non-null  int64
 6   LRNCELL    16446 non-null  int64
 7   HDADDX     16446 non-null  int64
 8   SEGRADES   16446 non-null  int64
 9   SEREPEAT   16446 non-null  int64
 10  SEGRADEQ   16446 non-null  int64
 11  FHCHECKX   16446 non-null  int64
 12  FHHELP     16446 non-null  int64
 13  RACEETH2   16446 non-null  int64
 14  EDCPUB     16446 non-null  int64
 15  EDCCAT     16446 non-null  int64
 16  EDCREL     16446 non-null  int64
 17  EDCPRI     16446 non-null  int64
 18  EDCINTK12  16446 non-null  int64
 19  EDCHSFL    16446 non-null  int64
dtypes: int64(20)
memory usage: 2.5 MB
```

In [87]:
```
data.isna().sum() #No NA values !!

data['CHLDNT'].mode() # 5 is most common
data['HDADDX'].mode()

f = data[data['HDADDX'] == 1].count() / data.count()
f


data[data['HVINTSPHO'] == 1].count() # 2 = 644, 1 = 6448

#7035 + 57 = 7092
```

Out[87]:
```
BASMID       16049
HVINTSPHO    16049
HVINTCOM     16049
CHLDNT       16049
LRNCOMP      16049
LRNTAB       16049
LRNCELL      16049
HDADDX       16049
SEGRADES     16049
SEREPEAT     16049
SEGRADEQ     16049
FHCHECKX     16049
FHHELP       16049
RACEETH2     16049
EDCPUB       16049
EDCCAT       16049
EDCREL       16049
EDCPRI       16049
EDCINTK12    16049
EDCHSFL      16049
dtype: int64
```

In [88]:
```
di = {1: 'Yes', 2: 'No' }
di2 = {1: 'Yes', 2: 'No', -1: 'Valid Skip' }
chldnt = {1 : 'Everyday', 2 : 'A few times a week', 3 : 'A few times a month', 4: 'A few times a year', 5 : 'Never'

grade = {1: "Mostly A's", 2: "Mostly B's", 3: "Mostly C's", 4: "Mostly D's or lower",
        5: "School does not give these grades", -1: "Valid Skip"}

description = {1: 'Excellent', 2: 'Above average', 3: 'Average', 4: 'Below Average',
        5: 'Failing', -1: 'Valid Skip'}

hhelp = {1: 'Less than once a week', 2: '1 to 2 days a week', 3: '3 to 4 days a week',
        4: '5 or more days a week', 5: 'Never', -1: 'Valid Skip'}

check = {1: 'Never', 2: 'Rarely', 3: 'Sometimes', 4: 'Always', -1: 'Valid Skip'}

race = {1:'White',
        2:'Black',
        3:'Mexican, Mexican American, or Chicano',
        4:'Puerto Rican',
        5:'Cuban',
        6:'Another Hispanic, Latino, or Spanish origin or more than one Hispanic, Latino, or Spanish origin',
        7:'Asian',
        8:'Native Hawaiian or other Pacific Islander',
        9:'American Indian or Alaska Natives',
        10:'All other races and multiple races'}


cat_data = data.replace({'HVINTSPHO': di, 'HVINTCOM':di, 'CHLDNT':chldnt, 'LRNCOMP':di2, 'LRNTAB':di2, 'LRNCELL': d
                         'HDADDX' : di,'SEGRADES': grade, 'SEREPEAT': di2, 'SEGRADEQ': description,
                         'FHCHECKX': check, 'FHHELP': hhelp,'RACEETH2':race, 'EDCPUB':di, 'EDCCAT':di,
                         'EDCREL':di, 'EDCPRI':di, 'EDCINTK12':di, 'EDCINTCOL':di, 'EDCCOL':di, 'EDCHSFL':di})
```

In [89]:
```
cat_data.to_csv('cat_data.csv', index = False)
```

In [3]:
```
cat_data = pd.read_csv("cat_data.csv")
```

In [7]:
```
cat_data.info()
cat_data.head()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16446 entries, 0 to 16445
Data columns (total 20 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   BASMID     16446 non-null  int64 
 1   HVINTSPHO  16446 non-null  object
 2   HVINTCOM   16446 non-null  object
 3   CHLDNT     16446 non-null  object
 4   LRNCOMP    16446 non-null  object
 5   LRNTAB     16446 non-null  object
 6   LRNCELL    16446 non-null  object
 7   HDADDX     16446 non-null  object
 8   SEGRADES   16446 non-null  object
 9   SEREPEAT   16446 non-null  object
 10  SEGRADEQ   16446 non-null  object
 11  FHCHECKX   16446 non-null  object
 12  FHHELP     16446 non-null  object
 13  RACEETH2   16446 non-null  object
 14  EDCPUB     16446 non-null  object
 15  EDCCAT     16446 non-null  object
 16  EDCREL     16446 non-null  object
 17  EDCPRI     16446 non-null  object
 18  EDCINTK12  16446 non-null  object
 19  EDCHSFL    16446 non-null  object
dtypes: int64(1), object(19)
memory usage: 2.5+ MB
```

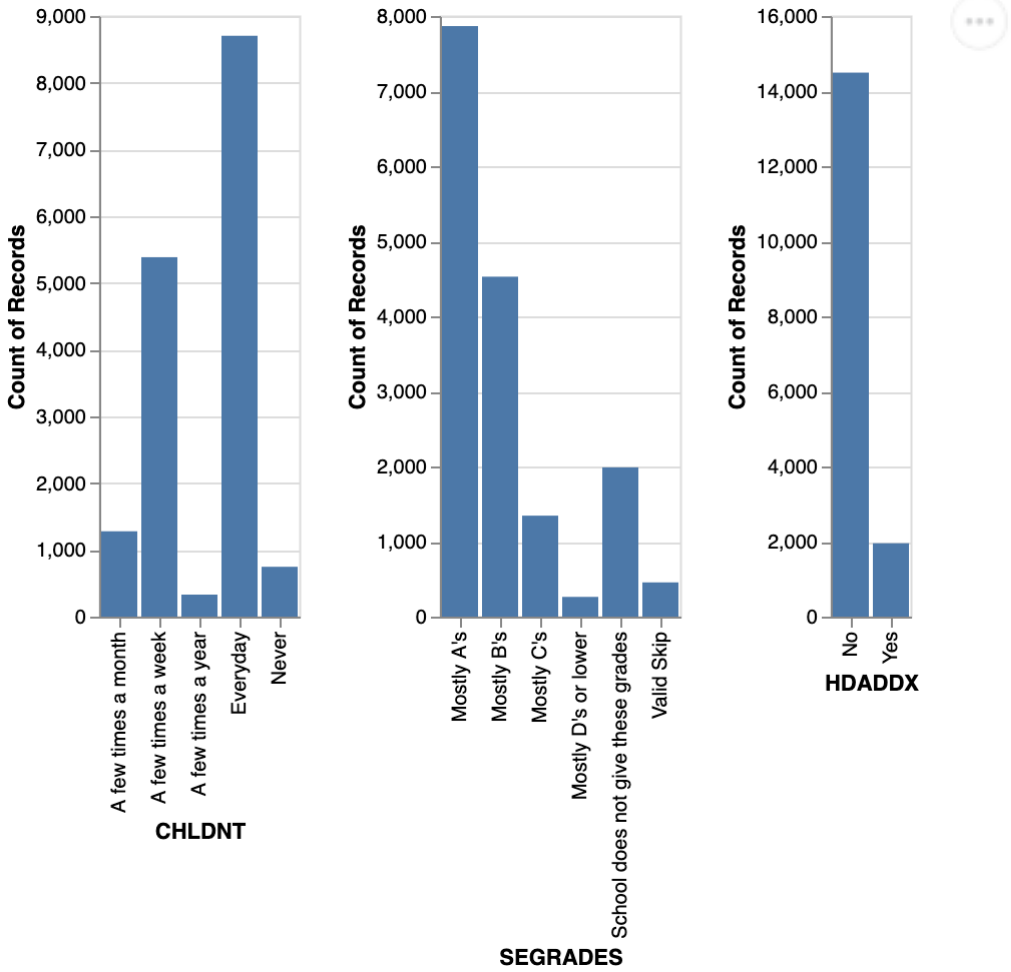
Out[7]:

	BASMID	HVINTSPHO	HVINTCOM	CHLDNT	LRNCOMP	LRNTAB	LRNCELL	HDADDX	SEGRADES	SEREPEAT	SEGRADEQ	FHCHECK
0	20191000012	Yes	Yes	Everyday	Yes	No	Yes	No	Mostly A's	No	Excellent	Sometime
1	20191000029	Yes	Yes	Everyday	Yes	Yes	Yes	No	School does not give these grades	No	Average	Alway
2	20191000059	Yes	No	A few times a year	Yes	No	Yes	No	Mostly C's	No	Below Average	Sometime
3	20191000070	Yes	Yes	A few times a week	Yes	Yes	Yes	No	Mostly A's	No	Excellent	Alway
4	20191000078	Yes	Yes	A few times a week	Yes	No	No	No	Mostly A's	No	Excellent	Alway

In [148…

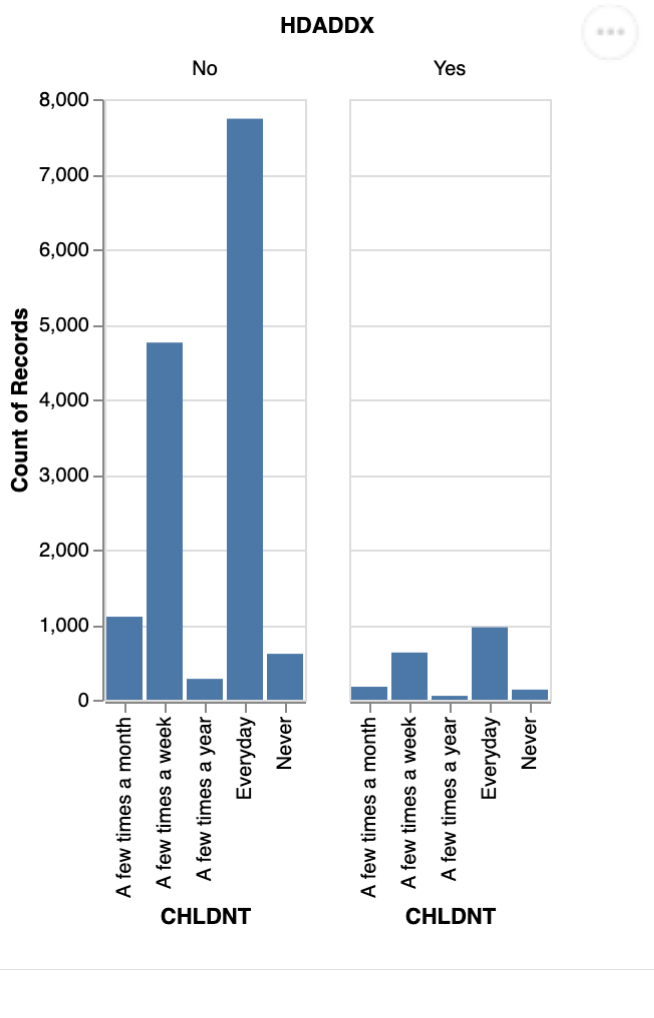
```
alt.data_transformers.disable_max_rows()
x = alt.Chart(cat_data).mark_bar().encode(
    x='CHLDNT',
    y='count()'
)


y = alt.Chart(cat_data).mark_bar().encode(
    x='SEGRADES',
    y='count()'
)

z = alt.Chart(cat_data).mark_bar().encode(
    x='HDADDX',
    y='count()'
)

z = alt.Chart(cat_data).mark_bar().encode(
    x='HDADDX',
    y='count()'
)


x | y | z
```

Out[148]:



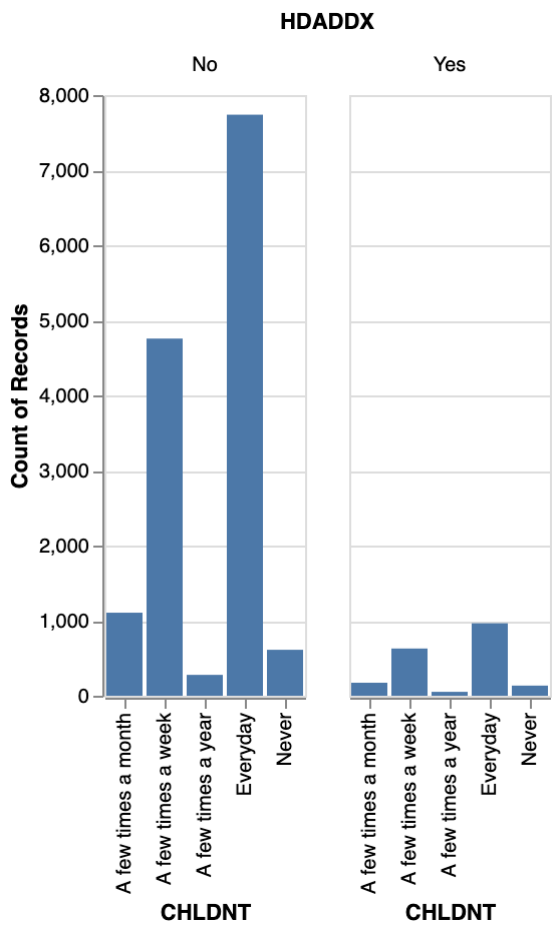
In [10]:
```
alt.Chart(cat_data).mark_bar().encode(
    x='CHLDNT',
    y='count()',
    facet='HDADDX:N',
)
```

Out[10]:



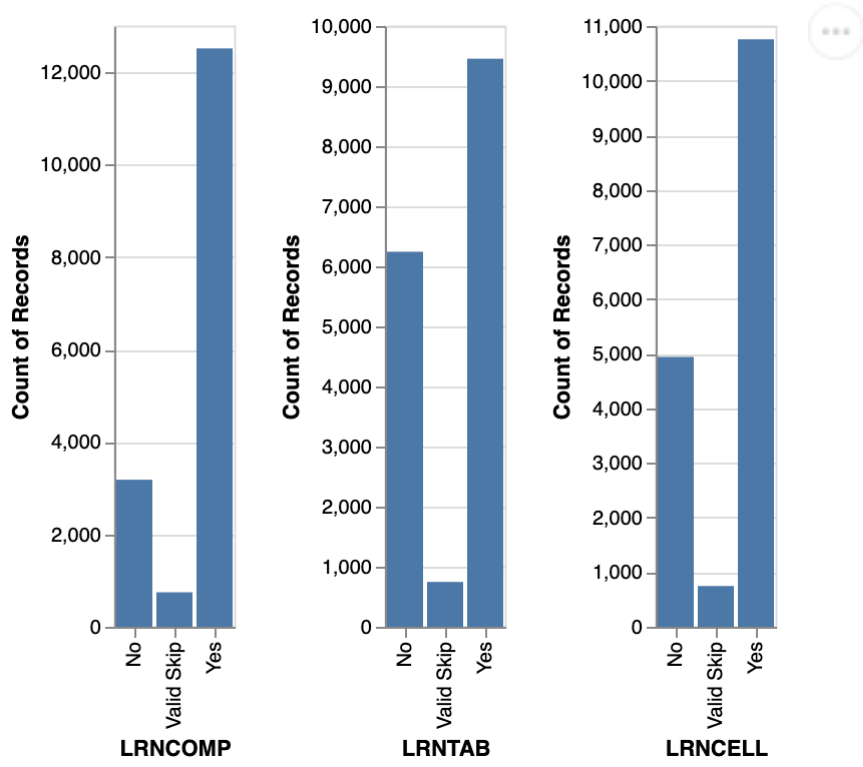
In [11]:

Out[11]:



In [32]:
```
a = alt.Chart(cat_data).mark_bar().encode(
    x=alt.X('LRNCOMP'),
    y='count()',

)
b = alt.Chart(cat_data).mark_bar().encode(
    x=alt.X('LRNTAB'),
    y='count()',

)
c = alt.Chart(cat_data).mark_bar().encode(
    x=alt.X('LRNCELL'),
    y='count()',
)


a | b | c # Seems that most children have access to all three but computer is most common
```

Out[32]:



In [5]:
```
#melted = cat_data.melt(id_vars=['BASMID', 'HVINTSPHO', 'HVINTCOM', 'CHLDNT', 'HDADDX', 'SEGRADES',
    #        'SEREPEAT', 'SEGRADEQ', 'FHCHECKX', 'FHHELP', 'RACEETH2', 'EDCPUB', 'EDCCAT', 'EDCREL', 'EDCPRI', 'EDC
#melted


chart = alt.Chart(cat_data, width=200, height=200).mark_circle().encode(
    x='CHLDNT',
    y='HDADDX',
)
chart
#melted
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[5], line 6
      1 #melted = cat_data.melt(id_vars=['BASMID', 'HVINTSPHO', 'HVINTCOM', 'CHLDNT', 'HDADDX', 'SEGRADES',
      2     #        'SEREPEAT', 'SEGRADEQ', 'FHCHECKX', 'FHHELP', 'RACEETH2', 'EDCPUB', 'EDCCAT', 'EDCREL', 'EDCP
RI', 'EDCINTK12', 'EDCHSFL'], var_name='LRN_type', value_name='LRN_response')
      3 #melted
----> 6 chart = alt.Chart(cat_data, width=200, height=200).mark_circle().encode(
      7     x='CHLDNT',
      8     y='HDADDX',
      9 )
     10 chart
     11 #melted

NameError: name 'cat_data' is not defined
```

In [ ]:

In [66]:

```
/tmp/ipykernel_149/1134858392.py:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecat
ed. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  cat_data.corr()
```

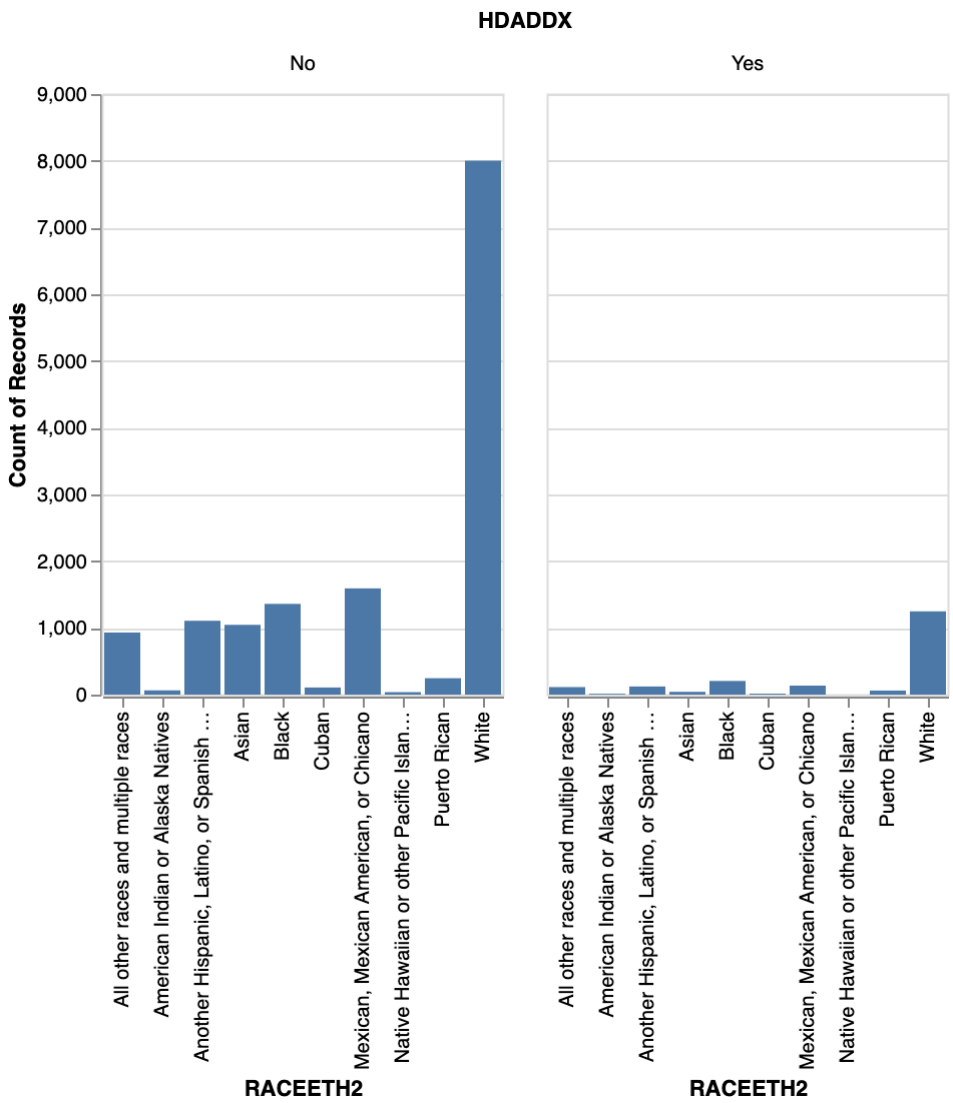
Out[66]:

	BASMID
BASMID	1.0

In [34]:

```
alt.Chart(cat_data).mark_bar().encode(
    x='RACEETH2',
    y='count()',
    facet='HDADDX:N', #HDADDX by RACE
)
```

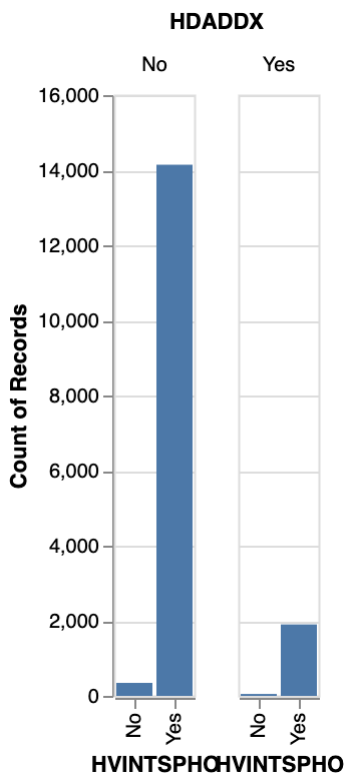
Out[34]:



In [35]:

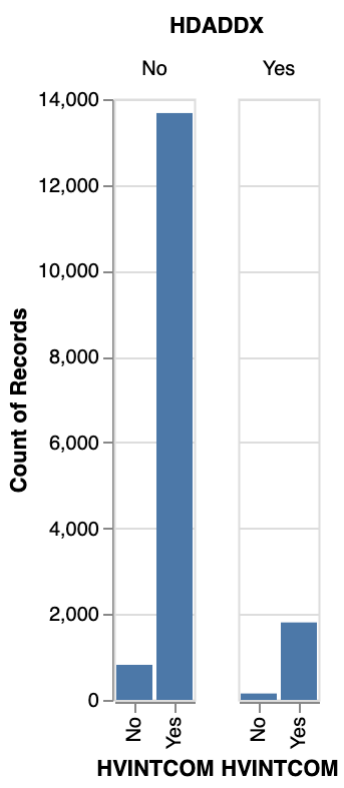
```
alt.data_transformers.disable_max_rows()
alt.Chart(cat_data).mark_bar().encode(
    x='HVINTSPHO',
    y='count()',
    facet='HDADDX'
)
```

Out[35]:



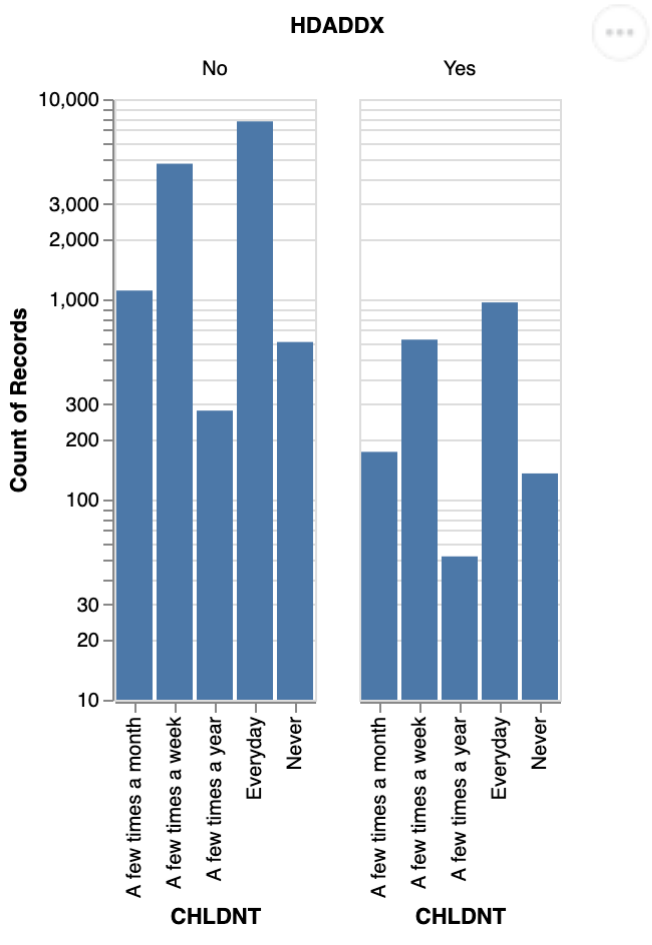
In [15]:
```
alt.data_transformers.disable_max_rows()
alt.Chart(cat_data).mark_bar().encode(
    x='HVINTCOM',
    y='count()',
    facet = 'HDADDX'
)
```

Out[15]:



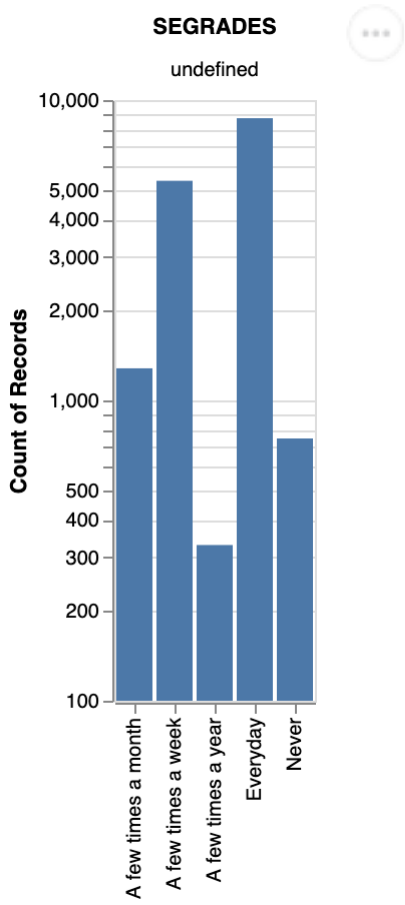
In [16]:
```
alt.Chart(cat_data).mark_bar().encode(
    x='CHLDNT',
    y=alt.Y('count()', scale=alt.Scale(type="log")),
    facet='HDADDX:N',
) # Here we have on LOG scale to more clearly visualize the distributions  - similar distributions but the HDADDX h
```

Out[16]:



In [45]:
```
alt.Chart(cat_data).mark_bar().encode(
    x='CHLDNT',
    y=alt.Y('count()', scale=alt.Scale(type="log")),
    facet='SEGRADES:N',
)
```

Out[45]:



In [46]:

Out[46]:

	BASMID	HVINTSPHO	HVINTCOM	CHLDNT	LRNCOMP	LRNTAB	LRNCELL	HDADDX
0	20191000012	Yes	Yes	Everyday	Yes	No	Yes	No
1	20191000029	Yes	Yes	Everyday	Yes	Yes	Yes	No
2	20191000059	Yes	No	A few times a year	Yes	No	Yes	No
3	20191000070	Yes	Yes	A few times a week	Yes	Yes	Yes	No
4	20191000078	Yes	Yes	A few times a week	Yes	No	No	No
...	...	...	...	...	...	...	...	...
16441	20191225472	Yes	Yes	A few times a week	Yes	Yes	Yes	No
16442	20191225475	Yes	Yes	Everyday	No	Yes	No	No
16443	20191225477	Yes	Yes	Everyday	Yes	No	Yes	No
16444	20191225479	Yes	No	A few times a week	No	No	Yes	No
16445	20191225500	Yes	Yes	Everyday	Yes	Yes	Yes	No

16446 rows × 8 columns

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [73]:

Out[73]:

	SEGRADES_Mostly A's	SEGRADES_Mostly B's	SEGRADES_Mostly C's	SEGRADES_Mostly D's or lower	SEGRADES_School does not give these grades	SEGRADES_Valid Skip
0	1	0	0	0	0	0
1	0	0	0	0	1	0
2	0	0	1	0	0	0
3	1	0	0	0	0	0
4	1	0	0	0	0	0
...	...	...	...	...	...	...
16441	1	0	0	0	0	0
16442	0	0	0	0	1	0
16443	1	0	0	0	0	0
16444	1	0	0	0	0	0
16445	0	1	0	0	0	0

16446 rows × 6 columns

In [78]:

Out[78]:

	BASMID	HVINTSPHO	HVINTCOM	CHLDNT	LRNCOMP	LRNTAB	LRNCELL	HDADDX
0	20191000012	Yes	Yes	Everyday	Yes	No	Yes	No
1	20191000029	Yes	Yes	Everyday	Yes	Yes	Yes	No
2	20191000059	Yes	No	A few times a year	Yes	No	Yes	No
3	20191000070	Yes	Yes	A few times a week	Yes	Yes	Yes	No
4	20191000078	Yes	Yes	A few times a week	Yes	No	No	No
...	...	...	...	...	...	...	...	...
16441	20191225472	Yes	Yes	A few times a week	Yes	Yes	Yes	No
16442	20191225475	Yes	Yes	Everyday	No	Yes	No	No
16443	20191225477	Yes	Yes	Everyday	Yes	No	Yes	No
16444	20191225479	Yes	No	A few times a week	No	No	Yes	No
16445	20191225500	Yes	Yes	Everyday	Yes	Yes	Yes	No

16446 rows × 8 columns

In [4]:
```
HDADDX_dummies = pd.get_dummies(cat_data.HDADDX, drop_first = False, prefix = 'HDADDX')
HVINTSPHO_dummies = pd.get_dummies(cat_data.HVINTSPHO, drop_first = False, prefix = 'HVINTSPHO')
HVINTCOM_dummies = pd.get_dummies(cat_data.HVINTCOM, drop_first = False, prefix = 'HVINTCOM')
CHLDNT_dummies = pd.get_dummies(cat_data.CHLDNT, drop_first = False, prefix = 'CHLDNT')
LRNCOMP_dummies = pd.get_dummies(cat_data.LRNCOMP, drop_first = False, prefix = 'LRNCOMP')
LRNTAB_dummies = pd.get_dummies(cat_data.LRNTAB, drop_first = False, prefix = 'LRNTAB')
LRNCELL_dummies = pd.get_dummies(cat_data.LRNCELL, drop_first = False, prefix = 'LRNCELL')
SEGRADES_dummies = pd.get_dummies(cat_data.SEGRADES, drop_first = False, prefix = 'SEGRADES')



cat_data_drop = cat_data[['BASMID', 'HVINTSPHO', 'HVINTCOM', 'CHLDNT', 'LRNCOMP', 'LRNTAB', 'LRNCELL', 'HDADDX']]
encoded_cat_data = pd.concat([HDADDX_dummies, HVINTSPHO_dummies, HVINTCOM_dummies, CHLDNT_dummies, LRNCOMP_dummies,
encoded_cat_data
```

Out[4]:

	HDADDX_No	HDADDX_Yes	HVINTSPHO_No	HVINTSPHO_Yes	HVINTCOM_No	HVINTCOM_Yes	CHLDNT_A few times a month	CHLDNT_A few times a week	CHLDNT_A few times a year
0	1	0	0	1	0	1	0	0	0
1	1	0	0	1	0	1	0	0	0
2	1	0	0	1	1	0	0	0	1
3	1	0	0	1	0	1	0	1	0
4	1	0	0	1	0	1	0	1	0
...	...	...	...	...	...	...	...	...	...
16441	1	0	0	1	0	1	0	1	0
16442	1	0	0	1	0	1	0	0	0
16443	1	0	0	1	0	1	0	0	0
16444	1	0	0	1	1	0	0	1	0
16445	1	0	0	1	0	1	0	0	0

16446 rows × 17 columns

In [5]:
```
x_mx = add_dummy_feature(encoded_cat_data, value = 1)
y = cat_data.iloc[:,8]

from sklearn.model_selection import train_test_split
x_mx_train, x_mx_test, y_train, y_test = train_test_split(x_mx, y, test_size=0.2, random_state=101)
```

In [73]:

Out[73]:
```
1423                          Mostly A's
1261                          Mostly A's
3028                          Mostly B's
11743                         Mostly A's
2468     School does not give these grades
                        ...
381      School does not give these grades
5632                          Mostly B's
2569                          Mostly A's
12827                         Mostly A's
14182                         Mostly B's
Name: SEGRADES, Length: 3290, dtype: object
```

In [13]:
```
import pandas as pd
import sklearn as sk
from sklearn.linear_model import LogisticRegression
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
from sklearn.neural_network import MLPClassifier

# configure module
mlr = LogisticRegression(solver='lbfgs', multi_class='multinomial', max_iter = 1000)


# fit model
mlr.fit(x_mx_train, y_train)

print(mlr.intercept_)
```

```
[ 0.72915078  1.06027206  0.28026502 -0.90450023  0.03018803 -1.19537566]
```

In [7]:
```
mlr.predict(x_mx_test)
round(mlr.score(x_mx_test,y_test), 6)

#49.36 predictive accuracy which is better than random guessing, but not good enough to suggest a relation
```

Out[7]:
```
0.493617
```

In [8]:
```
RF = RandomForestClassifier(n_estimators=1000, max_depth=10, random_state=0).fit(x_mx_train, y_train)
RF.predict(x_mx_test)
round(RF.score(x_mx_test, y_test), 6)
y_score = RF.fit(x_mx_train, y_train).predict_proba(x_mx_test)
```

In [ ]:

In [14]:
```
summary = RF.summary
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[14], line 1
----> 1 summary = RF.summary

AttributeError: 'RandomForestClassifier' object has no attribute 'summary'
```

In [ ]:

In [ ]:

In [ ]:

In [12]:
```
from sklearn.metrics import RocCurveDisplay

label_binarizer = LabelBinarizer().fit(y_train)
y_onehot_test = label_binarizer.transform(y_test)
y_onehot_test.shape  # (n_samples, n_classes)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[12], line 3
      1 from sklearn.metrics import RocCurveDisplay
----> 3 label_binarizer = LabelBinarizer().fit(y_train)
      4 y_onehot_test = label_binarizer.transform(y_test)
      5 y_onehot_test.shape  # (n_samples, n_classes)

NameError: name 'LabelBinarizer' is not defined
```

In [ ]:
```
from sklearn.metrics import RocCurveDisplay
RocCurveDisplay.from_predictions(
    y_onehot_test.ravel(),
    y_score.ravel(),
    name="micro-average OvR",
    color="darkorange",
)
plt.plot([0, 1], [0, 1], "k--", label="chance level (AUC = 0.5)")
plt.axis("square")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("Micro-averaged One-vs-Rest\nROC")
plt.legend()
plt.show()
```

USe a micro average because we have highly imbalanced classes

The micro averaged ROC curve is a way to summarize the performance of a multiclass classification model by considering all the individual predictions and outcomes together. It is calculated by treating all the predictions and outcomes as a single binary classification problem, and then calculating the true positive rate (TPR) and false positive rate (FPR) across all classes.

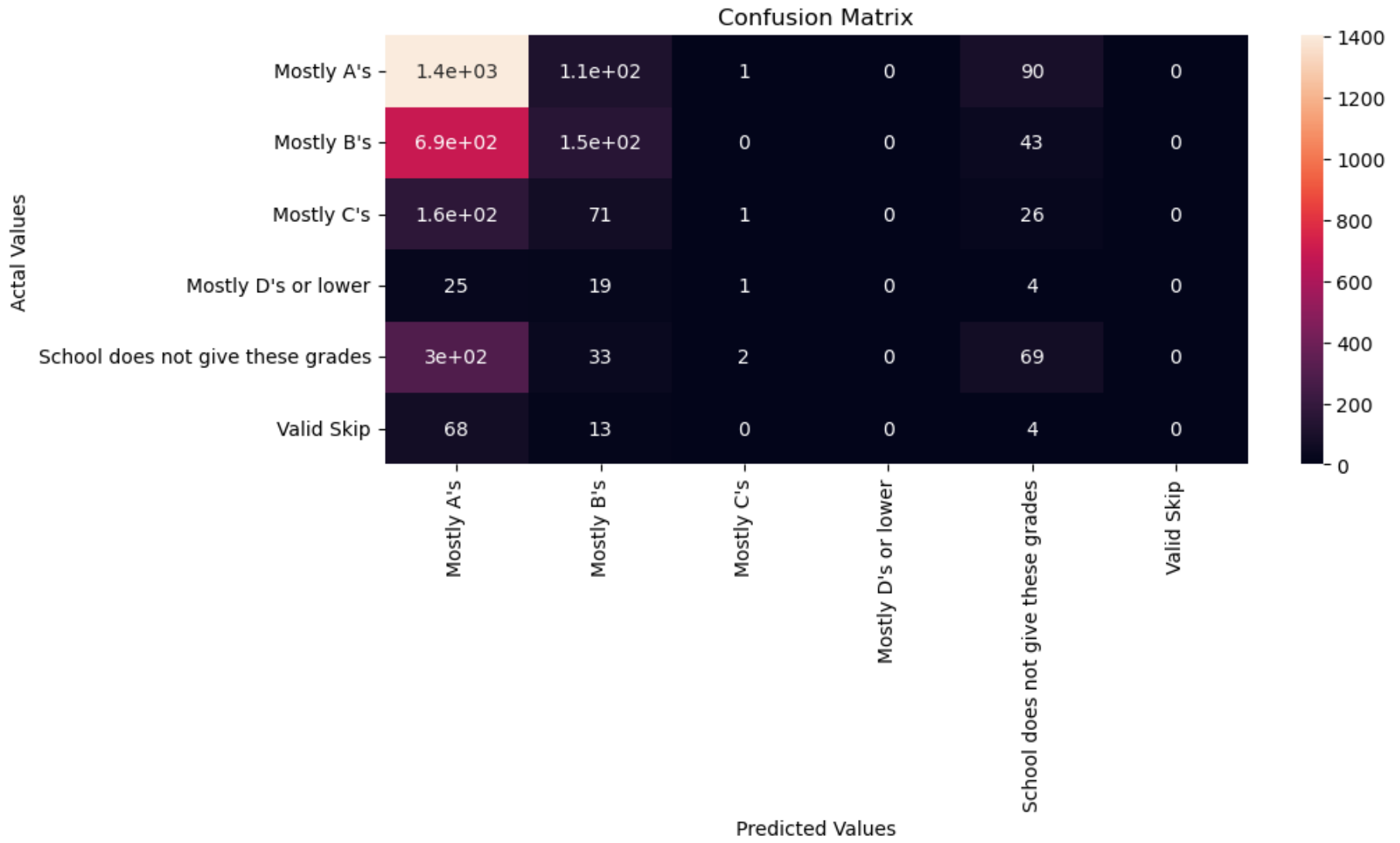
To calculate the micro averaged ROC curve, you can follow these steps:

> For each class, calculate the true positive rate (TPR) and false positive rate (FPR) using the predicted probabilities and the true labels. This will give you a separate ROC curve for each class.
>
> Combine all the true positive and false positive rates across all classes. To do this, you can sum up the true positives and false positives across all classes and calculate a single TPR and FPR for the entire dataset.
>
> Plot the combined TPR and FPR on a graph to get the micro averaged ROC curve.

Note that the micro averaged ROC curve is a way to evaluate the overall performance of a multiclass classification model, but it does not provide insights into the performance of individual classes. If you want to evaluate the performance of individual classes, you can use the macro averaged ROC curve or the ROC curve for each individual class.

In [10]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [11]:
```
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score, recall_score, ConfusionMatrixDisplay
# Generate predictions with the best model
y_pred = mlr.predict(x_mx_test)

# Create the confusion matrix
cm = confusion_matrix(y_test, y_pred)
# Creating a dataframe for a array-formatted Confusion matrix,so it will be easy for plotting.
cm_df = pd.DataFrame(cm,
                     index = ["Mostly A's", "Mostly B's", "Mostly C's", "Mostly D's or lower", "School does not giv
                     columns =["Mostly A's", "Mostly B's", "Mostly C's", "Mostly D's or lower", "School does not gi
#Plotting the confusion matrix
plt.figure(figsize=(10,4))
sns.heatmap(cm_df, annot=True)
plt.title('Confusion Matrix')
plt.ylabel('Actal Values')
plt.xlabel('Predicted Values')
plt.show()
```



In [ ]:

In [74]:

In [118…

In [ ]:

In [ ]:

In [ ]: