In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import numpy as np
import seaborn as sns
import re
```

In [2]:
```
# Opens all the csv files for all the schools
```

In [3]:
```
ucsc_files = open('./184_project/UCSC files.csv')
df_ucsc = pd.read_csv(ucsc_files)

uci_files = open('./184_project/UCI/UCI files.csv')
df_uci = pd.read_csv(uci_files)

sdsu_files = open('./184_project/SDSU/SDSU files.csv')
df_sdsu = pd.read_csv(sdsu_files)

chico_files = open('./184_project/CHICO/CHICO files.csv')
df_chico = pd.read_csv(chico_files)

sfsu_files = open('./184_project/SF State/SF_State_Data.csv')
df_sfsu = pd.read_csv(sfsu_files)

harveymud_files = open('./184_project/Harvey_files.csv')
df_harvey = pd.read_csv(harveymud_files)

panoma_files = open('./184_project/Pomona_files.csv')
df_panoma = pd.read_csv(panoma_files)

rice_files = open('./184_project/RICE/RICE files.csv')
df_rice = pd.read_csv(rice_files)
```

In [4]:
```
#Goes through all the schools and cleans up the term to a specific format (season + ' ' + year).
```

In [5]:
```
real_cols = ['num_people_enrolled', 'total_class_size', 'class_num', 'term',
'professor']
df_rice.columns = real_cols
df_rice
for row_num,i in enumerate(df_rice.iloc[:,3]):
    term_split = i.split('2')
    if term_split[0] == 'FALL':
        term_split[0] = 'Fall'
        term_split[1] = '2' + term_split[1]
    elif term_split[0] == 'SPRING':
        term_split[0] = 'Spring'
        term_split[1] = '2' + term_split[1]
    if len(term_split) == 3:
        term_split = [term_split[0], term_split[1] + '2']
    season_year = term_split[0] + ' ' + term_split[1]
    df_rice.at[row_num, 'term'] = season_year.upper()
#df_rice
```

In [6]:
```
for row_num, i in enumerate(df_harvey.iloc[:,3]):
    term_split = i.split(' ')
    if len(term_split[1]) < 4:
        year_last = term_split[1].split('0')
        term_split[1] = '200' + year_last[1]
    df_harvey.at[row_num, 'term'] = term_split[0] +' ' + term_split[1]
```

In [7]:
```
for row_num, i in enumerate(df_panoma.iloc[:,3]):
    term_split = i.split(' ')
    if len(term_split[1]) < 4:
        year_last = term_split[1].split('0')
        term_split[1] = '200' + year_last[1]
    df_panoma.at[row_num, 'term'] = term_split[0] + ' ' + term_split[1]
```

In [8]:
```
for row_num, i in enumerate(df_chico.iloc[:,3]):
    term_split = i.split('2')
    if term_split[0] == 'fa':
        term_split[0] = 'Fall'
        term_split[1] = '2' + term_split[1]
    elif term_split[0] == 'spr':
        term_split[0] = 'Spring'
        term_split[1] = '2' + term_split[1]
    if len(term_split) == 3:
        term_split = [term_split[0], term_split[1] + '2']
    season_year = term_split[0] + ' ' + term_split[1]
    df_chico.at[row_num, 'term'] = season_year.upper()
```

In [9]:
```
for row_num, i in enumerate(df_sfsu.iloc[:,1]):
    term_split = i.split(' ')
    season_year = term_split[0] + ' ' + term_split[1]
    df_sfsu.at[row_num, 'term'] = season_year.upper()

for row_num, class_num in enumerate(df_sfsu['class_number']):
    class_split = class_num.split(' ')
    df_sfsu.at[row_num, 'class_number'] = class_split[0] + class_split[1]
```

In [10]:

```
for row_num, i in enumerate(df_sdsu.iloc[:,3]):
    term_split = i.split(' ')
    if len(term_split[0]) == 1:
        term_split[0] = '200' + term_split[0]
    else:
        term_split[0] = '20' + term_split[0]
    df_sdsu.at[row_num, 'term'] = (term_split[1] + ' ' + term_split[0]).upper
()
```

In [11]:

```
for row_num, i in enumerate(df_uci.iloc[:,3]):
    term_split = i.split(' ')
    df_uci.at[row_num, 'term'] = (term_split[1] + ' ' + term_split[0]).upper()
```

In [12]:

```
for row_num, i in enumerate(df_ucsc.iloc[:, 3]):
    term_split = i.split(' ')
    if len(term_split[1]) < 4:
        year = term_split[1].split('0')
        new_year = year[0] + '00' + year[1]
        term_split[1] = new_year
    df_ucsc.at[row_num, 'term'] = (term_split[0] + ' ' + term_split[1]).upper
()
```

In [13]:

```
# Goes through all the data and adds the percentages of class to the end of th
e dataframe
```

In [14]:

```
def add_percentage(self, num_enroll, total_size):
    self['Percentage'] = 0.
    for row_num, i in enumerate(self.iloc[:, num_enroll]):
        try:
            float(self.iloc[row_num, total_size])
        except:
            self.iloc[row_num,total_size] = self.iloc[row_num, total_size].spl
it('W')[0]
        if float(self.iloc[row_num, total_size]) == 0:
            self.iloc[row_num, total_size] = self.iloc[row_num, num_enroll]
        if float(self.iloc[row_num, total_size]) == 0 and float(self.iloc[row_
num, num_enroll]) ==0:
            self.iloc[row_num, total_size ] = 1;
        percentage = float(self.iloc[row_num, num_enroll])/float(self.iloc[row
_num,total_size])
        if percentage > 1:
            percentage = 1
        if percentage <= 0:
            percentage = None
        self.at[row_num, 'Percentage'] = percentage
    return self
```

In [15]:

```
# Deletes all dataframe rows with Nan and cleans up the class
```

In [16]:
```
def clean_nan(self):
    self = self.replace(0.,np.NaN)
    self = self.dropna()
    return self
```

In [17]:
```
def clean_class_nums(self):
    for row_num,i in enumerate(self.iloc[:,2]):
        cs_classes = i.split(' ')
        self.iloc[row_num,2] = cs_classes[0].strip()
    return self
```

In [18]:
```
# Changes rice university current enrollement to an integer and removes any values greater than 500.
# Any values greater than 500 are noise, and should be removed
```

In [19]:
```
for num, i in enumerate(df_rice.iloc[:,0]):
    df_rice.iloc[num, 0] = int(re.sub('[^0-9]', '', i))
    if int(df_rice.iloc[num,0]) >= 500:
        df_rice.iloc[num,0] = 0
    df_rice.iloc[num, 1] = int(re.sub('[^0-9]', '', df_rice.iloc[num,1]))

for num, i in enumerate(df_rice.iloc[:,2]):
    splitted = i.split()
    df_rice.iloc[num,2] = splitted[0] + splitted[1]
```

In [20]:
```
# Adds the perccentages to every school and cleans up all NaN's in data.
# Change the data such that all data has the same columns indices.
```

In [21]:
```
add_percentage(df_chico, 0, 1)
add_percentage(df_sfsu, 3, 4)
add_percentage(df_ucsc, 0, 1)
add_percentage(df_sdsu, 0, 1)
add_percentage(df_uci, 0, 1)
add_percentage(df_harvey, 0, 1)
add_percentage(df_panoma,0, 1)
add_percentage(df_rice, 0, 1)

col = ['num_people_enrolled', 'total_class_size', 'class_number', 'term', 'Prof Name', 'Percentage', 'Unnamed: 0']
df_sfsu = df_sfsu.loc[:,col]
df_sfsu.drop('Unnamed: 0', axis = 1, inplace = True)
df_sfsu.columns = ['num_people_enrolled', 'total_class_size', 'class_number', 'term' ,'professor', 'Percentage']

df_chico = clean_nan(df_chico)
df_sdsu= clean_nan(df_sdsu)
df_ucsc = clean_nan(df_ucsc)
df_uci = clean_nan(df_uci)
df_sfsu = clean_nan(df_sfsu)
df_rice = clean_nan(df_rice)
df_harvey = clean_nan(df_harvey)

df_panoma = clean_class_nums(df_panoma)
df_harvey = clean_class_nums(df_harvey)
```

In [22]:
```
# Splits the data into percentages for each term, so it makes it easier to plot
```

In [23]:
```
def getPercentages(self, term_location):
    unique_vals = []
    newData = []
    newTerm = []
    for i in self.iloc[:,term_location]:
        if i not in unique_vals:
            unique_vals.append(i)
    newTerm = list(unique_vals)
    for i in newTerm:
        newData.append(self.Percentage[self.term == i])
    return newData, newTerm
```

In [24]:
```
# Goes through each term and sorts the term with accordance to the data
```

In [25]:
```
def sort_term(term, data):
    to_sort = []
    for i in term:
        if 'FALL' in i:
            split = i.replace('FALL', '')
            split = split + '2'
        if 'SPRING' in i:
            split = i.replace('SPRING', '')
            split = split+'3'
        if 'WINTER' in i:
            split = i.replace('WINTER', '')
            split = split + '1'
        to_sort.append((int(int(split)%1000)))
    _, term, data = zip(*sorted(zip(to_sort, term, data)))
    return term, data
```

In [26]:
```
def term_to_num(term):
    quant = {'SPRING':.33, 'FALL':.66,  'WINTER':.99}
    df = pd.DataFrame()
    term_split = term.split(' ')
    return quant[term_split[0]] + int(term_split[1])

def floor_term_num(term):
    df = pd.DataFrame()
    return float(term.split()[1])
```

In [27]:
```
# Adds a term_num to the data so that when data is plot, spring, fall, and winter are equally distributed.
# Adds a year term to the data so when doing year graphs, they correlate to a year and not a term.
# Sorts every data frame by the term_num so that each dataframe is in order
# Removes any classes that have a class size of 2
```

In [28]:
```
df_uci['term_num'] = df_uci['term'].apply(term_to_num)
df_ucsc['term_num'] = df_ucsc['term'].apply(term_to_num)
df_chico['term_num'] = df_chico['term'].apply(term_to_num)
df_sdsu['term_num'] = df_sdsu['term'].apply(term_to_num)
df_harvey['term_num'] = df_harvey['term'].apply(term_to_num)
df_panoma['term_num'] = df_panoma['term'].apply(term_to_num)
df_sfsu['term_num'] = df_sfsu['term'].apply(term_to_num)
df_rice['term_num'] = df_rice['term'].apply(term_to_num)

df_ucsc = df_ucsc.sort_values(by = ['term_num'])
df_uci = df_uci.sort_values(by = ['term_num'])
df_chico = df_chico.sort_values(by = ['term_num'])
df_sdsu = df_sdsu.sort_values(by = ['term_num'])
df_harvey = df_harvey.sort_values(by = ['term_num'])
df_sfsu = df_sfsu.sort_values(by=['term_num'])
df_rice = df_rice.sort_values(by=['term_num'])

#df_rice
```

In [29]:

```
def find_popular_class(self):
    unique_classes = {}
    difference_list = list()
    for i in self['class_number']:
        if i not in unique_classes:
            unique_classes.update({i: 1})
        else:
            unique_classes.update({i: unique_classes.get(i) + 1})
    class_popular = {}
    for i in unique_classes:
        if unique_classes.get(i) > 7 and unique_classes.get(i) < 15:
            class_popular.update({i:unique_classes.get(i)})
    for i in class_popular:
        this_class = self[self['class_number'] == i]
        percent_lower = 0
        percent_upper = 0
        lower = len(this_class)/2
        top = len(this_class)
        for i in range(int(lower)):
            percent_lower += this_class.iloc[i]['Percentage']
        for i in range(int(lower), int(top)):
            percent_upper += this_class.iloc[i]['Percentage']
        percent_lower /= int(lower)
        percent_upper /= int(top) - int(lower)
        difference = percent_upper - percent_lower
        difference_list.append((difference, this_class))
    difference_list = sorted(difference_list, key = lambda x: x[0], reverse =
False)

    popular_classes = list()
    for percent, actual_class in difference_list[:5]:
        popular_classes.append(actual_class)
    return(popular_classes)

#find_popular_class(df_ucsc) #ucsc 181, 116, 122 Computer Security, Databases,
software engineering
#find_popular_class(df_sdsu) # 537 - PROGRAMMING FOR GIS, software engineering
- 532, 503 Databases
#find_popular_class(df_uci)  #uci 35320 DataMining, intro to data mgmt 34200
 (databases), Computer Security, 34350 AI
#find_popular_class(df_chico) #430 software engineering, intro unix 144
#find_popular_class(df_harvey) #neural networks 152, Computer networks 125, Ma
chine Learning
#find_popular_class(df_sfsu) #212 software engineering # 656 Computer Organiza
tion # 305 Social and ethical computing,
#Computer eval 641
```

In [30]:

```
#classifying lower and upper div class for rice
#upper is 300+
#lower is < 300
df_rice['class_ref'] = df_rice['class_num'].str.extract('(\d\d\d)', expand=Tru
e)
df_rice_lower = df_rice[df_rice['class_ref'] < '300']
df_rice_upper = df_rice[df_rice['class_ref'] > '299']
```

In [31]:

```
#classifying lower and upper div class for harvey mudd
#upper is 100+
#lower is < 100
df_harvey['class_ref'] = df_harvey['class_number'].str.extract('(\d\d\d)', expand=True)
df_harvey_lower = df_harvey[df_harvey['class_ref'] < '100']
df_harvey_upper = df_harvey[df_harvey['class_ref'] >= '100']
```

In [32]:

```
#classifying lower and upper div class for sdsu
#https://registrar.sdsu.edu/students/registration/course_numbering

#upper is 300-599
#lower is 100-299
df_sdsu['class_ref'] = df_sdsu['class_number'].str.extract('(\d\d\d)', expand=True)
df_sdsu_lower = df_sdsu[(df_sdsu['class_ref'] >= '100') & (df_sdsu['class_ref'] <= '299')]
df_sdsu_upper = df_sdsu[(df_sdsu['class_ref'] >= '300') & (df_sdsu['class_ref'] <= '599')]

#df_sdsu_lower
#df_sdsu_upper
```

In [33]:

```
#classifying lower and upper div class for sfsu
#https://ueap.sfsu.edu/content/curriculum/courses/x-course-numbering-system-sfsu-course-review-approval-guidelines

#upper is 300-699
#lower is 100-299
df_sfsu['class_ref'] = df_sfsu['class_number'].str.extract('(\d\d\d)', expand=True)
df_sfsu_lower = df_sfsu[(df_sfsu['class_ref'] >= '100') & (df_sfsu['class_ref'] <= '299')]
df_sfsu_upper = df_sfsu[(df_sfsu['class_ref'] >= '300') & (df_sfsu['class_ref'] <= '699')]

#df_sfsu_lower
#df_sfsu_upper
```

In [34]:

```
#classifying lower and upper div class for chico
#https://www.csuchico.edu/pres/em/2017/17-012.shtml

#upper is 300-599
#lower is 100-299
df_chico['class_ref'] = df_chico['class_number'].str.extract('(\d\d\d)', expand=True)
df_chico_lower = df_chico[(df_chico['class_ref'] >= '100') & (df_chico['class_ref'] <= '299')]
df_chico_upper = df_chico[(df_chico['class_ref'] >= '300') & (df_chico['class_ref'] <= '599')]

#df_chico_lower
#df_chico_upper
```

In [35]:

```
#classifying lower and upper div class for ucsc

#upper is 100-199
#lower is < 99

df_ucsc['class_ref'] = df_ucsc['class_number'].str.extract('(\d\d\d)', expand=
True)
df_ucsc_lower = df_ucsc[df_ucsc['class_ref'].isna()]
df_ucsc_upper = df_ucsc[(df_ucsc['class_ref'] >= '100') & (df_ucsc['class_ref'
] <= '199')]

#df_ucsc_lower
#df_ucsc_upper
```

In [36]:

```
def avg(self, to_avg = 'term_num', data_avg = 'num_people_enrolled'):
    term_size = []
    unique_term = self[to_avg].unique()
    for i in unique_term:
        term_selfdf = self[self[to_avg] == i]
        average_size = 0
        for i in term_selfdf[data_avg]:
            average_size += i
        term_size.append(average_size/len(term_selfdf))
    return term_size
```

In [37]:

```
def avg_sum(self, to_avg = 'term_num', data_avg = 'num_people_enrolled'):
    term_size = []
    unique_term = self[to_avg].unique()
    for i in unique_term:
        term_selfdf = self[self[to_avg] == i]
        average_size = 0
        for i in term_selfdf[data_avg]:
            average_size += i
        term_size.append(average_size)
    return term_size
```

In [38]:

```
def jitter(self, amount = .1):
    return self + amount * np.random.rand(len(self))- amount/2
```

In [39]:

```
df_ucsc_lower['term_num'] = df_ucsc_lower['term'].apply(term_to_num)
df_chico_lower['term_num'] = df_chico_lower['term'].apply(term_to_num)
df_sdsu_lower['term_num'] = df_sdsu_lower['term'].apply(term_to_num)
df_harvey_lower['term_num'] = df_harvey_lower['term'].apply(term_to_num)
df_sfsu_lower['term_num'] = df_sfsu_lower['term'].apply(term_to_num)
df_rice_lower['term_num'] = df_rice_lower['term'].apply(term_to_num)

df_ucsc_lower = df_ucsc_lower.sort_values(by = ['term_num'])
df_chico_lower = df_chico_lower.sort_values(by = ['term_num'])
df_sdsu_lower = df_sdsu_lower.sort_values(by = ['term_num'])
df_harvey_lower = df_harvey_lower.sort_values(by = ['term_num'])
df_sfsu_lower = df_sfsu_lower.sort_values(by=['term_num'])
df_rice_lower = df_rice_lower.sort_values(by=['term_num'])
```

```
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  """Entry point for launching an IPython kernel.
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy

C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports u
ntil
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  after removing the cwd from sys.path.
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  """
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
```

In [40]:

```
df_ucsc_lower['year'] = df_ucsc_lower['term'].apply(floor_term_num)
df_chico_lower['year'] = df_chico_lower['term'].apply(floor_term_num)
df_sdsu_lower['year'] = df_sdsu_lower['term'].apply(floor_term_num)
df_sfsu_lower['year'] = df_sfsu_lower['term'].apply(floor_term_num)
df_rice_lower['year'] = df_rice_lower['term'].apply(floor_term_num)
df_harvey_lower['year'] = df_harvey_lower['term'].apply(floor_term_num)
```

In [41]:

```
df_ucsc_upper['term_num'] = df_ucsc_upper['term'].apply(term_to_num)
df_chico_upper['term_num'] = df_chico_upper['term'].apply(term_to_num)
df_sdsu_upper['term_num'] = df_sdsu_upper['term'].apply(term_to_num)
df_harvey_upper['term_num'] = df_harvey_upper['term'].apply(term_to_num)
df_sfsu_upper['term_num'] = df_sfsu_upper['term'].apply(term_to_num)
df_rice_upper['term_num'] = df_rice_upper['term'].apply(term_to_num)

df_ucsc_upper = df_ucsc_upper.sort_values(by = ['term_num'])
df_chico_upper = df_chico_upper.sort_values(by = ['term_num'])
df_sdsu_upper = df_sdsu_upper.sort_values(by = ['term_num'])
df_harvey_upper = df_harvey_upper.sort_values(by = ['term_num'])
df_sfsu_upper = df_sfsu_upper.sort_values(by=['term_num'])
df_rice_upper = df_rice_upper.sort_values(by=['term_num'])
```

```
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:1: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  """Entry point for launching an IPython kernel.
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:2: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy

C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  This is separate from the ipykernel package so we can avoid doing imports u
ntil
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  after removing the cwd from sys.path.
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:5: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
  """
C:\Users\terry\Anaconda3\lib\site-packages\ipykernel_launcher.py:6: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/st
able/indexing.html#indexing-view-versus-copy
```

In [42]:

```python
df_ucsc_upper['year'] = df_ucsc_upper['term'].apply(floor_term_num)
df_chico_upper['year'] = df_chico_upper['term'].apply(floor_term_num)
df_sdsu_upper['year'] = df_sdsu_upper['term'].apply(floor_term_num)
df_sfsu_upper['year'] = df_sfsu_upper['term'].apply(floor_term_num)
df_rice_upper['year'] = df_rice_upper['term'].apply(floor_term_num)
df_harvey_upper['year'] = df_harvey_upper['term'].apply(floor_term_num)
```

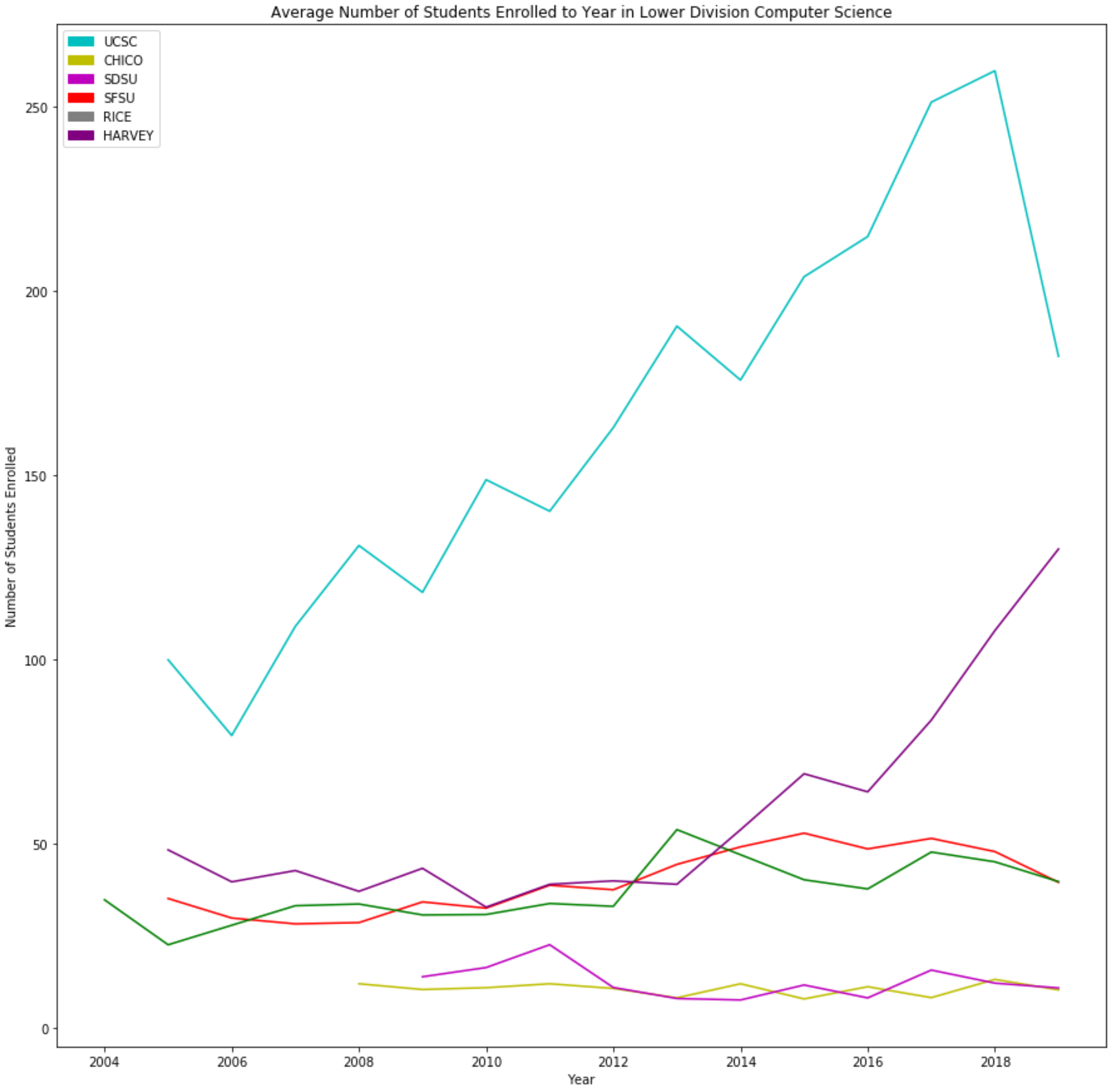
In [43]:

```
# Adds the scatter plots in terms of year for every school in UC, State, and private schools
```

In [44]:

```
fig, axes = plt.subplots(1,1, figsize=(15,15))

df_ucsc_lower = df_ucsc_lower[df_ucsc_lower['num_people_enrolled'] >= 3]
df_chico_lower = df_chico_lower[df_chico_lower['num_people_enrolled'] >= 3]
df_sdsu_lower = df_sdsu_lower[df_sdsu_lower['num_people_enrolled'] >= 3]
df_sfsu_lower = df_sfsu_lower[df_sfsu_lower['num_people_enrolled'] >= 3]
df_harvey_lower = df_harvey_lower[df_harvey_lower['num_people_enrolled'] >= 3]
df_rice_lower = df_rice_lower[df_rice_lower['num_people_enrolled'] >= 3]

axes.plot(df_ucsc_lower['year'].unique(), avg(df_ucsc_lower, to_avg = 'year'), 'c-')
axes.plot(df_chico_lower['year'].unique(), avg(df_chico_lower, to_avg = 'year'), 'y-')
axes.plot(df_sdsu_lower['year'].unique(), avg(df_sdsu_lower, to_avg = 'year'), 'm-')
axes.plot(df_sfsu_lower['year'].unique(), avg(df_sfsu_lower, to_avg = 'year'), 'r-')
axes.plot(df_harvey_lower['year'].unique(), avg(df_harvey_lower, to_avg = 'year'), 'g-')
axes.plot(df_rice_lower['year'].unique(), avg(df_rice_lower, to_avg = 'year'), 'purple')

ucsc_patch = mpatches.Patch(color='c', label='UCSC')
chico_patch = mpatches.Patch(color='y', label='CHICO')
sdsu_patch = mpatches.Patch(color='m', label='SDSU')
sfsu_patch = mpatches.Patch(color='r', label='SFSU')
rice_patch = mpatches.Patch(color = 'grey', label = 'RICE')
harvey_patch = mpatches.Patch(color = 'purple', label  = 'HARVEY')
plt.legend(handles=[ucsc_patch, chico_patch, sdsu_patch, sfsu_patch, rice_patch, harvey_patch])

axes.set_title('Average Number of Students Enrolled to Year in Lower Division Computer Science')
axes.set_xlabel('Year')
axes.set_ylabel('Number of Students Enrolled')
```

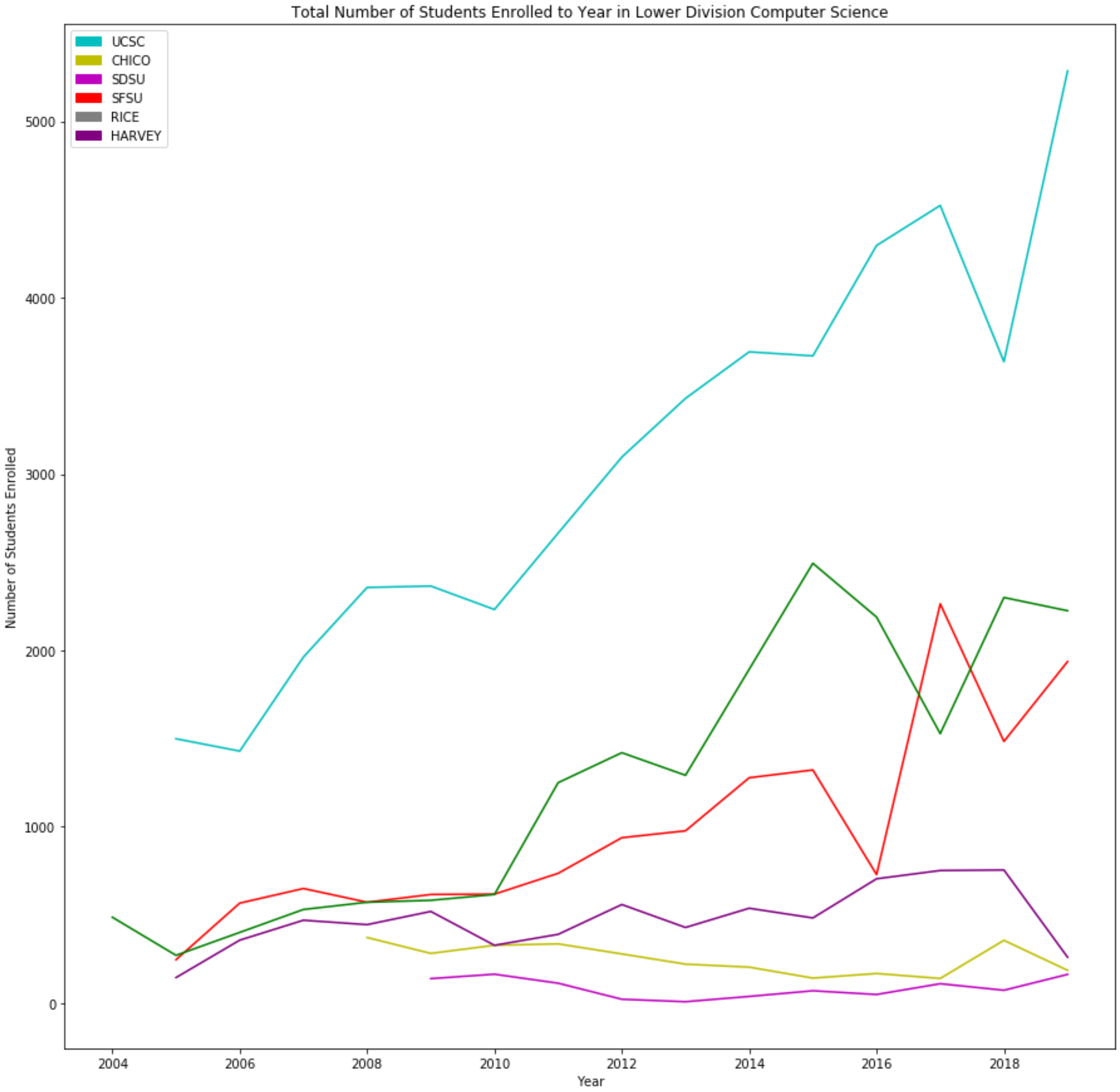
```
Out[44]: Text(0, 0.5, 'Number of Students Enrolled')
```



In [45]:

```
fig, axes = plt.subplots(1,1, figsize=(15,15))

df_ucsc_lower = df_ucsc_lower[df_ucsc_lower['num_people_enrolled'] >= 3]
df_chico_lower = df_chico_lower[df_chico_lower['num_people_enrolled'] >= 3]
df_sdsu_lower = df_sdsu_lower[df_sdsu_lower['num_people_enrolled'] >= 3]
df_sfsu_lower = df_sfsu_lower[df_sfsu_lower['num_people_enrolled'] >= 3]
df_harvey_lower = df_harvey_lower[df_harvey_lower['num_people_enrolled'] >= 3]
df_rice_lower = df_rice_lower[df_rice_lower['num_people_enrolled'] >= 3]

axes.plot(df_ucsc_lower['year'].unique(), avg_sum(df_ucsc_lower, to_avg = 'yea
r'), 'c-')
axes.plot(df_chico_lower['year'].unique(), avg_sum(df_chico_lower, to_avg = 'y
ear'), 'y-')
axes.plot(df_sdsu_lower['year'].unique(), avg_sum(df_sdsu_lower, to_avg = 'yea
r'), 'm-')
axes.plot(df_sfsu_lower['year'].unique(), avg_sum(df_sfsu_lower, to_avg = 'yea
r'), 'r-')
axes.plot(df_harvey_lower['year'].unique(), avg_sum(df_harvey_lower, to_avg =
'year'), 'g-')
axes.plot(df_rice_lower['year'].unique(), avg_sum(df_rice_lower, to_avg = 'yea
r'), 'purple')

ucsc_patch = mpatches.Patch(color='c', label='UCSC')
chico_patch = mpatches.Patch(color='y', label='CHICO')
sdsu_patch = mpatches.Patch(color='m', label='SDSU')
sfsu_patch = mpatches.Patch(color='r', label='SFSU')
rice_patch = mpatches.Patch(color = 'grey', label = 'RICE')
harvey_patch = mpatches.Patch(color = 'purple', label  = 'HARVEY')
plt.legend(handles=[ucsc_patch, chico_patch, sdsu_patch, sfsu_patch, rice_patc
h, harvey_patch])

axes.set_title('Total Number of Students Enrolled to Year in Lower Division Co
mputer Science')
axes.set_xlabel('Year')
axes.set_ylabel('Number of Students Enrolled')
```

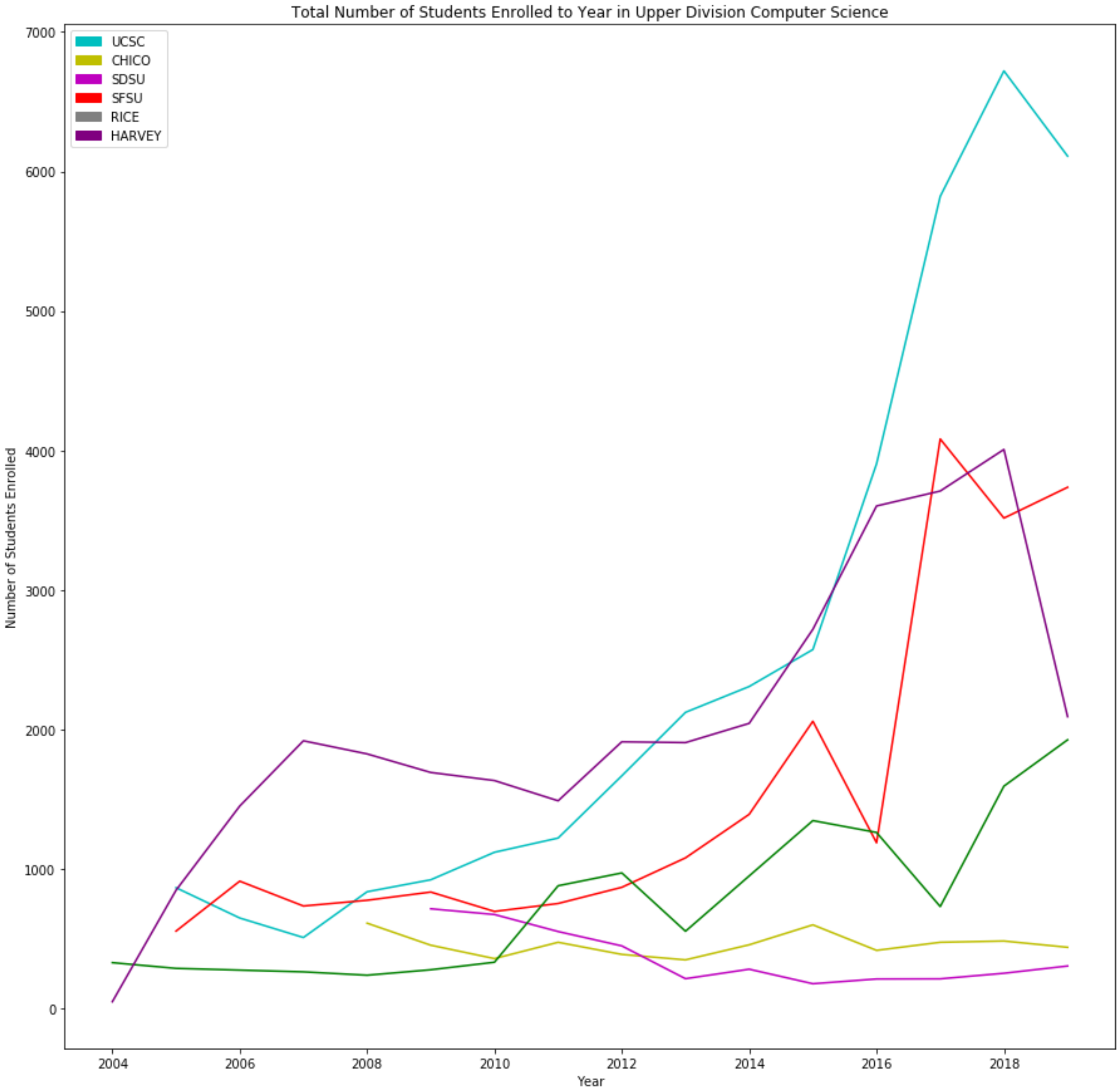
```
Out[45]: Text(0, 0.5, 'Number of Students Enrolled')
```



In [46]:

```
fig, axes = plt.subplots(1,1, figsize=(15,15))

df_ucsc_upper = df_ucsc_upper[df_ucsc_upper['num_people_enrolled'] >= 3]
df_chico_upper = df_chico_upper[df_chico_upper['num_people_enrolled'] >= 3]
df_sdsu_upper = df_sdsu_upper[df_sdsu_upper['num_people_enrolled'] >= 3]
df_sfsu_upper = df_sfsu_upper[df_sfsu_upper['num_people_enrolled'] >= 3]
df_harvey_upper = df_harvey_upper[df_harvey_upper['num_people_enrolled'] >= 3]
df_rice_upper = df_rice_upper[df_rice_upper['num_people_enrolled'] >= 3]


axes.plot(df_ucsc_upper['year'].unique(), avg_sum(df_ucsc_upper, to_avg = 'year'), 'c-')
axes.plot(df_chico_upper['year'].unique(), avg_sum(df_chico_upper, to_avg = 'year'), 'y-')
axes.plot(df_sdsu_upper['year'].unique(), avg_sum(df_sdsu_upper, to_avg = 'year'), 'm-')
axes.plot(df_sfsu_upper['year'].unique(), avg_sum(df_sfsu_upper, to_avg = 'year'), 'r-')
axes.plot(df_harvey_upper['year'].unique(), avg_sum(df_harvey_upper, to_avg = 'year'), 'g-')
axes.plot(df_rice_upper['year'].unique(), avg_sum(df_rice_upper, to_avg = 'year'), 'purple')

ucsc_patch = mpatches.Patch(color='c', label='UCSC')
chico_patch = mpatches.Patch(color='y', label='CHICO')
sdsu_patch = mpatches.Patch(color='m', label='SDSU')
sfsu_patch = mpatches.Patch(color='r', label='SFSU')
rice_patch = mpatches.Patch(color = 'grey', label = 'RICE')
harvey_patch = mpatches.Patch(color = 'purple', label  = 'HARVEY')
plt.legend(handles=[ucsc_patch, chico_patch, sdsu_patch, sfsu_patch, rice_patch, harvey_patch])

axes.set_title('Total Number of Students Enrolled to Year in Upper Division Computer Science')
axes.set_xlabel('Year')
axes.set_ylabel('Number of Students Enrolled')
```

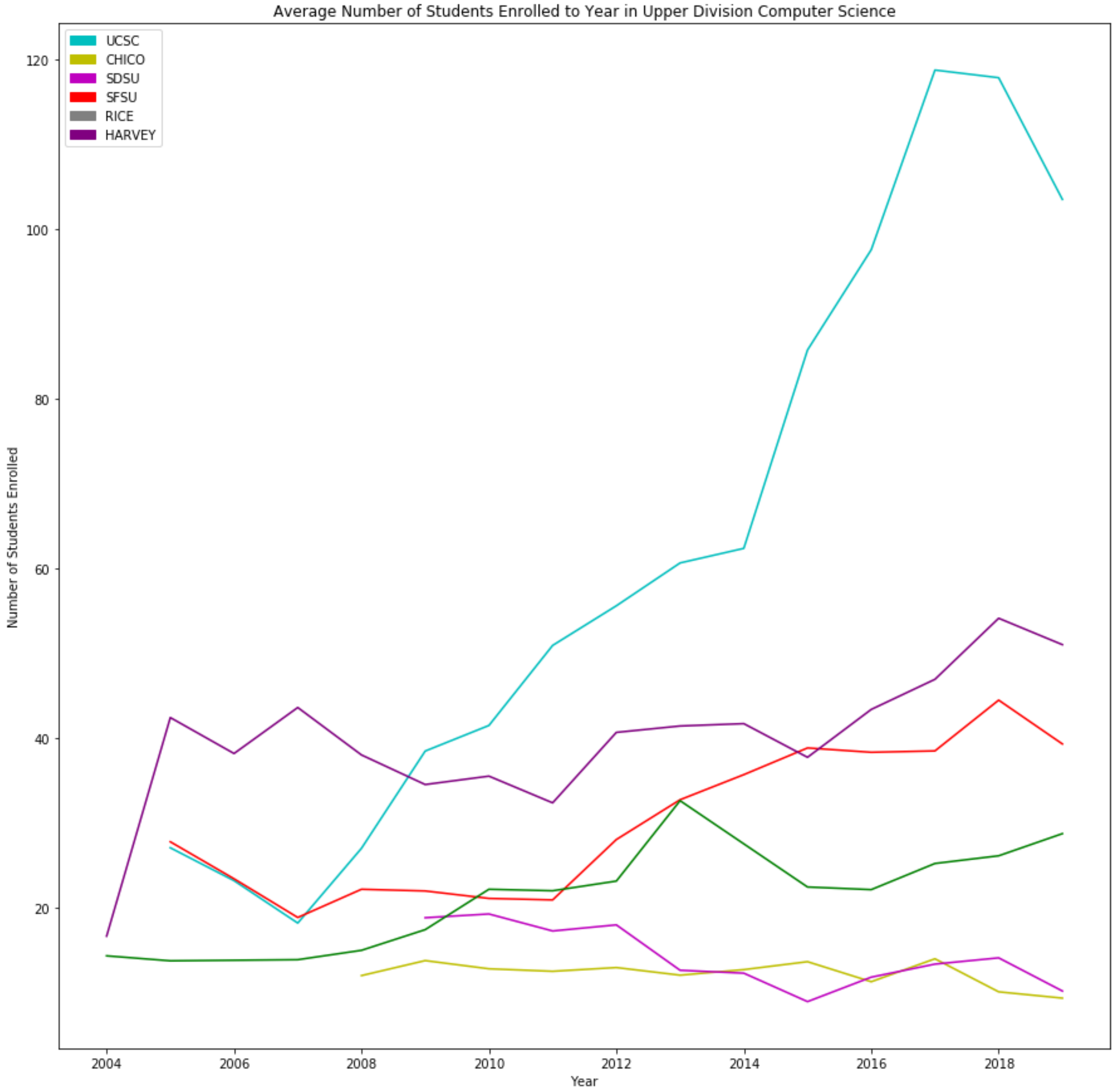
```
Out[46]: Text(0, 0.5, 'Number of Students Enrolled')
```



In [47]:

```
fig, axes = plt.subplots(1,1, figsize=(15,15))

df_ucsc_upper = df_ucsc_upper[df_ucsc_upper['num_people_enrolled'] >= 3]
df_chico_upper = df_chico_upper[df_chico_upper['num_people_enrolled'] >= 3]
df_sdsu_upper = df_sdsu_upper[df_sdsu_upper['num_people_enrolled'] >= 3]
df_sfsu_upper = df_sfsu_upper[df_sfsu_upper['num_people_enrolled'] >= 3]
df_harvey_upper = df_harvey_upper[df_harvey_upper['num_people_enrolled'] >= 3]
df_rice_upper = df_rice_upper[df_rice_upper['num_people_enrolled'] >= 3]


axes.plot(df_ucsc_upper['year'].unique(), avg(df_ucsc_upper, to_avg = 'year'), 'c-')
axes.plot(df_chico_upper['year'].unique(), avg(df_chico_upper, to_avg = 'year'), 'y-')
axes.plot(df_sdsu_upper['year'].unique(), avg(df_sdsu_upper, to_avg = 'year'), 'm-')
axes.plot(df_sfsu_upper['year'].unique(), avg(df_sfsu_upper, to_avg = 'year'), 'r-')
axes.plot(df_harvey_upper['year'].unique(), avg(df_harvey_upper, to_avg = 'year'), 'g-')
axes.plot(df_rice_upper['year'].unique(), avg(df_rice_upper, to_avg = 'year'), 'purple')

ucsc_patch = mpatches.Patch(color='c', label='UCSC')
chico_patch = mpatches.Patch(color='y', label='CHICO')
sdsu_patch = mpatches.Patch(color='m', label='SDSU')
sfsu_patch = mpatches.Patch(color='r', label='SFSU')
rice_patch = mpatches.Patch(color = 'grey', label = 'RICE')
harvey_patch = mpatches.Patch(color = 'purple', label  = 'HARVEY')
plt.legend(handles=[ucsc_patch, chico_patch, sdsu_patch, sfsu_patch, rice_patch, harvey_patch])

axes.set_title('Average Number of Students Enrolled to Year in Upper Division Computer Science')
axes.set_xlabel('Year')
axes.set_ylabel('Number of Students Enrolled')
```

```
Out[47]: Text(0, 0.5, 'Number of Students Enrolled')
```

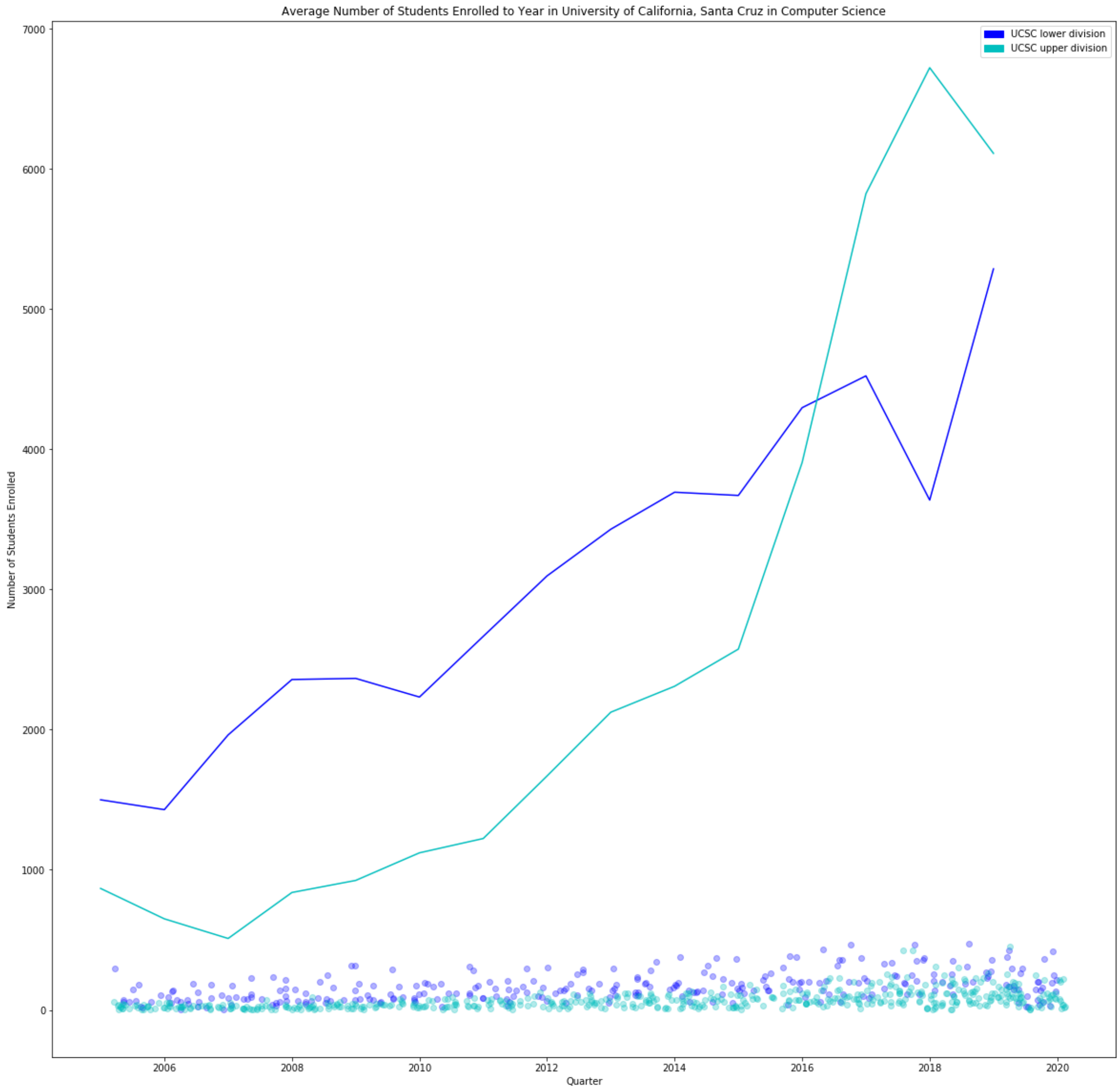


```
In [48]: #adds a plot to show comparison between lower/upper div
```

In [49]:

```
fig, axes = plt.subplots(1,1, figsize=(20,20))

ucsc_jitter_lower, ucsc_n_jitter_lower = jitter(df_ucsc_lower['term_num'], .3
), jitter(df_ucsc_lower['num_people_enrolled'], 1)
ucsc_jitter_upper, ucsc_n_jitter_upper = jitter(df_ucsc_upper['term_num'], .3
), jitter(df_ucsc_upper['num_people_enrolled'], 1)


axes.scatter(ucsc_jitter_lower, ucsc_n_jitter_lower, alpha = .3, color = 'b')
axes.scatter(ucsc_jitter_upper, ucsc_n_jitter_upper, alpha = .3, color = 'c')

axes.plot(df_ucsc_lower['year'].unique(), avg(df_ucsc_lower, to_avg = 'year'),
'b-')
axes.plot(df_ucsc_upper['year'].unique(), avg(df_ucsc_upper, to_avg = 'year'),
'c-')

ucsc_patch_lower = mpatches.Patch(color='b', label='UCSC lower division')
ucsc_patch_upper = mpatches.Patch(color='c', label='UCSC upper division')

plt.legend(handles=[ucsc_patch_lower, ucsc_patch_upper])

axes.set_title('Average Number of Students Enrolled to Year in University of C
alifornia, Santa Cruz in Computer Science')
axes.set_xlabel('Quarter')
axes.set_ylabel('Number of Students Enrolled')
```

Out[49]:
```
Text(0, 0.5, 'Number of Students Enrolled')
```

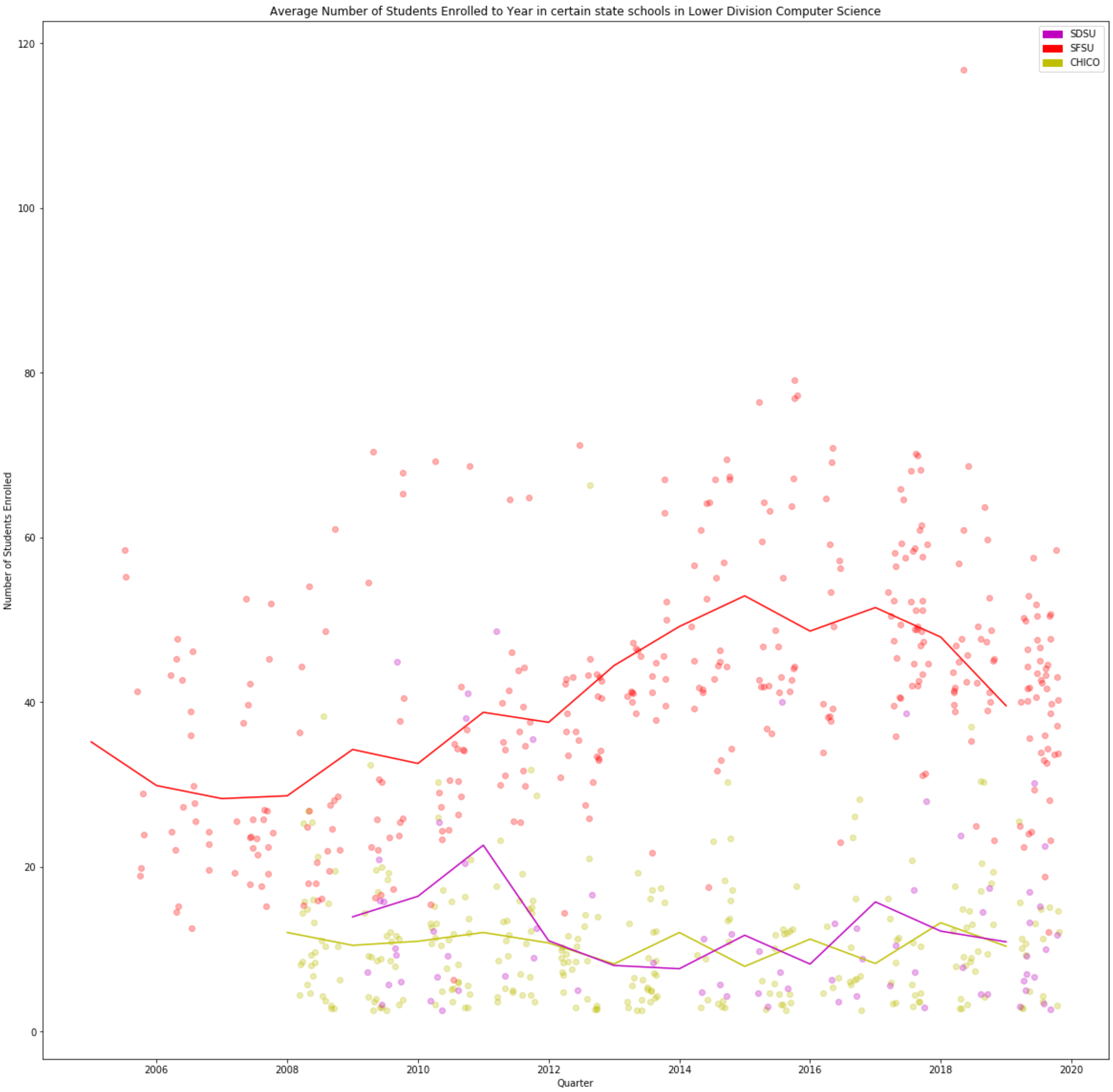


In [50]:
```
#adds plot to show lower div in selected state schools
```

In [51]:

```
fig, axes = plt.subplots(1,1, figsize=(20,20))
chico_jitter_lower, chico_n_jitter_lower = jitter(df_chico_lower['term_num'],
.3), jitter(df_chico_lower['num_people_enrolled'], 1)
sdsu_jitter_lower, sdsu_n_jitter_lower = jitter(df_sdsu_lower['term_num'], .3
), jitter(df_sdsu_lower['num_people_enrolled'], 1)
sfsu_jitter_lower, sfsu_n_jitter_lower = jitter(df_sfsu_lower['term_num'], .3
), jitter(df_sfsu_lower['num_people_enrolled'], 1)

axes.scatter(chico_jitter_lower, chico_n_jitter_lower, alpha = .3, color = 'y'
)
axes.scatter(sdsu_jitter_lower, sdsu_n_jitter_lower, alpha = .3, color = 'm')
axes.scatter(sfsu_jitter_lower, sfsu_n_jitter_lower, alpha = .3, color = 'r')


axes.plot(df_chico_lower['year'].unique(), avg(df_chico_lower, to_avg = 'year'
), 'y-')
axes.plot(df_sdsu_lower['year'].unique(), avg(df_sdsu_lower, to_avg = 'year'),
'm-')
axes.plot(df_sfsu_lower['year'].unique(), avg(df_sfsu_lower, to_avg = 'year'),
'r-')

chico_patch = mpatches.Patch(color='y', label='CHICO')
sdsu_patch = mpatches.Patch(color='m', label='SDSU')
sfsu_patch = mpatches.Patch(color='r', label='SFSU')

plt.legend(handles=[sdsu_patch, sfsu_patch, chico_patch])

axes.set_title('Average Number of Students Enrolled to Year in certain state s
chools in Lower Division Computer Science')
axes.set_xlabel('Quarter')
axes.set_ylabel('Number of Students Enrolled')
```

```
Out[51]: Text(0, 0.5, 'Number of Students Enrolled')
```

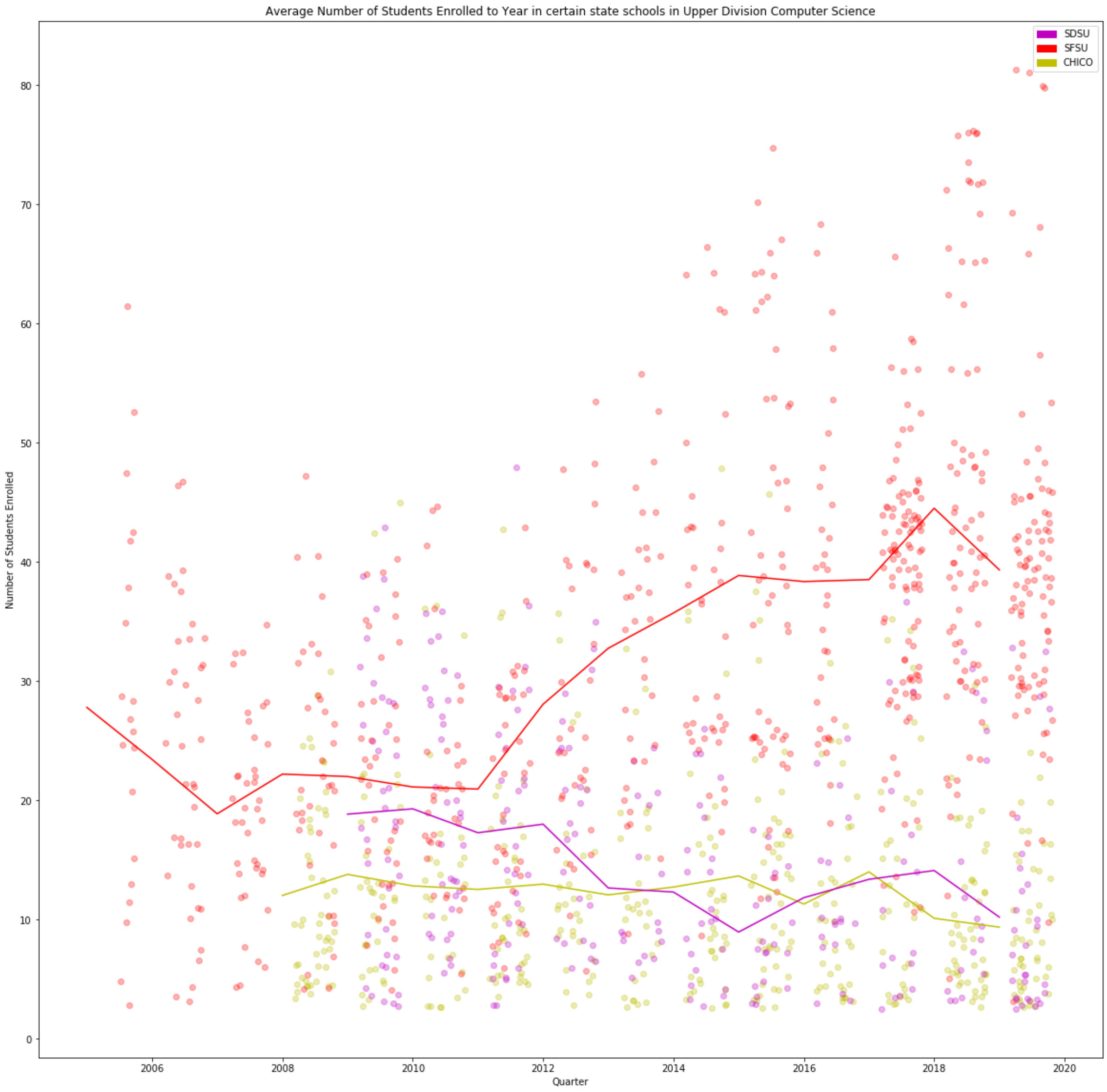


```
In [52]: #adds plot to show selected upper div class in state schools
```

In [53]:

```
fig, axes = plt.subplots(1,1, figsize=(20,20))
chico_jitter_upper, chico_n_jitter_upper = jitter(df_chico_upper['term_num'],
.3), jitter(df_chico_upper['num_people_enrolled'], 1)
sdsu_jitter_upper, sdsu_n_jitter_upper = jitter(df_sdsu_upper['term_num'], .3
), jitter(df_sdsu_upper['num_people_enrolled'], 1)
sfsu_jitter_upper, sfsu_n_jitter_upper = jitter(df_sfsu_upper['term_num'], .3
), jitter(df_sfsu_upper['num_people_enrolled'], 1)

axes.scatter(chico_jitter_upper, chico_n_jitter_upper, alpha = .3, color = 'y'
)
axes.scatter(sdsu_jitter_upper, sdsu_n_jitter_upper, alpha = .3, color = 'm')
axes.scatter(sfsu_jitter_upper, sfsu_n_jitter_upper, alpha = .3, color = 'r')

axes.plot(df_chico_upper['year'].unique(), avg(df_chico_upper, to_avg = 'year'
), 'y-')
axes.plot(df_sdsu_upper['year'].unique(), avg(df_sdsu_upper, to_avg = 'year'),
'm-')
axes.plot(df_sfsu_upper['year'].unique(), avg(df_sfsu_upper, to_avg = 'year'),
'r-')


chico_patch = mpatches.Patch(color='y', label='CHICO')
sdsu_patch = mpatches.Patch(color='m', label='SDSU')
sfsu_patch = mpatches.Patch(color='r', label='SFSU')

plt.legend(handles=[sdsu_patch, sfsu_patch, chico_patch])

axes.set_title('Average Number of Students Enrolled to Year in certain state s
chools in Upper Division Computer Science')
axes.set_xlabel('Quarter')
axes.set_ylabel('Number of Students Enrolled')
```

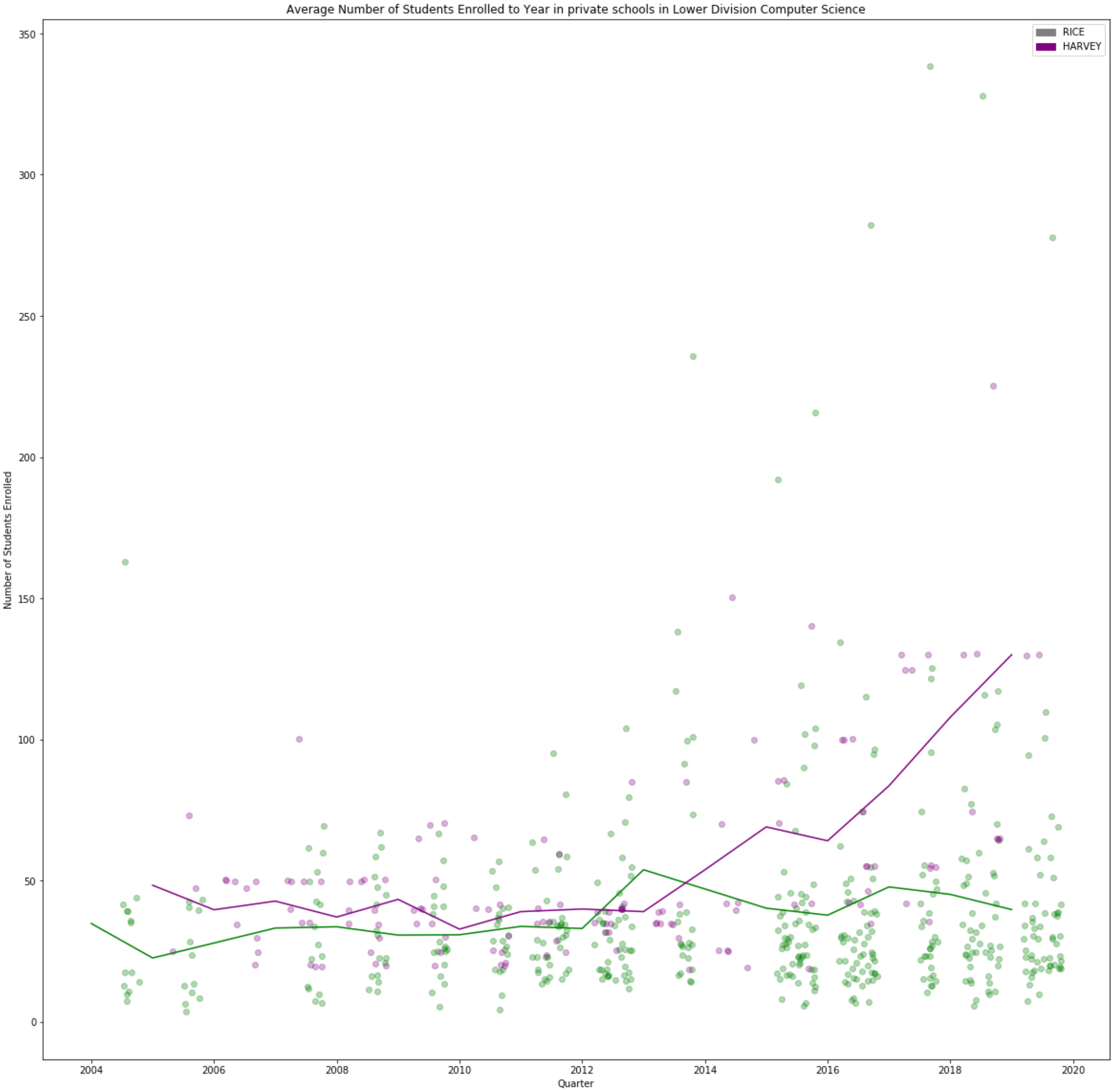
```
Out[53]: Text(0, 0.5, 'Number of Students Enrolled')
```



```
In [54]: #adds plot to show selected private school lower div class
```

In [55]:
```
fig, axes = plt.subplots(1,1, figsize=(20,20))

harvey_jitter_lower, harvey_n_jitter_lower = jitter(df_harvey_lower['term_num'], .3), jitter(df_harvey_lower['num_people_enrolled'], 1)
rice_jitter_lower, rice_n_jitter_lower = jitter(df_rice_lower['term_num'], .3), jitter(df_rice_lower['num_people_enrolled'], 1)

axes.scatter(harvey_jitter_lower, harvey_n_jitter_lower, alpha = .3, color = 'g')
axes.scatter(rice_jitter_lower, rice_n_jitter_lower, alpha = .3, color = 'purple')

axes.plot(df_harvey_lower['year'].unique(), avg(df_harvey_lower, to_avg = 'year'), 'g-')
axes.plot(df_rice_lower['year'].unique(), avg(df_rice_lower, to_avg = 'year'), 'purple')

rice_patch = mpatches.Patch(color = 'grey', label = 'RICE')
harvey_patch = mpatches.Patch(color = 'purple', label  = 'HARVEY')

plt.legend(handles=[rice_patch, harvey_patch])

axes.set_title('Average Number of Students Enrolled to Year in private schools in Lower Division Computer Science')
axes.set_xlabel('Quarter')
axes.set_ylabel('Number of Students Enrolled')
```

```
Out[55]: Text(0, 0.5, 'Number of Students Enrolled')
```

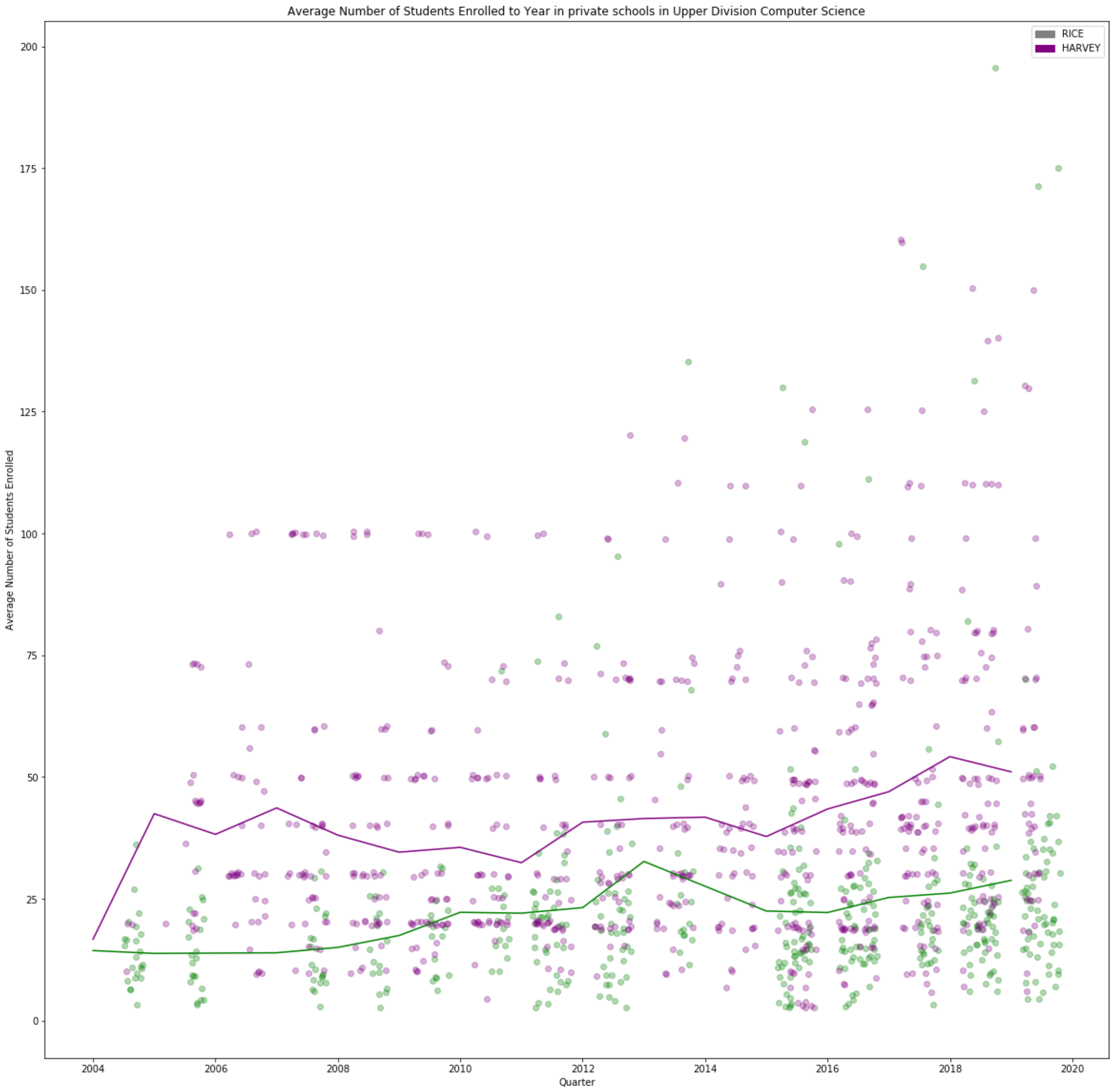


```
In [56]: #adds plot to show selected private school upper div class
```

In [57]:

```
fig, axes = plt.subplots(1,1, figsize=(20,20))

harvey_jitter_upper, harvey_n_jitter_upper = jitter(df_harvey_upper['term_num'
], .3), jitter(df_harvey_upper['num_people_enrolled'], 1)
rice_jitter_upper, rice_n_jitter_upper = jitter(df_rice_upper['term_num'], .3
), jitter(df_rice_upper['num_people_enrolled'], 1)

axes.scatter(harvey_jitter_upper, harvey_n_jitter_upper, alpha = .3, color =
'g')
axes.scatter(rice_jitter_upper, rice_n_jitter_upper, alpha = .3, color = 'purp
le')

axes.plot(df_harvey_upper['year'].unique(), avg(df_harvey_upper, to_avg = 'yea
r'), 'g-')
axes.plot(df_rice_upper['year'].unique(), avg(df_rice_upper, to_avg = 'year'),
'purple')

rice_patch = mpatches.Patch(color = 'grey', label = 'RICE')
harvey_patch = mpatches.Patch(color = 'purple', label  = 'HARVEY')

plt.legend(handles=[rice_patch, harvey_patch])

axes.set_title('Average Number of Students Enrolled to Year in private schools
in Upper Division Computer Science')
axes.set_xlabel('Quarter')
axes.set_ylabel('Average Number of Students Enrolled')
```

```
Out[57]: Text(0, 0.5, 'Average Number of Students Enrolled')
```



```
In [ ]:
```