Comprehensive Study on Performance of Students due to Alcohol Consumption

About

Abstract:

The data were obtained in a survey of students maths language course in secondary school. It contains a lot of interesting social, gender and study information about students. We have use it for some EDA-Explorative Data Analysis.

Children Attraction to Alcohol

1. Puberty time
2. Copy older habits
3. Like to be other Friends
4. Problem with family, school & friends
5. Alcohol and drinking all around them
6. To prove they are grown up

We found this dataset very fascinating as it contains many columns about students which say many things about their grades, performance, family and other factors related to thier personal life. We could find multiple stories from this dataset but as of now, We only want to focus over the performance of students. For this purpose, We found multiple parameters which I can use and they affect the performance of students for example address / alcohol consumption / Pstatus / Medu / Fedu / Mjob / Fjob / famsup / schoolsup.

Findings

We want to understand how these parameters are impacting performance of students and how students are progressing while dealing with these parameters in their life.

Proceedings

In terms of proceedings, First I am dividing this datasets based on the address field and getting data for 'rural' and 'urban' areas students.

After getting these two parts, I have started over the analysis with the help of 'Dalc' field which says how much amount of alcohol consumption is been regualrly taken by the students on the daily basis. I am trying to find out the impact of this consumption over their performance in exams.

Secondly, I find 'Pstatus' field is very important to analysis the same. This field tells that for how many students parents are seperated or living together. I thought this field can be very useful to analyse their

performance as it is a very emotional / mind / time consuming matter for all the generation nowadays.

After these two fields, when I looked into in this datasets, then I reliaze that apart from the region, the education of parents and family support can also be good factors for affecting the performance of students. So I have also used these fields to get insights more about performance.

At the end, I have used school support and higher education field to get insight about how school helps students and what students think about their future as they are the one who will decide the future of nation.

Python Library

The Pandas Library

pandas is everyone's favorite data analyis library providing fast, flexible, and expressive data structures designed to work with relational or table-like data (SQL table or Excel spreadsheet). It is a fundamental high-level building block for doing practical, real world data analysis in Python.

pandas is well suited for:

```
- Tabular data with heterogeneously-typed columns, as in an SQL table
 or Excel spreadsheet

- Ordered and unordered (not necessarily fixed-frequency) time series
 data.

- Arbitrary matrix data (homogeneously typed or heterogeneous) with ro
w and column labels

- Any other form of observational / statistical data sets.
  The data actually need not be labeled at all to be placed into a pan
das data structure
```

The NumPy Library

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.

Using NumPy, a developer can perform the following operations −

```
- Mathematical and logical operations on arrays.

- Fourier transforms and routines for shape manipulation.

- Operations related to linear algebra. NumPy has in-built functions f
or linear algebra and random number generation.
```

The Seaborn Library

Seaborn aims to make visualization a central part of exploring and understanding data. Its dataset-oriented plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

Here is some of the functionality that seaborn offers:

- A dataset-oriented API for examining relationships between multiple variables
- Specialized support for using categorical variables to show observations or aggregate statistics
- Options for visualizing univariate or bivariate distributions and for comparing them between subsets of data
- Automatic estimation and plotting of linear regression models for different kinds dependent variables
- Convenient views onto the overall structure of complex datasets
- High-level abstractions for structuring multi-plot grids that let you easily build complex visualizations
- Concise control over matplotlib figure styling with several built-in themes
- Tools for choosing color palettes that faithfully reveal patterns in your data

Importing Dataset

Dataset :: Student Alcohol Consumption you can download the dataset from link as follows::

```
File Url ="https://www.kaggle.com/uciml/student-alcohol-consumption"
```

Dataset consist of 3 parts::

1. Index
2. Column Names
3. Values(Data)

Exploring DataSet

Database Description

Contents : This datasets has many columns like school, sex, age, address, famsize, Pstatus, Medu, Fedu, Mjob, Fjob, reason, guardian, traveltime, studytime, failures, schoolsup, famsup, paid, activities nursery, higher, internet, romantic, famrel, freetime, goout, Dalc, Walc, health, absences, G1, G2, G3 (grades).

column name	Description	data type	value
school	student's school	binary	'GP' - Gabriel Pereira or 'MS' - Mousinho da Silveira
sex	student's sex	binary	'F' - female or 'M' - male

age	student's age	numeric	from 15 to 22
address	student's home address type	binary	'U' - urban or 'R' - rural
famsize	family size	binary	'LE3' - less or equal to 3 or 'GT3' - greater than 3
Pstatus	parent's cohabitation status	binary	'T' - living together or 'A' - apart
Medu	mother's education	numeric	0 - none, 1 - primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 – higher education
Fedu	father's education	numeric	0 - none, 1 - primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 higher education
Mjob	mother's job	nominal	'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other'
Fjob	father's job	nominal	'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other'
reason	reason to choose this school	nominal	close to 'home', school 'reputation', 'course' preference or 'other'
guardian	student's guardian	nominal	'mother', 'father' or 'other'
traveltime	home to school travel time	numeric	1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour
studytime	weekly study time	numeric	1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours
failures	number of past class failures	numeric	n if 1<=n<3, else 4
schoolsup	extra educational support	binary	yes or no
famsup	family educational support	binary	yes or no
paid	extra paid classes within the course subject (Math or Portuguese)	binary	yes or no
activities	extra-curricular activities	binary	yes or no
nursery	attended nursery school	binary	yes or no
higher	wants to take higher education	binary	yes or no
internet	Internet access at home	binary	yes or no
romantic	with a romantic relationship	binary	yes or no
famrel	quality of family relationships	numeric	from 1 - very bad to 5 - excellent
freetime	free time after school	numeric	from 1 - very low to 5 - very high
goout	going out with friends	numeric	from 1 - very low to 5 - very high
Dalc	workday alcohol consumption	numeric	from 1 - very low to 5 - very high
Walc	weekend alcohol consumption	numeric	from 1 - very low to 5 - very high
health	current health status	numeric	from 1 - very bad to 5 - very good
absences	number of school absences	numeric	from 0 to 93

G1	first period grade	numeric	from 0 to 20
G2	second period grade	numeric	from 0 to 20
G3	final grade	numeric	from 0 to 20, output target

Importing Libraries

In [1]:
```
import pandas as pd
import seaborn as sea
import numpy as np
import pandas_profiling
import seaborn as sns
from sklearn import preprocessing
import matplotlib.pyplot as plt
import os
import sys


%matplotlib inline
%pylab inline

print('Python : {}'.format(sys.version[0:5]))
print('Numpy : {}'.format(np.__version__))
print('Pandas : {}'.format(pd.__version__))
print('Matplotlib : {}'.format(matplotlib.__version__))
print('Seaborns : {}'.format(sns.__version__))
```

```
Populating the interactive namespace from numpy and matplotlib
Python : 3.8.1
Numpy : 1.19.0
Pandas : 1.0.5
Matplotlib : 3.2.2
Seaborns : 0.10.1
```

Importing CSV file

In [2]:
```
data =pd.read_csv("student-mat.csv")
```

Defining Columns in the DataSet.

In [3]:
```
data.columns
```

Out[3]:
```
Index(['school', 'sex', 'age', 'address', 'famsize', 'Pstatus', 'Medu',
'Fedu',
       'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime',
       'failures', 'schoolsup', 'famsup', 'paid', 'activities', 'nurser
y',
       'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout',
'Dalc',
       'Walc', 'health', 'absences', 'G1', 'G2', 'G3'],
      dtype='object')
```

Defining values in the DataSet.

In [4]:
```
data.values
```

Out[4]:
```
array([['GP', 'F', 18, ..., 5, 6, 6],

       ['GP', 'F', 17, ..., 5, 5, 6],
       ['GP', 'F', 15, ..., 7, 8, 10],
       ...,
       ['MS', 'M', 21, ..., 10, 8, 7],
       ['MS', 'M', 18, ..., 11, 12, 10],
       ['MS', 'M', 19, ..., 8, 9, 9]], dtype=object)
```

In [5]:
```
data.head()
```

Out[5]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	fr
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	2
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	3

5 rows × 33 columns

In [6]:
```
data.tail()
```

Out[6]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel
390	MS	M	20	U	LE3	A	2	2	services	services	...	5
391	MS	M	17	U	LE3	T	3	1	services	services	...	2
392	MS	M	21	R	GT3	T	1	1	other	other	...	5
393	MS	M	18	R	LE3	T	3	2	services	other	...	4
394	MS	M	19	U	LE3	T	1	1	other	at_home	...	3

5 rows × 33 columns

Description of Data Fields

In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 395 entries, 0 to 394
Data columns (total 33 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   school      395 non-null    object
 1   sex         395 non-null    object
 2   age         395 non-null    int64
 3   address     395 non-null    object
 4   famsize     395 non-null    object
 5   Pstatus     395 non-null    object
 6   Medu        395 non-null    int64
 7   Fedu        395 non-null    int64
 8   Mjob        395 non-null    object
 9   Fjob        395 non-null    object
 10  reason      395 non-null    object
```

```
 11  guardian    395 non-null    object

 12  traveltime  395 non-null    int64
 13  studytime   395 non-null    int64
 14  failures    395 non-null    int64
 15  schoolsup   395 non-null    object
 16  famsup      395 non-null    object
 17  paid        395 non-null    object
 18  activities  395 non-null    object
 19  nursery     395 non-null    object
 20  higher      395 non-null    object
 21  internet    395 non-null    object
 22  romantic    395 non-null    object
 23  famrel      395 non-null    int64
 24  freetime    395 non-null    int64
 25  goout       395 non-null    int64
 26  Dalc        395 non-null    int64
 27  Walc        395 non-null    int64
 28  health      395 non-null    int64
 29  absences    395 non-null    int64
 30  G1          395 non-null    int64
 31  G2          395 non-null    int64
 32  G3          395 non-null    int64
dtypes: int64(16), object(17)
memory usage: 75.7+ KB
```

In [8]:
```
# T means Transpose
data.describe().T
```

Out[8]:

	count	mean	std	min	25%	50%	75%	max
age	395.0	16.696203	1.276043	15.0	16.0	17.0	18.0	22.0
Medu	395.0	2.749367	1.094735	0.0	2.0	3.0	4.0	4.0
Fedu	395.0	2.521519	1.088201	0.0	2.0	2.0	3.0	4.0
traveltime	395.0	1.448101	0.697505	1.0	1.0	1.0	2.0	4.0
studytime	395.0	2.035443	0.839240	1.0	1.0	2.0	2.0	4.0
failures	395.0	0.334177	0.743651	0.0	0.0	0.0	0.0	3.0
famrel	395.0	3.944304	0.896659	1.0	4.0	4.0	5.0	5.0
freetime	395.0	3.235443	0.998862	1.0	3.0	3.0	4.0	5.0
goout	395.0	3.108861	1.113278	1.0	2.0	3.0	4.0	5.0
Dalc	395.0	1.481013	0.890741	1.0	1.0	1.0	2.0	5.0
Walc	395.0	2.291139	1.287897	1.0	1.0	2.0	3.0	5.0
health	395.0	3.554430	1.390303	1.0	3.0	4.0	5.0	5.0
absences	395.0	5.708861	8.003096	0.0	0.0	4.0	8.0	75.0
G1	395.0	10.908861	3.319195	3.0	8.0	11.0	13.0	19.0
G2	395.0	10.713924	3.761505	0.0	9.0	11.0	13.0	19.0
G3	395.0	10.415190	4.581443	0.0	8.0	11.0	14.0	20.0

Checking For Null Values

In [9]:
```
np.count_nonzero(data.isnull())
```

Out[9]:
```
0
```

In [10]:
```
rural_data = data[data['address'] == 'R']
urban_data = data[data['address'] == 'U']
```

Categorical features

In [11]:
```
categorical_features = (data.select_dtypes(include=['object']).columns.value
s)
categorical_features
```

Out[11]:
```
array(['school', 'sex', 'address', 'famsize', 'Pstatus', 'Mjob', 'Fjob',
       'reason', 'guardian', 'schoolsup', 'famsup', 'paid', 'activities',
       'nursery', 'higher', 'internet', 'romantic'], dtype=object)
```

Numerical Features

In [12]:
```
numerical_features = data.select_dtypes(include = ['float64', 'int64']).colu
mns.values
numerical_features
```

Out[12]:
```
array(['age', 'Medu', 'Fedu', 'traveltime', 'studytime', 'failures',
       'famrel', 'freetime', 'goout', 'Dalc', 'Walc', 'health',
       'absences', 'G1', 'G2', 'G3'], dtype=object)
```

In [13]:
```
pandas_profiling.ProfileReport(data)
```

Pandas Profiling Report

Overview

Overview | Reproduction | Warnings 5

Dataset statistics

Number of variables	33
Number of observations	395
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	75.7 KiB
Average record size in memory	196.2 B

Variable types

NUM	13
CAT	12
BOOL	8

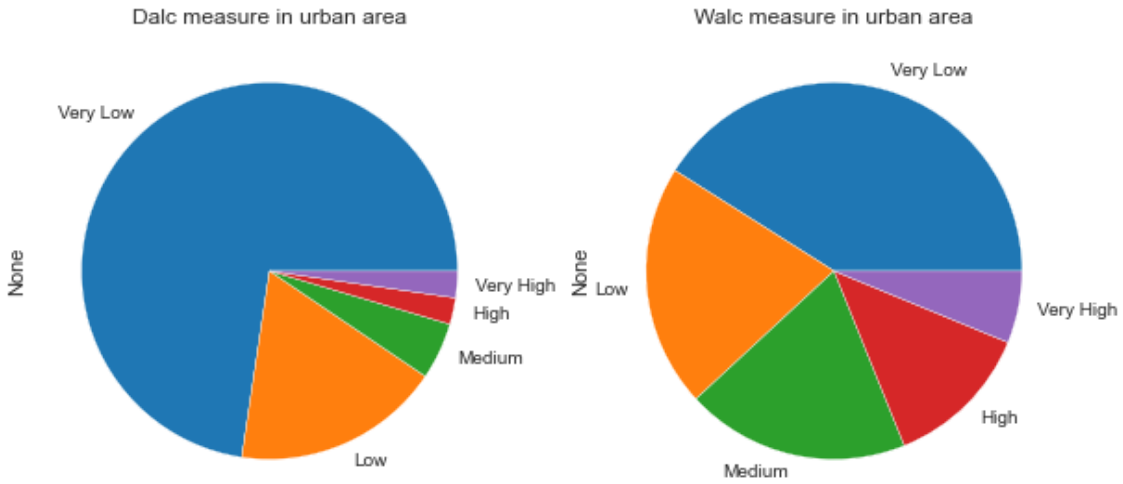
Variables

Out[13]:

In [14]:
```
fig, ax = plt.subplots(1, 2)
urban_data.groupby(['Dalc']).size().plot(kind='pie',figsize=(10,5), ax=ax[0
], title='Dalc measure in urban area',labels=('Very Low','Low','Medium','Hig
h','Very High'))
urban_data.groupby(['Walc']).size().plot(kind='pie',figsize=(10,5), ax=ax[1
```

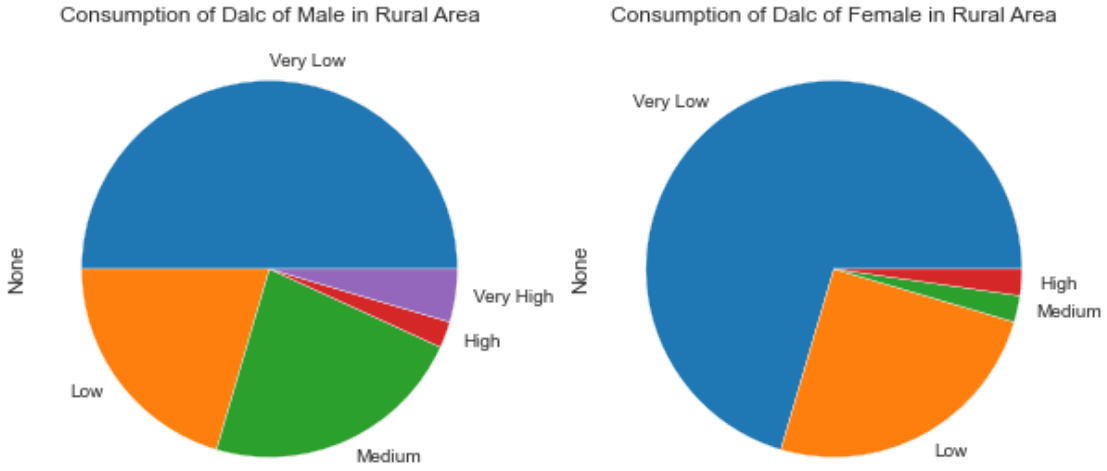
```
], title='Walc measure in urban area',labels=('Very Low','Low','Medium','Hig
h','Very High'))
```

Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x25277670>



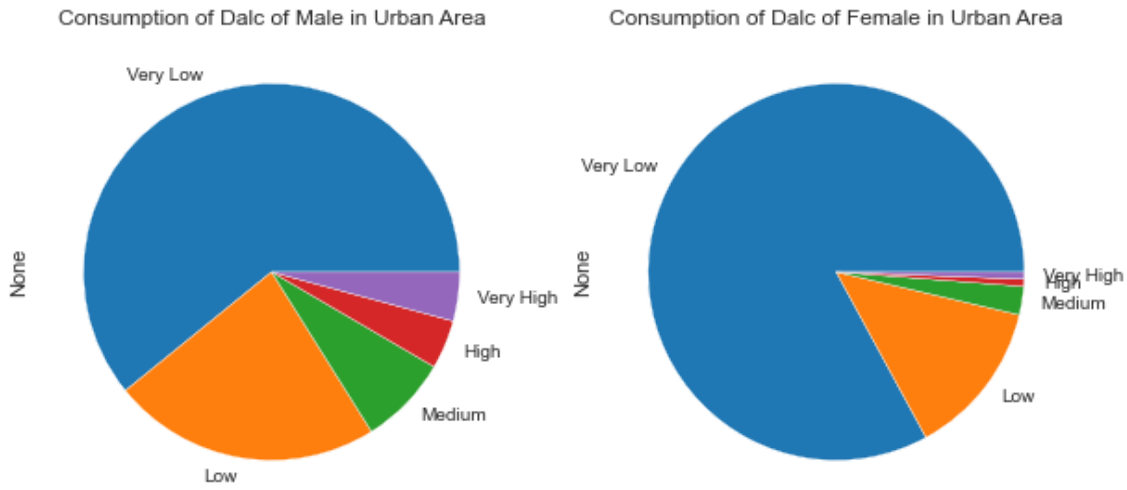
In [15]:
```
fig, ax = plt.subplots(1, 2)

rural_df = rural_data[['sex', 'Dalc']]
rural_df = rural_df[rural_df['sex'] == 'M']
rural_df.groupby(['sex', 'Dalc']).size().plot(kind='pie', ax=ax[0],figsize=(
10,8),labels=('Very Low','Low','Medium','High','Very High'), title="Consumpt
ion of Dalc of Male in Rural Area")

rural_df = rural_data[['sex', 'Dalc']]
rural_df = rural_df[rural_df['sex'] == 'F']
rural_df.groupby(['sex', 'Dalc']).size().plot(kind='pie', ax=ax[1], figsize=
(10,8),labels=('Very Low','Low','Medium','High','Very High'), title="Consump
tion of Dalc of Female in Rural Area")
```

Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x254494c0>



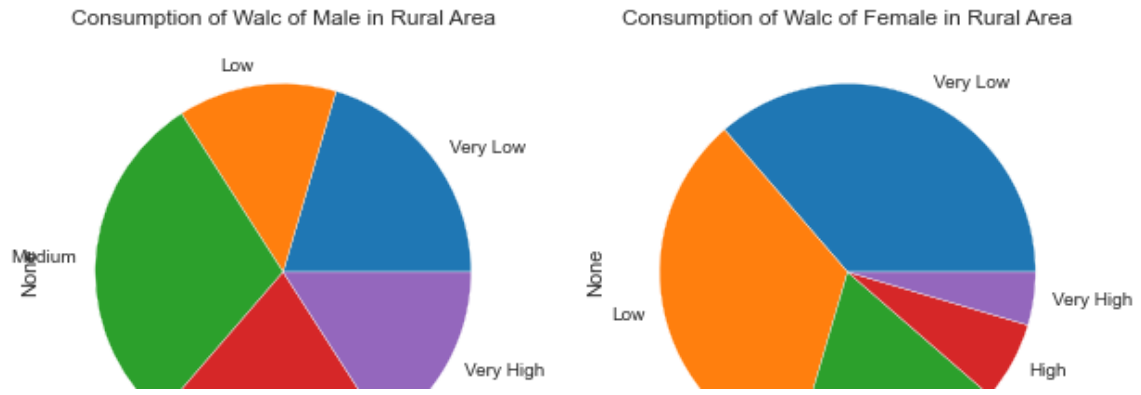
In [16]:
```
fig, ax = plt.subplots(1, 2)

urban_df = urban_data[['sex', 'Dalc']]
urban_df = urban_df[urban_df['sex'] == 'M']
urban_df.groupby(['sex', 'Dalc']).size().plot(kind='pie',figsize=(10,8),labe
ls=('Very Low','Low','Medium','High','Very High'), ax=ax[0], title="Consumpt
ion of Dalc of Male in Urban Area")

urban_df = urban_data[['sex', 'Dalc']]
urban_df = urban_df[urban_df['sex'] == 'F']
urban_df.groupby(['sex', 'Dalc']).size().plot(kind='pie',figsize=(10,8),labe
ls=('Very Low','Low','Medium','High','Very High'), ax=ax[1], title="Consumpt
ion of Dalc of Female in Urban Area")
```

Out[16]: <matplotlib.axes._subplots.AxesSubplot at 0x25d408b0>



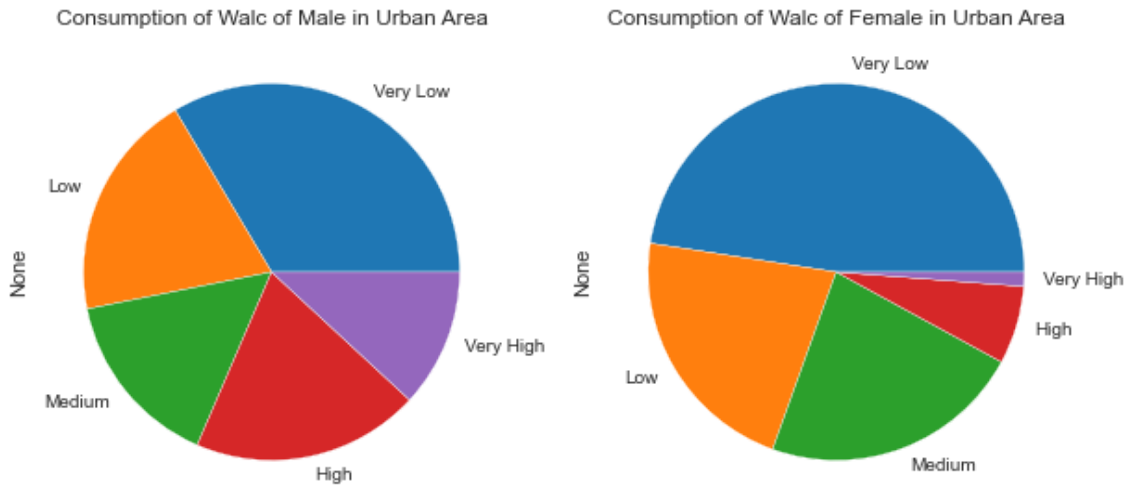
In [17]:
```
fig, ax = plt.subplots(1, 2)

rural_df = rural_data[['sex', 'Walc']]
rural_df = rural_df[rural_df['sex'] == 'M']
rural_df.groupby(['sex', 'Walc']).size().plot(kind='pie', ax=ax[0],figsize=(
10,8),labels=('Very Low','Low','Medium','High','Very High'), title="Consumpt
ion of Walc of Male in Rural Area")

rural_df = rural_data[['sex', 'Walc']]
rural_df = rural_df[rural_df['sex'] == 'F']
rural_df.groupby(['sex', 'Walc']).size().plot(kind='pie', ax=ax[1], figsize=
(10,8),labels=('Very Low','Low','Medium','High','Very High'), title="Consump
tion of Walc of Female in Rural Area")
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0x25245568>



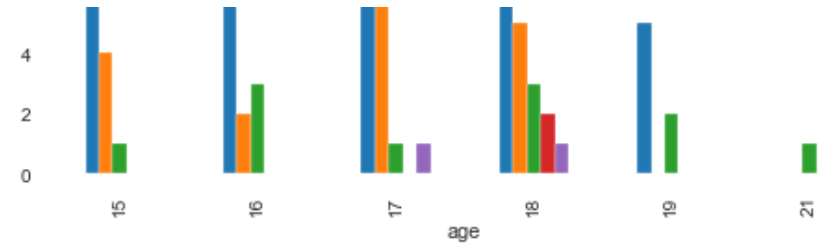
In [18]:
```
fig, ax = plt.subplots(1, 2)

urban_df = urban_data[['sex', 'Walc']]
urban_df = urban_df[urban_df['sex'] == 'M']
urban_df.groupby(['sex', 'Walc']).size().plot(kind='pie',figsize=(10,8),labe
ls=('Very Low','Low','Medium','High','Very High'), ax=ax[0], title="Consumpt
ion of Walc of Male in Urban Area")

urban_df = urban_data[['sex', 'Walc']]
urban_df = urban_df[urban_df['sex'] == 'F']
urban_df.groupby(['sex', 'Walc']).size().plot(kind='pie',figsize=(10,8),labe
ls=('Very Low','Low','Medium','High','Very High'), ax=ax[1], title="Consumpt
ion of Walc of Female in Urban Area")
```

Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0x25369298>



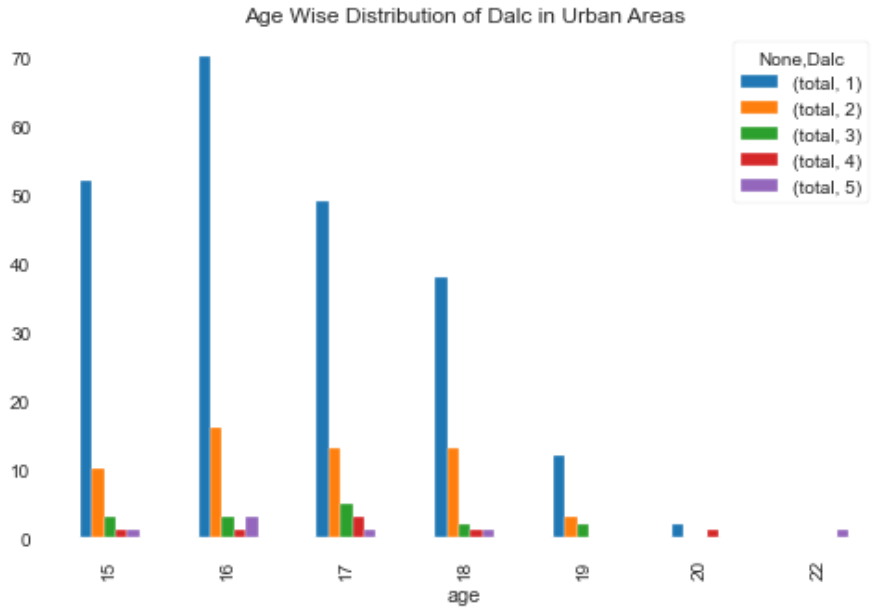
In [19]:
```
# age wise distribution of Daily consumption in rural areas
# rural_df.head()
rural_df = rural_data[['age', 'Dalc']].copy()
rural_df['total'] = 1
rural_df = rural_df.groupby(['age', 'Dalc']).sum().reset_index()
pd.pivot_table(rural_df, index=['age'], columns=['Dalc'], values=['total']).
fillna(0).plot(kind='bar',figsize=(8,5), stacked=False,

title="Age Wise Distribution of Dalc in Rural Areas")
```

Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x253808e0>



In [20]:
```
urban_df = urban_data[['age', 'Dalc']].copy()
urban_df['total'] = 1
urban_df = urban_df.groupby(['age', 'Dalc']).sum().reset_index()
pd.pivot_table(urban_df, index=['age'], columns=['Dalc'], values=['total']).
fillna(0).plot(kind='bar',figsize=(8,5), stacked=False,

title="Age Wise Distribution of Dalc in Urban Areas")
```

Out[20]: <matplotlib.axes._subplots.AxesSubplot at 0x25934f58>

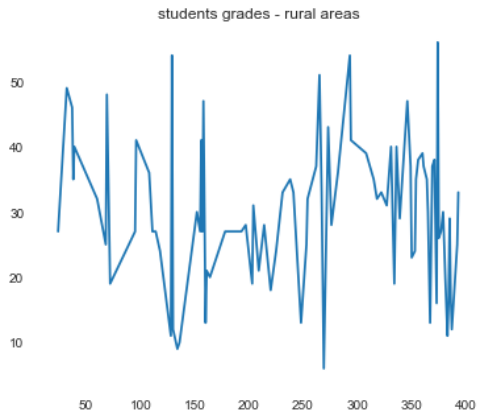


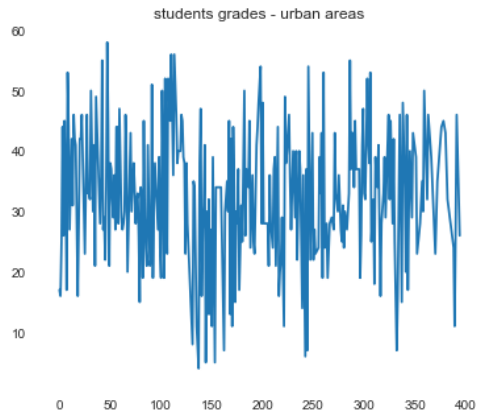
Performance of Students in Rural & Urban Areas

In [21]:
```
# total grades in rural and urban areas

fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

rural_df = rural_data[['G1', 'G2', 'G3']].copy()
rural_df['total'] = rural_df.apply(sum, axis=1)
rural_df['total'].plot(ax=ax[0],figsize=(10,5), title="students grades - rur
al areas")

urban_df = urban_data[['G1', 'G2', 'G3']].copy()
urban_df['total'] = urban_df.apply(sum, axis=1)
urban_df['total'].plot(ax=ax[1],figsize=(15,5), title="students grades - urb
an areas")

rural_df.mean()
urban_df.mean()
```

Out[21]:
```
G1       11.032573
G2       10.967427
```

```
G3       10.674267
total    32.674267
dtype: float64
```





In [22]:
```
print("Performance of Students in Rural Areas",rural_df['total'].mean())
print("Performance of Students in Urban Areas",urban_df['total'].mean())
```

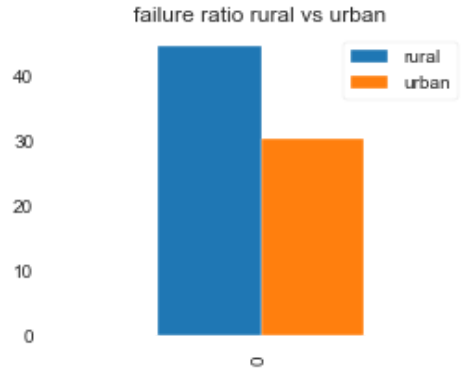
```
Performance of Students in Rural Areas 29.818181818181817
Performance of Students in Urban Areas 32.6742671009772
```

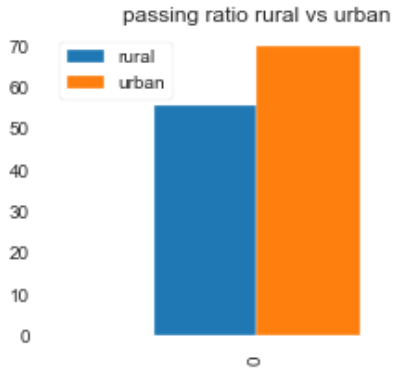
This analysis shows that how is the total grades are going for rural / urban areas students. First thing which one can see that urban area chart is very dense and ofcourse it is because we have more data for urban areas students than rural areas. Another thing, which one see that on an average urban areas students are more productive in terms of performance than rural areas students as average grade for rural areas students is 29.81 whereas for urban areas students it is 32.67.

Past Failures and Passing Ratios in Rural and Urban Areas

In [23]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

rural_failure_ratio = (1.0 * rural_data['failures'].sum()) / len(rural_data)
* 100

urban_failure_ratio = (1.0 * urban_data['failures'].sum()) / len(urban_data)
* 100

failure_df = pd.DataFrame({'rural' : [rural_failure_ratio,], 'urban' : [urba
n_failure_ratio,]})

rural_passing_ratio = (1 - (1.0 * rural_data['failures'].sum()) / len(rural_
data)) * 100

urban_passing_ratio = (1 - (1.0 * urban_data['failures'].sum()) / len(urban_
data)) * 100

passing_df = pd.DataFrame({'rural' : [rural_passing_ratio,], 'urban' : [urba
n_passing_ratio,]})

failure_df.plot(kind='bar', legend=True, ax=ax[0],figsize=(10,3), title="fai
lure ratio rural vs urban")

passing_df.plot(kind='bar', ax=ax[1], legend=True,figsize=(10,3), title="pas
sing ratio rural vs urban")
```

Out[23]: <matplotlib.axes._subplots.AxesSubplot at 0x25a3f460>





In [24]:
```
print("Failure Ratio of Rural & Urban Area \n\n Rural Area - ",rural_failure
_ratio,"\n Urban Area - ",urban_failure_ratio)
print("\n\nPassing Ratio of Rural & Urban Area \n\n Rural Area - ",rural_pas
sing_ratio,"\n Urban Area - ",urban_passing_ratio)
```

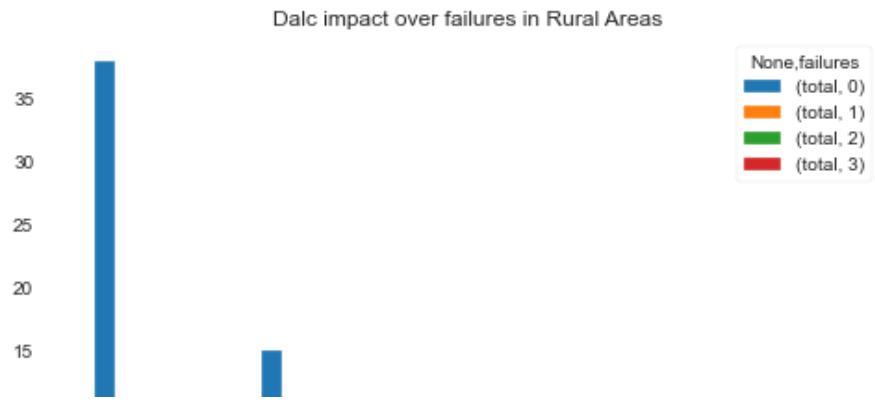
```
Failure Ratio of Rural & Urban Area

 Rural Area -  44.31818181818182
 Urban Area -  30.293159609120522


Passing Ratio of Rural & Urban Area

 Rural Area -  55.68181818181819
 Urban Area -  69.70684039087948
```

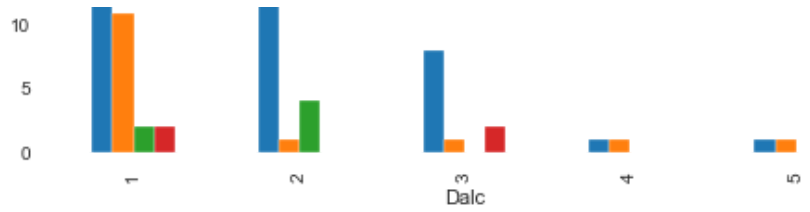
This analysis shows that what is the past passing & failure ratio of students in rural / urban areas. Thing, which one see that on an average urban areas students are more productive in terms of performance than rural areas students. As the Passing Ratio in Urban Area is more than Rural Area and Failure Ratio is more in Rural Area than Urban Area.

Impact of Dalc over Failures in Rural Areas

In [25]:
```
rural_df = rural_data[['failures', 'Dalc']].copy()
rural_df['total'] = 1
rural_df.head()
rural_df = rural_df.groupby(['Dalc', 'failures']).sum().reset_index()
pd.pivot_table(rural_df, index=['Dalc'], columns=['failures'], values=['tota
l']).fillna(0).plot(kind='bar',
                                                    stacked=False, figsize=(8,5), ti
tle="Dalc impact over failures in Rural Areas")
```

Out[25]: <matplotlib.axes._subplots.AxesSubplot at 0x25a768b0>

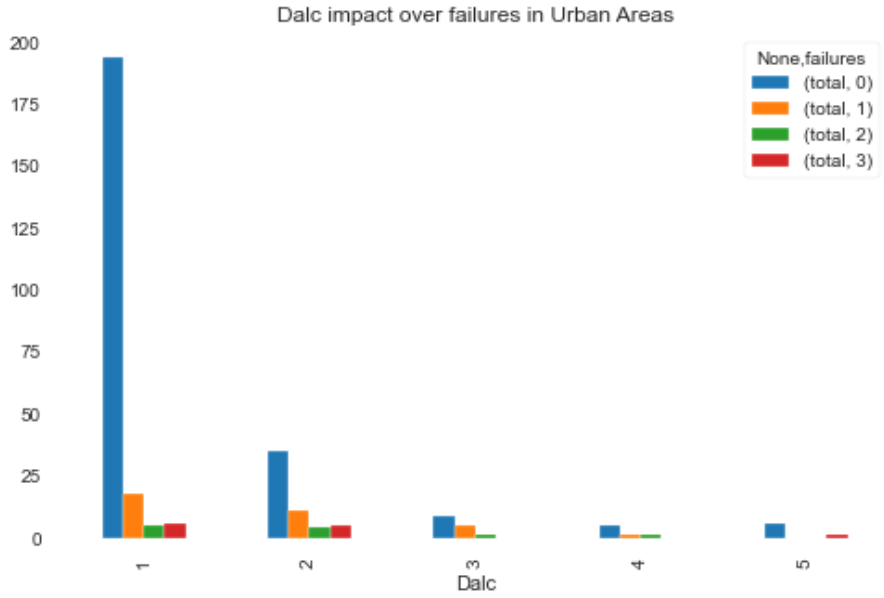




Impact of Dalc over Failures in Urban Areas

In [26]:
```
urban_df = urban_data[['failures', 'Dalc']].copy()
urban_df['total'] = 1
urban_df.head()
urban_df = urban_df.groupby(['Dalc', 'failures']).sum().reset_index()
pd.pivot_table(urban_df, index=['Dalc'], columns=['failures'], values=['tota
l']).fillna(0).plot(kind='bar',
                                                    stacked=False, figsize=(8,5), ti
tle="Dalc impact over failures in Urban Areas")
```

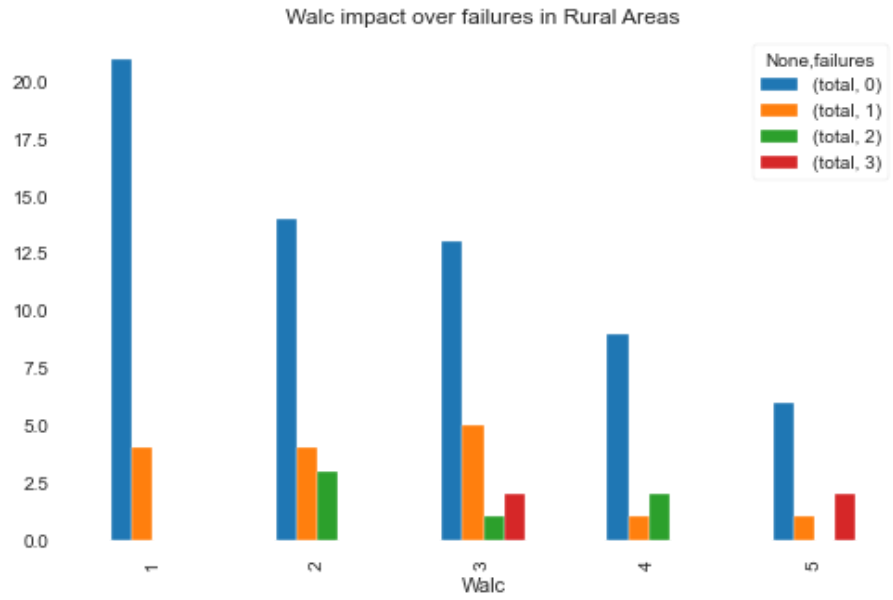
Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x2606b910>



Impact of Walc over Failures in Rural Areas

In [27]:
```
rural_df = rural_data[['failures', 'Walc']].copy()
rural_df['total'] = 1
rural_df.head()
rural_df = rural_df.groupby(['Walc', 'failures']).sum().reset_index()
pd.pivot_table(rural_df, index=['Walc'], columns=['failures'], values=['tota
l']).fillna(0).plot(kind='bar',
                                                    stacked=False, figsize=(8,5), ti
tle="Walc impact over failures in Rural Areas")
```

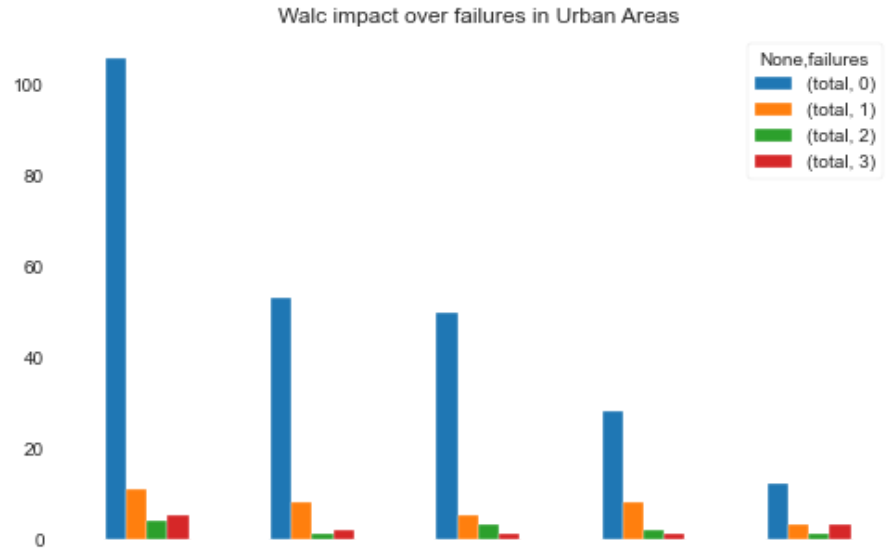
Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x2776fcd0>



Impact of Walc over Failures in Urban Areas

In [28]:
```
urban_df = urban_data[['failures', 'Walc']].copy()
urban_df['total'] = 1
urban_df.head()
urban_df = urban_df.groupby(['Walc', 'failures']).sum().reset_index()
pd.pivot_table(urban_df, index=['Walc'], columns=['failures'], values=['tota
l']).fillna(0).plot(kind='bar',
                                                    stacked=False, figsize=(8,5), ti
tle="Walc impact over failures in Urban Areas")
```

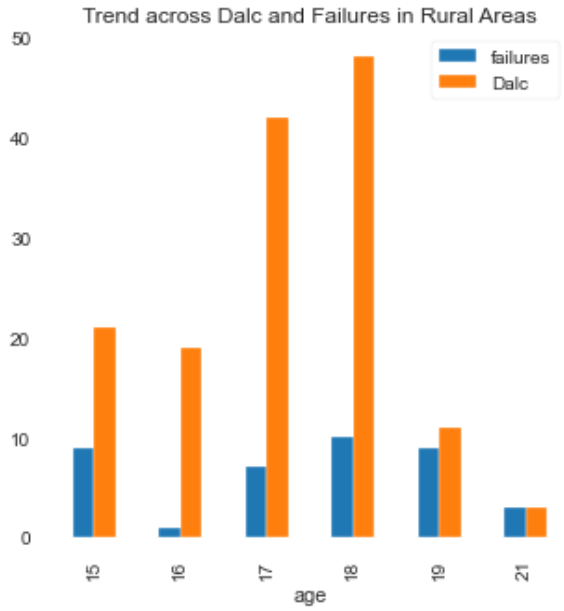
Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x2607db50>



Trend across Dalc and failures in Rural Areas

In [29]:
```
rural_df = rural_data[['failures', 'age', 'Dalc']]

rural_df.groupby(['age']).sum().plot(kind='bar',figsize=(5,5),title="Trend across Dalc and Failures in Rural Areas")
```

Out[29]: <matplotlib.axes._subplots.AxesSubplot at 0x2780ee68>

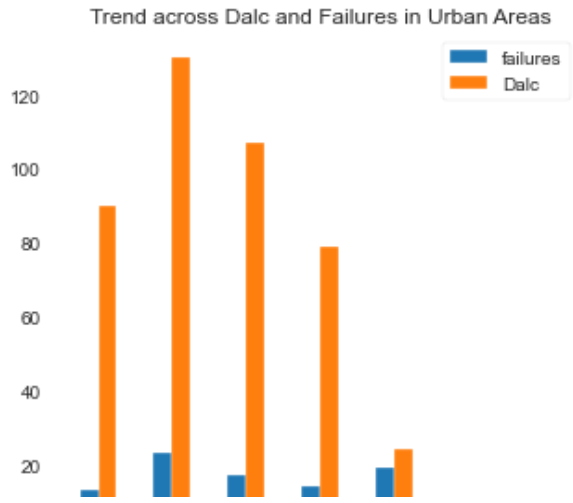


This analysis shows that Trend across Dalc and Failures in Rural Areas. Thing, which one see that on age of 18 Maximum Daily Alcohol Consumption and Failure is Maximum

Trend across Dalc and failures in Urban Areas

In [30]:
```
urban_df = urban_data[['failures', 'age', 'Dalc']]

urban_df.groupby(['age']).sum().plot(kind='bar',figsize=(5,5),title="Trend across Dalc and Failures in Urban Areas")
```

Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x2791abc8>

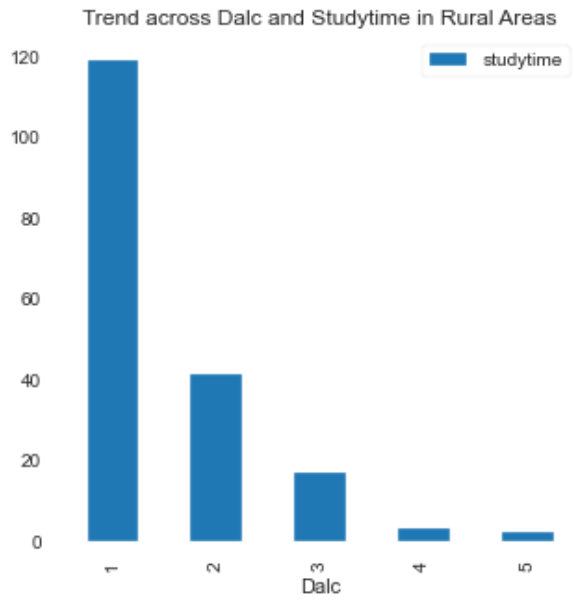


This analysis shows that Trend across Dalc and Failures in Rural Areas. Thing, which one see that on age of 16 Maximum Daily Alcohol Consumption and Failure is Maximum

Trend across Dalc and Studytime in Rural Areas

In [31]:
```
rural_df = rural_data[['studytime', 'Dalc']]
rural_df.groupby(['Dalc']).sum().plot(kind='bar',figsize=(5,5),title="Trend
 across Dalc and Studytime in Rural Areas")
```

Out[31]: <matplotlib.axes._subplots.AxesSubplot at 0x27966eb0>

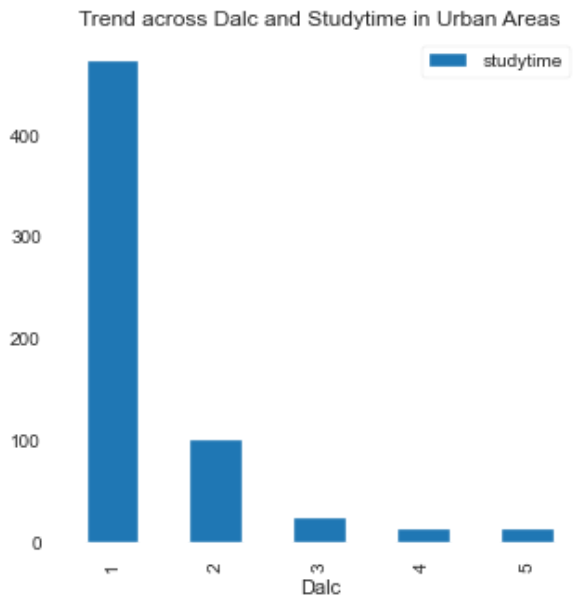


This analysis shows that Trend across Dalc and Studytime in Rural Areas. Thing, which one can see that Students whose daily alcohol consumption is very low has maximum studytime in rural areas.

Trend across Dalc and Studytime in Urban Areas

In [32]:
```
urban_df = urban_data[['studytime', 'Dalc']]
urban_df.groupby(['Dalc']).sum().plot(kind='bar',figsize=(5,5),title="Trend
 across Dalc and Studytime in Urban Areas")
```

Out[32]: <matplotlib.axes._subplots.AxesSubplot at 0x279a0700>

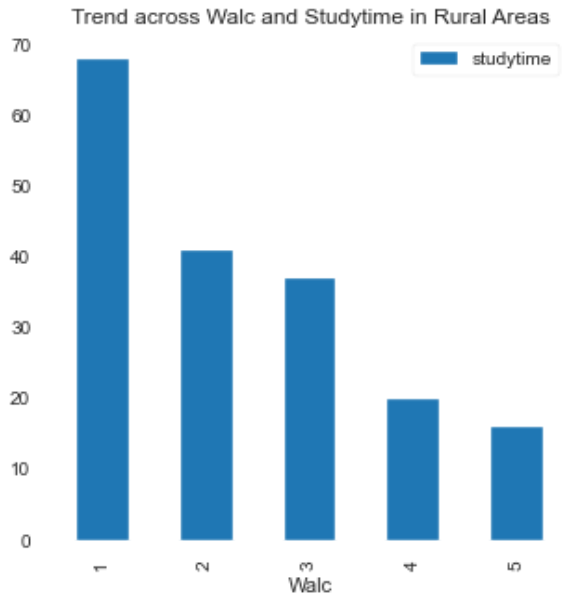


This analysis shows that Trend across Dalc and Studytime in Urban Areas. Thing, which one can see that Students whose daily alcohol consumption is very low has maximum studytime in urban areas

Trend across Walc and Studytime in Rural Areas

In [33]:
```
rural_df = rural_data[['studytime', 'Walc']]
rural_df.groupby(['Walc']).sum().plot(kind='bar',figsize=(5,5),title="Trend
 across Walc and Studytime in Rural Areas")
```

Out[33]: <matplotlib.axes._subplots.AxesSubplot at 0x27bfd580>

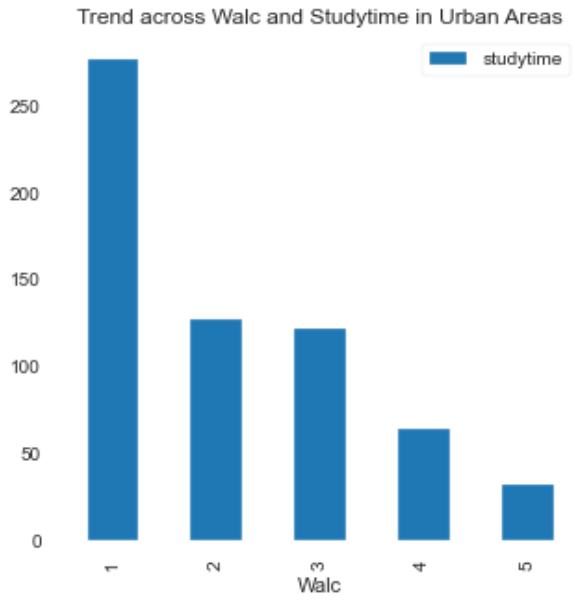


This analysis shows that Trend across Walc and Studytime in Rural Areas. Thing, which one can see that Students whose weekly alcohol consumption is very low has maximum studytime in rural areas

Trend across Walc and Studytime in Urban Areas

In [34]:
```
urban_df = urban_data[['studytime', 'Walc']]
urban_df.groupby(['Walc']).sum().plot(kind='bar',figsize=(5,5),title="Trend
 across Walc and Studytime in Urban Areas")
```

Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x27c36628>

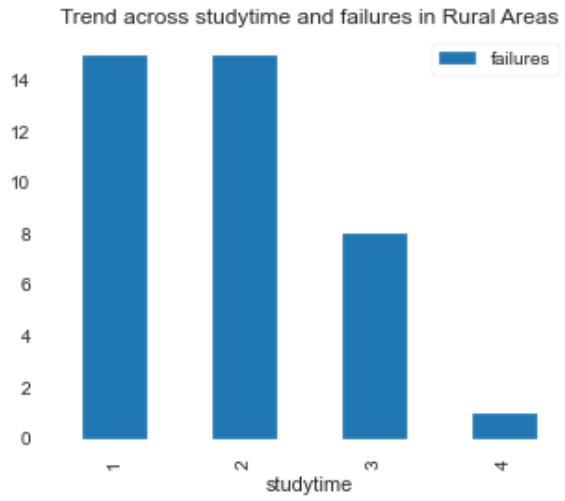


This analysis shows that Trend across Walc and Studytime in Urban Areas. Thing, which one can see that Students whose daily alcohol consumption is very low has maximum studytime in urban areas

Trend across Studytime and Failures in Rural Areas

In [35]:
```
rural_df = rural_data[['studytime', 'failures']]
rural_df.groupby(['studytime']).sum().plot(kind='bar',figsize=(5,4),title="T
rend across studytime and failures in Rural Areas")
```

Out[35]: <matplotlib.axes._subplots.AxesSubplot at 0x27c637c0>

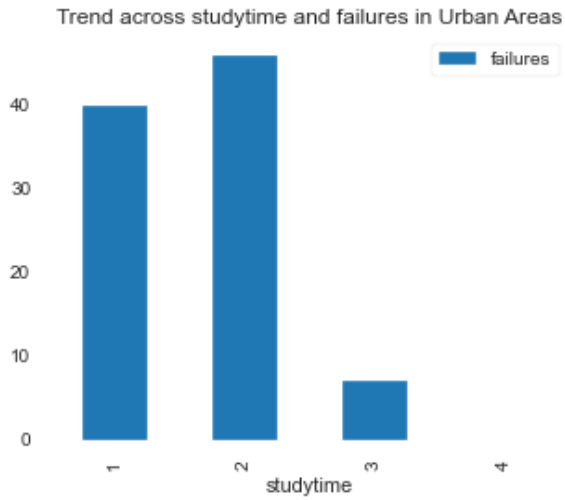


This analysis shows that Trend across Studytime and Failures in Rural Areas. Thing, which one can see that Students whose Studytime is less than 5 hours has maximum failure ratio in rural areas

Trend across Studytime and Failures in Urban Areas

In [36]:
```
urban_df = urban_data[['studytime', 'failures']]
urban_df.groupby(['studytime']).sum().plot(kind='bar',figsize=(5,4),title="T
rend across studytime and failures in Urban Areas")
```

Out[36]: <matplotlib.axes._subplots.AxesSubplot at 0x27c99d78>

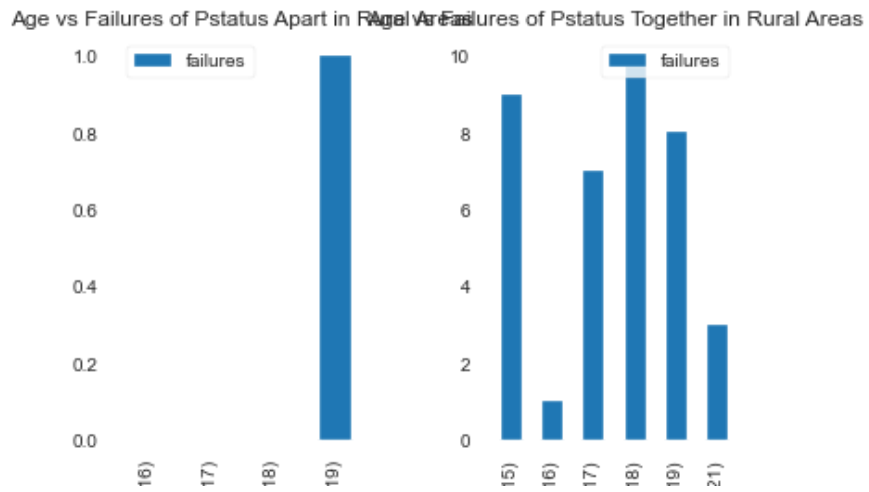


This analysis shows that Trend across Studytime and Failures in Urban Areas. Thing, which one can see that Students whose Studytime is 2 to 5 hours has maximum failure ratio in urban areas.

Trend Age vs Failures of Pstatus Apart & Together in Rural Areas

In [37]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

rural_df = rural_data[['Pstatus', 'failures', 'age']]
rural_df_A = rural_df[rural_df['Pstatus'] == 'A']
rural_df_A.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[0], title
="Age vs Failures of Pstatus Apart in Rural Areas")

rural_df_T = rural_df[rural_df['Pstatus'] == 'T']
rural_df_T.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[1], title
="Age vs Failures of Pstatus Together in Rural Areas")
```

Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x25a05910>

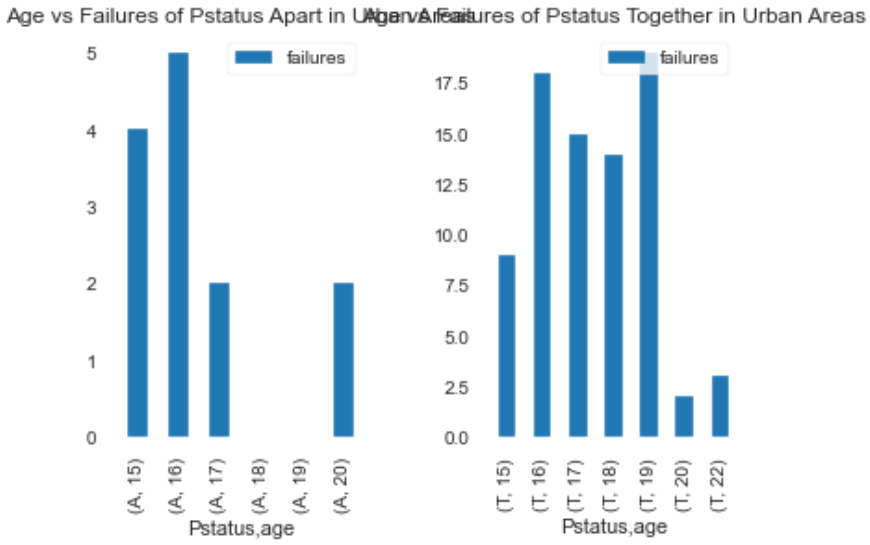


This analysis shows that Age vs Failures of Pstatus Apart in Rural Areas. First Thing, which one can see that Students whose parents are not living together n whose age is 19 has failure ratio high. Second Thing, which one can see that Students whose parents are living together n whose age is 18 has failure ratio high.

Trend Age vs Failures of Pstatus Apart & Together in Urban Areas

In [38]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

urban_df = urban_data[['Pstatus', 'failures', 'age']]
urban_df_A = urban_df[urban_df['Pstatus'] == 'A']
urban_df_A.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[0], title
="Age vs Failures of Pstatus Apart in Urban Areas")

urban_df_T = urban_df[urban_df['Pstatus'] == 'T']
urban_df_T.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[1], title
="Age vs Failures of Pstatus Together in Urban Areas")
```

Out[38]: <matplotlib.axes._subplots.AxesSubplot at 0x250cf958>



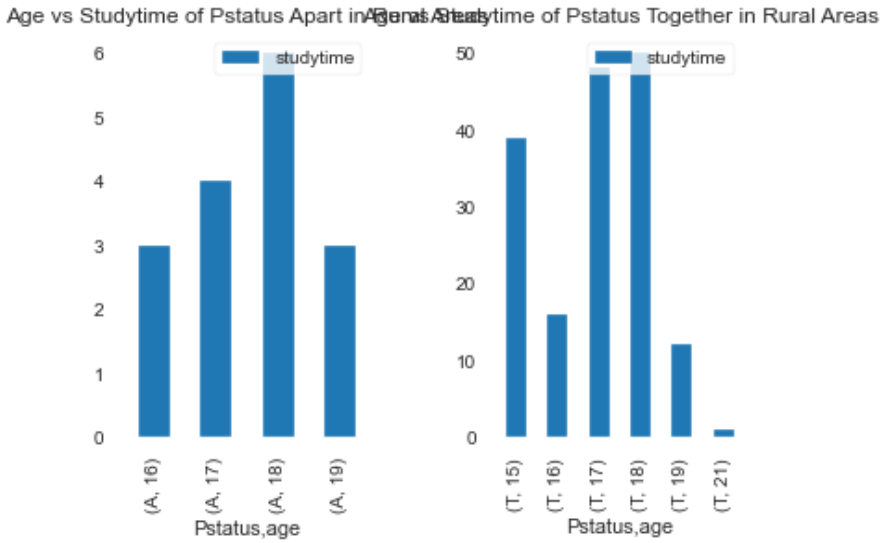
This analysis shows that Age vs Failures of Pstatus Apart in Urban Areas. First Thing, which one can see that Students whose parents are not living together n whose age is 16 has failure ratio high. Second Thing, which one can see that Students whose parents are living together n whose age is 19 has failure ratio high.

Trend Age vs Studytime of Pstatus Apart & Together in Rural Areas

In [39]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

rural_df = rural_data[['Pstatus', 'studytime', 'age']]
rural_df_A = rural_df[rural_df['Pstatus'] == 'A']
rural_df_A.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[0], title
="Age vs Studytime of Pstatus Apart in Rural Areas")
```

```
rural_df_T = rural_df[rural_df['Pstatus'] == 'T']
rural_df_T.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[1], title
="Age vs Studytime of Pstatus Together in Rural Areas")
```

Out[39]: <matplotlib.axes._subplots.AxesSubplot at 0x24f2fef8>

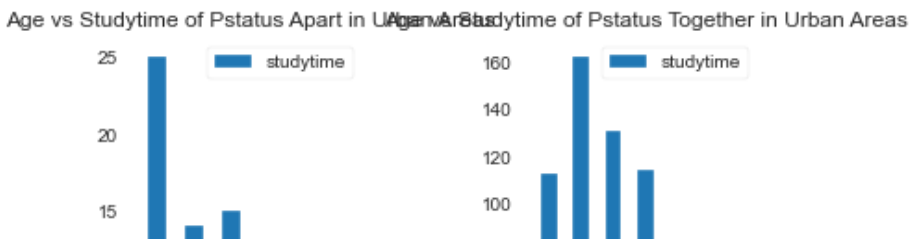


This analysis shows that Age vs Studytime of Pstatus Apart & Together in Rural Areas. First Thing, which one can see that Students whose age is 18 has maximum Studytime whether his\her parents are apart or staying together in rural areas.

Age vs Studytime of Pstatus Apart & Together in Urban Areas

In [40]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.75)

urban_df = urban_data[['Pstatus', 'studytime', 'age']]
urban_df_A = urban_df[urban_df['Pstatus'] == 'A']
urban_df_A.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[0], title
="Age vs Studytime of Pstatus Apart in Urban Areas")

urban_df_T = urban_df[urban_df['Pstatus'] == 'T']
urban_df_T.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[1], title
="Age vs Studytime of Pstatus Together in Urban Areas")
```

Out[40]: <matplotlib.axes._subplots.AxesSubplot at 0x14a83b68>

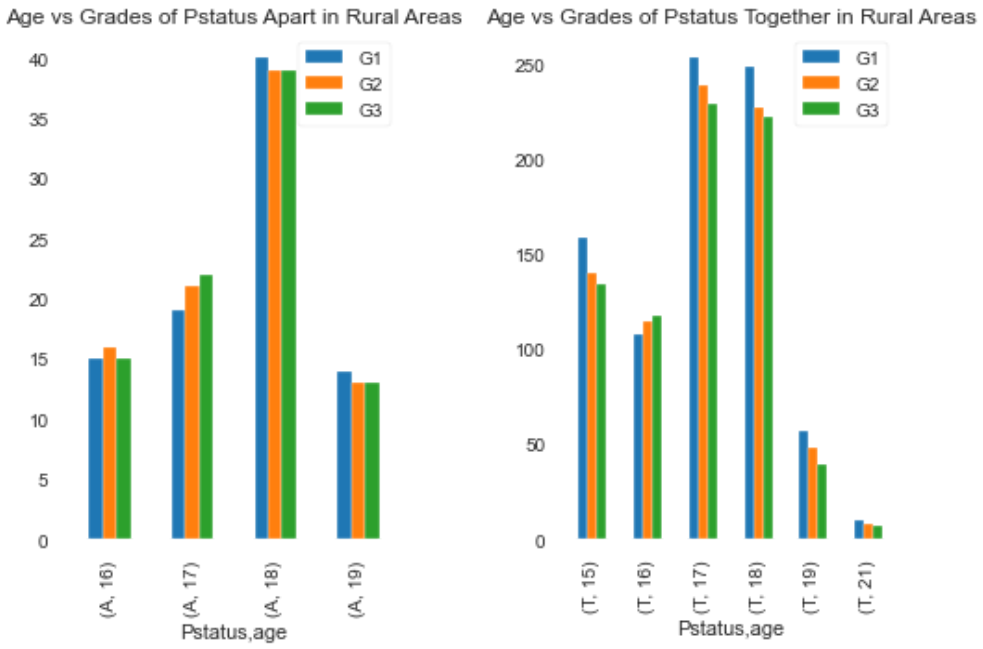


This analysis shows that Age vs Studytime of Pstatus Apart & Together in Urban Areas. First Thing, which one can see that Students whose age is 15 & 16 has maximum Studytime while his\her parents are apart or staying together respectively in urban areas.

Age vs Grades of Pstatus Apart in Rural Areas

In [41]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

rural_df = rural_data[['Pstatus', 'G1', 'G2', 'G3', 'age']]

rural_df_A = rural_df[rural_df['Pstatus'] == 'A']
rural_df_A.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[0],figsiz
e=(8,5), title="Age vs Grades of Pstatus Apart in Rural Areas")

rural_df_T = rural_df[rural_df['Pstatus'] == 'T']
rural_df_T.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[1],figsiz
e=(8,5), title="Age vs Grades of Pstatus Together in Rural Areas")
```

Out[41]: <matplotlib.axes._subplots.AxesSubplot at 0x23f2c178>

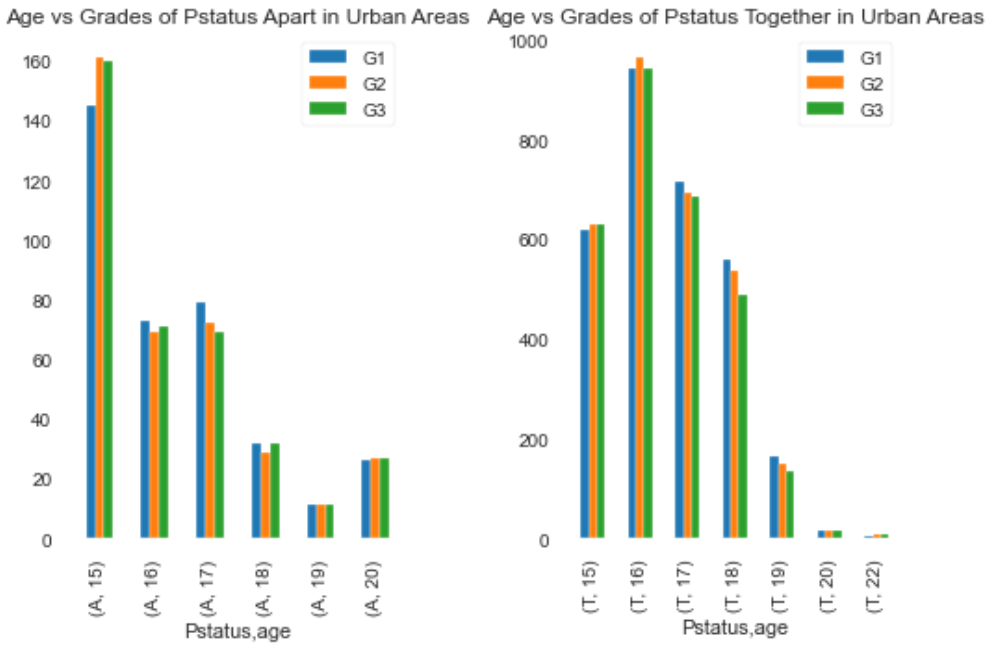


This analysis shows that Age vs Grades of Pstatus Apart & Together in Rural Areas. First Thing, which one can see that Students whose age is 18 has scores Maximum while his\her parents are apart in rural areas. Second Thing, which one can see that Students whose age is 17 & 18 has scores Maximum while his\her parents are together in rural areas.

Age vs Grades of Pstatus Apart & Together in Urban Areas

In [42]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

urban_df = urban_data[['Pstatus', 'G1', 'G2', 'G3', 'age']]

urban_df_A = urban_df[urban_df['Pstatus'] == 'A']
urban_df_A.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[0],figsiz
e=(8,5), title="Age vs Grades of Pstatus Apart in Urban Areas")

urban_df_T = urban_df[urban_df['Pstatus'] == 'T']
urban_df_T.groupby(['Pstatus', 'age']).sum().plot(kind='bar',ax=ax[1],figsiz
e=(8,5), title="Age vs Grades of Pstatus Together in Urban Areas")
```

Out[42]: <matplotlib.axes._subplots.AxesSubplot at 0x23de3520>

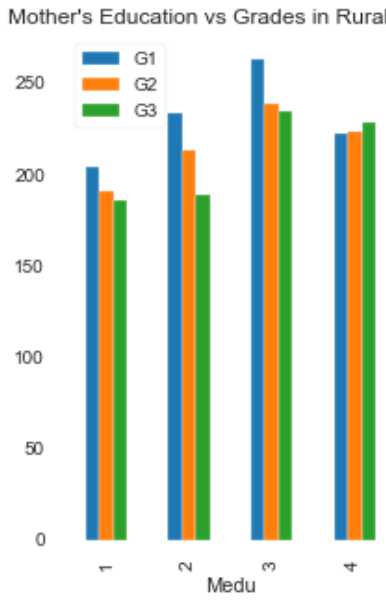


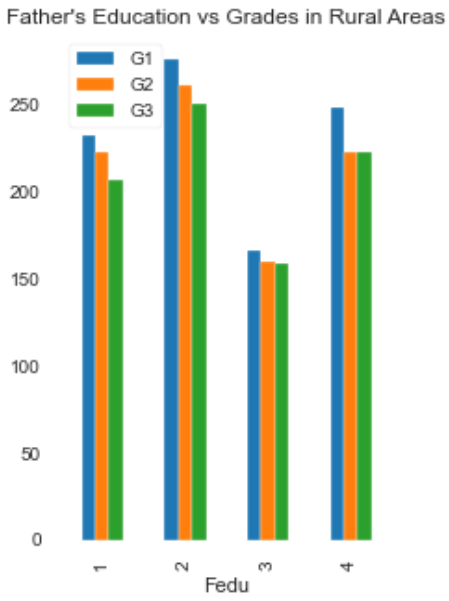
This analysis shows that Age vs Grades of Pstatus Apart & Together in Urban Areas. First Thing, which one can see that Students whose age is 15 has scores Maximum while his\her parents are apart in Urban areas. Second Thing, which one can see that Students whose age is 16 has scores Maximum while his\her parents are together in Urban areas.

Trend Mother's & Father's Education vs Grades in Rural Areas

In [43]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

rural_df = rural_data[['Medu', 'G1', 'G2', 'G3']]
rural_df.groupby(['Medu']).sum().plot(kind='bar',ax=ax[0], legend=True,figsi
ze=(8,5), title="Mother's Education vs Grades in Rural Areas")

rural_df = rural_data[['Fedu', 'G1', 'G2', 'G3']]
rural_df.groupby(['Fedu']).sum().plot(kind='bar',ax=ax[1], legend=True,figsi
ze=(8,5), title="Father's Education vs Grades in Rural Areas")
```

Out[43]: <matplotlib.axes._subplots.AxesSubplot at 0x23c65ce8>



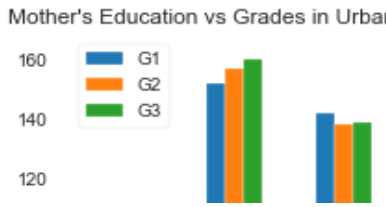


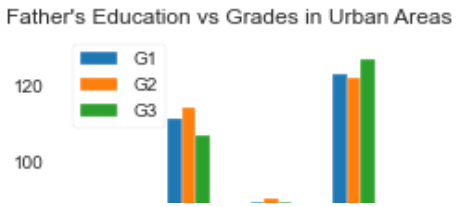
This analysis shows that Trend Mother's & Father's Education vs Grades in Rural Areas. Thing, which one can see that Students whose scores Maximum,mostly his\her parents has education upto 9th only in Rural areas.

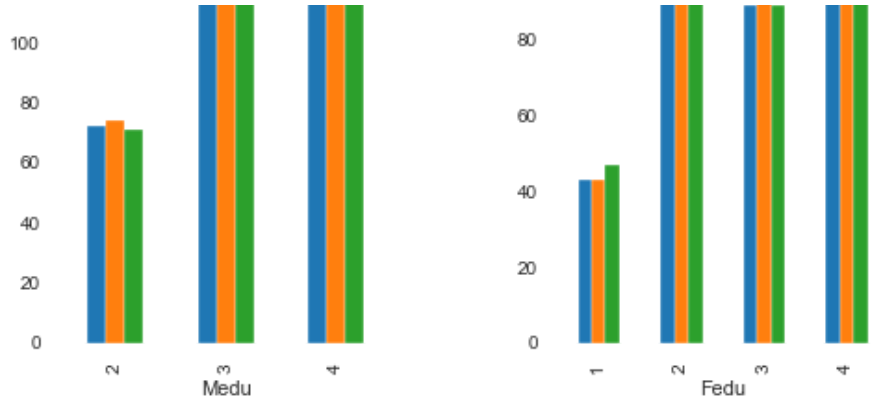
Trend Mother's & Father's Education vs Grades in Urban Areas

In [44]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

urban_df = urban_data[['Pstatus', 'Medu', 'G1', 'G2', 'G3']]
urban_df_A = urban_df[urban_df['Pstatus'] == 'A']
urban_df_A.groupby(['Medu']).sum().plot(kind='bar',ax=ax[0], legend=True, fi
gsize=(8,5), title="Mother's Education vs Grades in Urban Areas")

urban_df = urban_data[['Pstatus', 'Fedu', 'G1', 'G2', 'G3']]
urban_df_A = urban_df[urban_df['Pstatus'] == 'A']
urban_df_A.groupby(['Fedu']).sum().plot(kind='bar',ax=ax[1], legend=True, fi
gsize=(8,5), title="Father's Education vs Grades in Urban Areas")
```

Out[44]: <matplotlib.axes._subplots.AxesSubplot at 0x23b12a78>





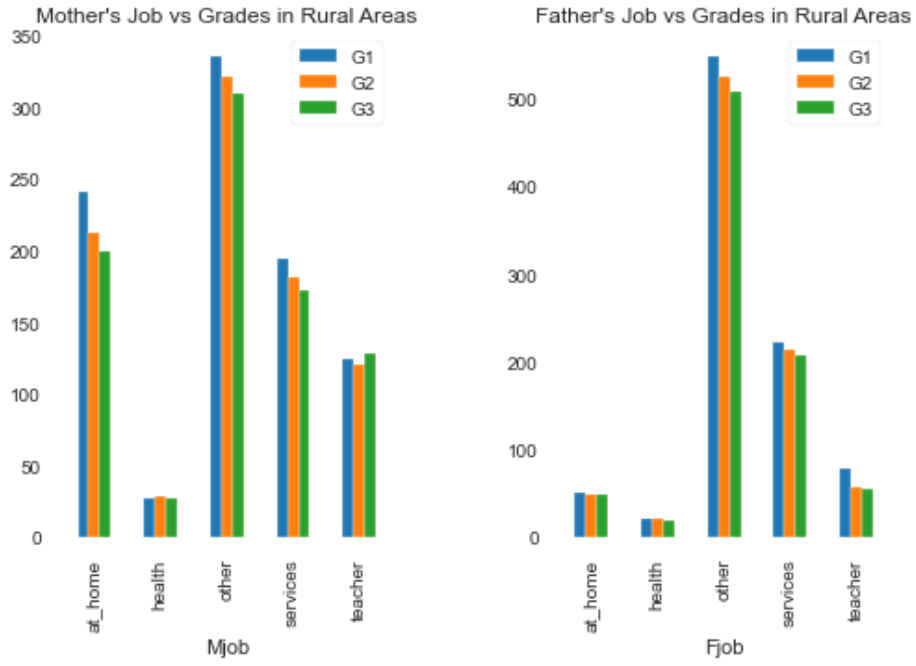


This analysis shows that Trend Mother's & Father's Education vs Grades in Urban Areas. Thing, which one can see that Students whose scores Maximum,mostly his\her parents has secondary education or more in Urban areas.

Trend Mother's & Father's Job vs Grades in Rural Areas

In [45]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)


rural_df = rural_data[['Mjob', 'G1', 'G2', 'G3']]
rural_df.groupby(['Mjob']).sum().plot(kind='bar',ax=ax[0], legend=True, figs
ize=(8,5), title="Mother's Job vs Grades in Rural Areas")

rural_df = rural_data[['Fjob', 'G1', 'G2', 'G3']]
rural_df.groupby(['Fjob']).sum().plot(kind='bar',ax=ax[1], legend=True, figs
ize=(8,5), title="Father's Job vs Grades in Rural Areas")
```

Out[45]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x239cbec8>
```

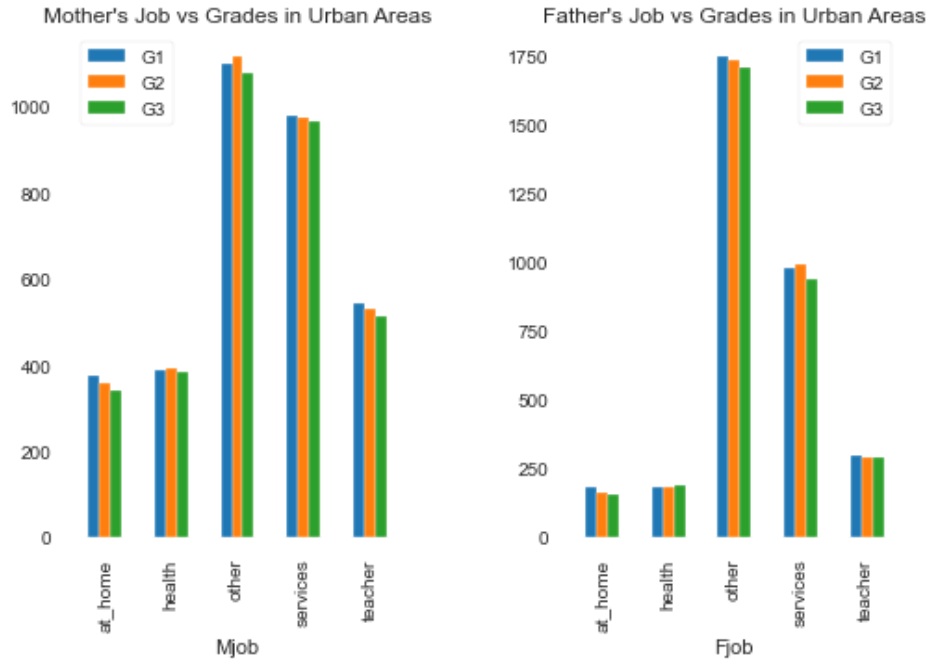


Trend Mother's & Father's Job vs Grades in Urban Areas

In [46]:
```
fig, ax = plt.subplots(1, 2)
```

```
fig.subplots_adjust(hspace=0.5, wspace=0.5)


urban_df = urban_data[['Mjob', 'G1', 'G2', 'G3']]
urban_df.groupby(['Mjob']).sum().plot(kind='bar',ax=ax[0], legend=True, figs
ize=(8,5), title="Mother's Job vs Grades in Urban Areas")

urban_df = urban_data[['Fjob', 'G1', 'G2', 'G3']]
urban_df.groupby(['Fjob']).sum().plot(kind='bar',ax=ax[1], legend=True, figs
ize=(8,5), title="Father's Job vs Grades in Urban Areas")
```

Out[46]: <matplotlib.axes._subplots.AxesSubplot at 0x22955118>



Trend Mother's & Father's Job vs grades while Parent Status is Apart in Rural Area

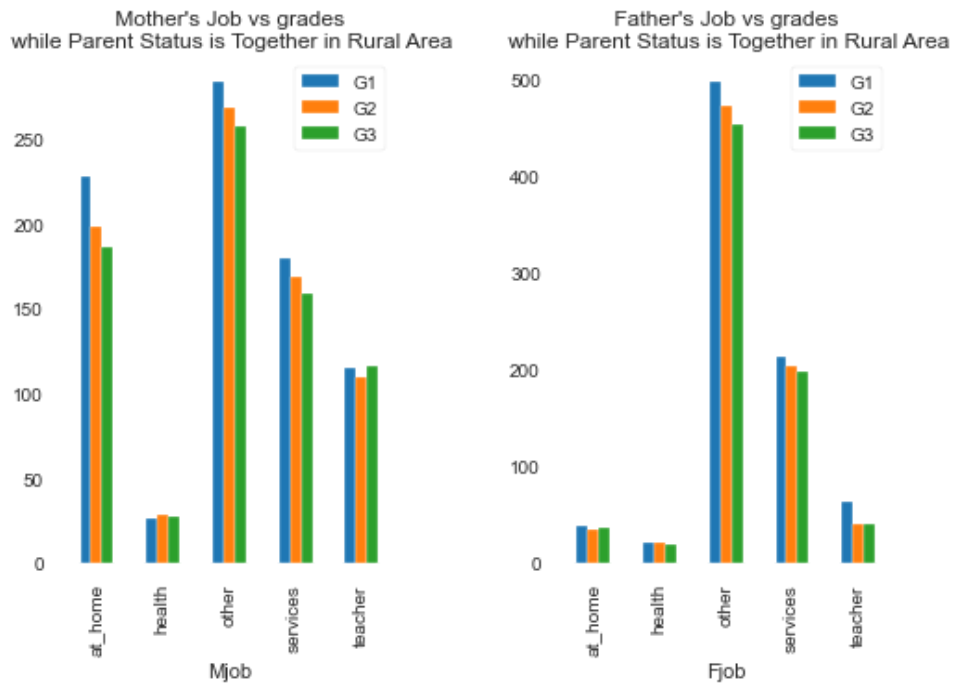
In [47]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=1.5)


rural_df = rural_data[['Pstatus', 'Mjob', 'G1', 'G2', 'G3']]
rural_df_A = rural_df[rural_df['Pstatus'] == 'A']
rural_df_A.groupby(['Mjob']).sum().plot(kind='bar',ax=ax[0], legend=True, fi
gsize=(8,5), title="Mother's Job vs grades \n while Parent Status is Apart i
n Rural Area")

rural_df = rural_data[['Pstatus', 'Fjob', 'G1', 'G2', 'G3']]
rural_df_A = rural_df[rural_df['Pstatus'] == 'A']
rural_df_A.groupby(['Fjob']).sum().plot(kind='bar',ax=ax[1], legend=True, fi
gsize=(8,5), title="Father's Job vs grades \n while Parent Status is Apart i
n Rural Area")
```

Out[47]: <matplotlib.axes._subplots.AxesSubplot at 0x22552cb8>

Trend Mother's & Father's Job vs grades while Parent Status is Together in Rural Area

In [48]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

rural_df = rural_data[['Pstatus', 'Mjob', 'G1', 'G2', 'G3']]
rural_df_A = rural_df[rural_df['Pstatus'] == 'T']
rural_df_A.groupby(['Mjob']).sum().plot(kind='bar',ax=ax[0], legend=True, fi
gsize=(8,5), title=" Mother's Job vs grades \n while Parent Status is Togeth
er in Rural Area")

rural_df = rural_data[['Pstatus', 'Fjob', 'G1', 'G2', 'G3']]
rural_df_A = rural_df[rural_df['Pstatus'] == 'T']
rural_df_A.groupby(['Fjob']).sum().plot(kind='bar',ax=ax[1], legend=True, fi
gsize=(8,5), title=" Father's Job vs grades \n while Parent Status is Togeth
er in Rural Area")
```

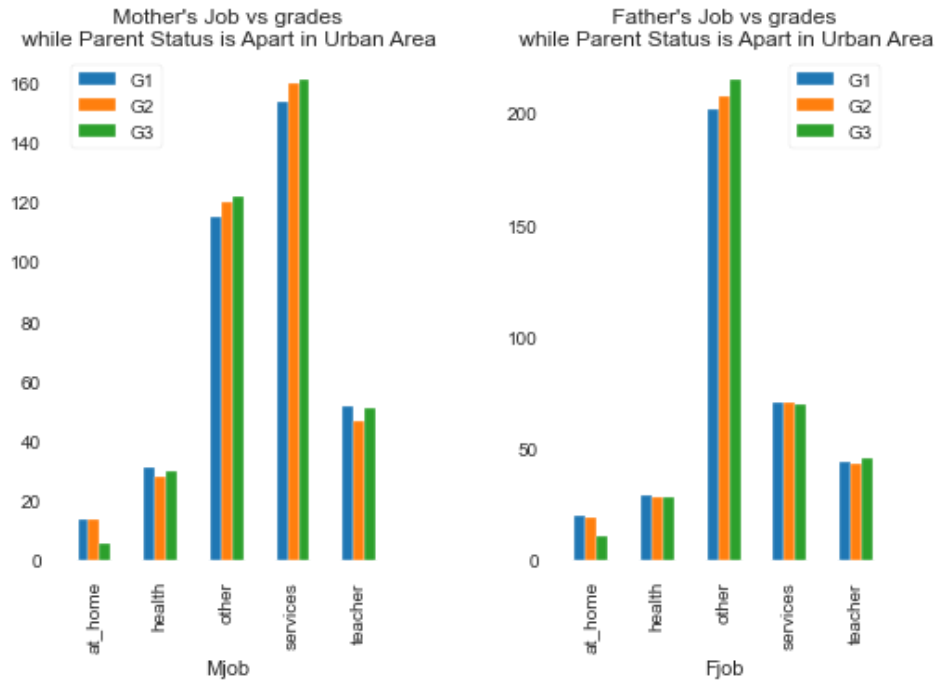
Out[48]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x21258058>
```



Trend Mother's & Father's Job vs grades while Parent Status is Apart in Urban Area

In [49]:
```
fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

urban_df = urban_data[['Pstatus', 'Mjob', 'G1', 'G2', 'G3']]
urban_df_A = urban_df[urban_df['Pstatus'] == 'A']
urban_df_A.groupby(['Mjob']).sum().plot(kind='bar',ax=ax[0], legend=True, fi
gsize=(8,5), title=" Mother's Job vs grades \n while Parent Status is Apart
 in Urban Area")

urban_df = urban_data[['Pstatus', 'Fjob', 'G1', 'G2', 'G3']]
urban_df_A = urban_df[urban_df['Pstatus'] == 'A']
urban_df_A.groupby(['Fjob']).sum().plot(kind='bar',ax=ax[1], legend=True, fi
gsize=(8,5), title=" Father's Job vs grades \n while Parent Status is Apart
 in Urban Area")
```

Out[49]: <matplotlib.axes._subplots.AxesSubplot at 0x1da714d8>

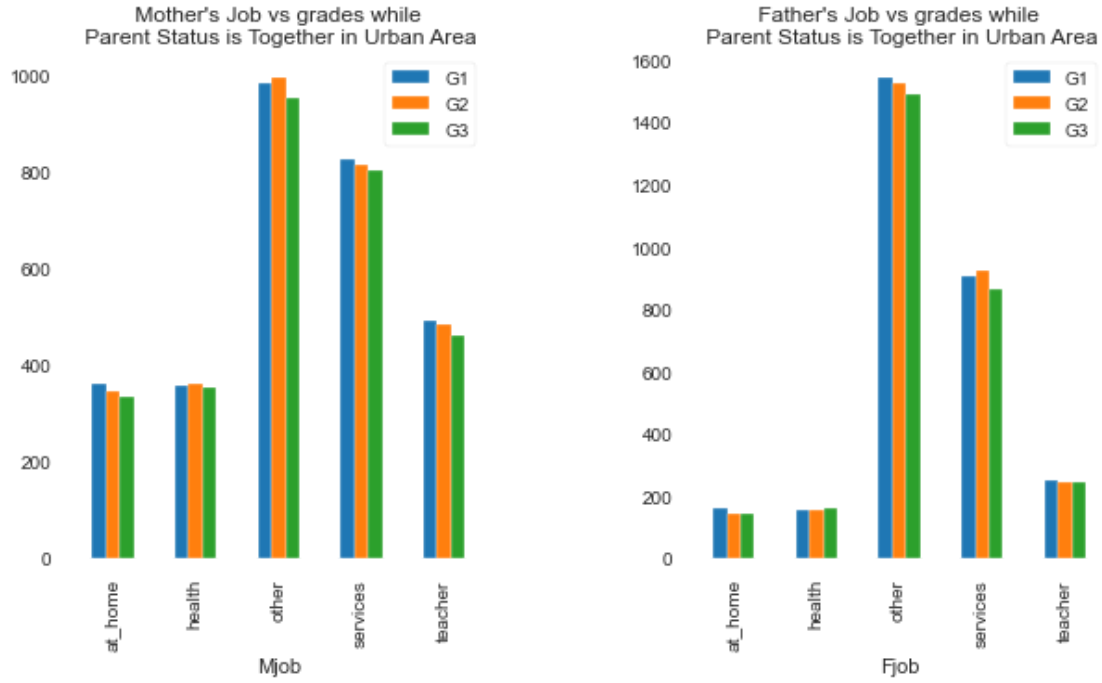


Trend Mother's & Father's Job vs grades while Parent Status is Together in Urban Area

In [50]:
```
## lets focus over support for education now from parents / school
## using Pstatus Mjob / Fjob field here to show impact of this over performa
nce
## urban areas

fig, ax = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

urban_df = urban_data[['Pstatus', 'Mjob', 'G1', 'G2', 'G3']]
urban_df_A = urban_df[urban_df['Pstatus'] == 'T']
urban_df_A.groupby(['Mjob']).sum().plot(kind='bar',ax=ax[0], legend=True, fi
gsize=(10,5), title=" Mother's Job vs grades while \n Parent Status is Toget
her in Urban Area")

urban_df = urban_data[['Pstatus', 'Fjob', 'G1', 'G2', 'G3']]
urban_df_A = urban_df[urban_df['Pstatus'] == 'T']
urban_df_A.groupby(['Fjob']).sum().plot(kind='bar',ax=ax[1], legend=True, fi
```

```
gsize=(10,5), title=" Father's Job vs grades while \n Parent Status is Toget
her in Urban Area")
```

Out[50]: <matplotlib.axes._subplots.AxesSubplot at 0x1a0498e0>



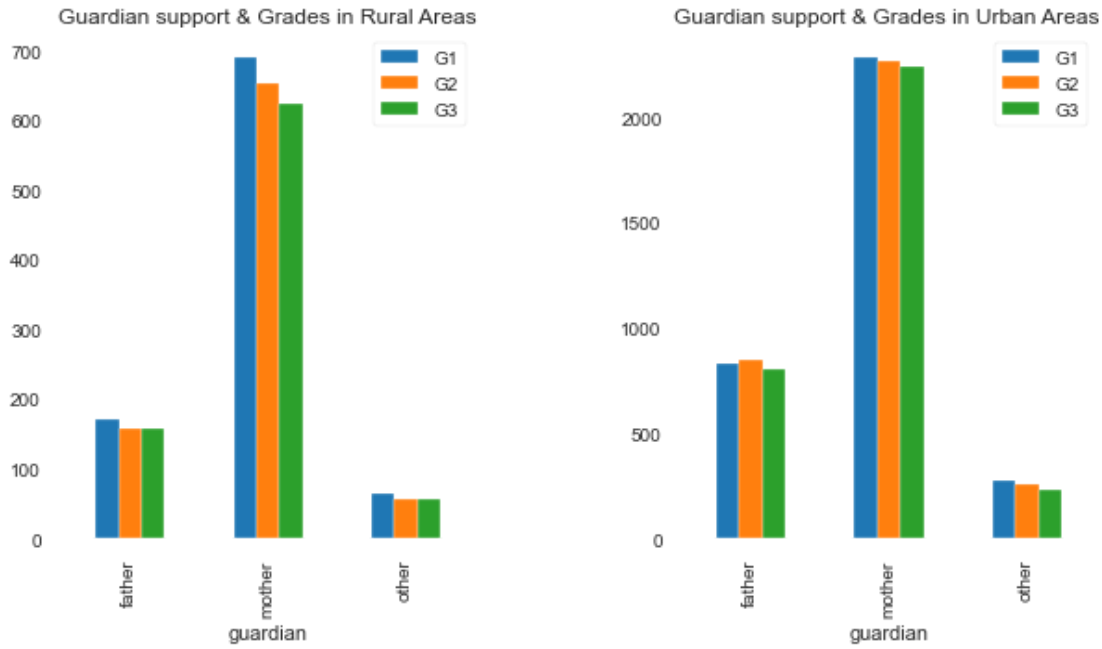
This analysis shows that Age vs Grades of Pstatus Apart & Together in Urban Areas. First Thing, which one can see that Students whose age is 15 has scores Maximum while his\her parents are apart in Urban areas. Second Thing, which one can see that Students whose age is 16 has scores Maximum while his\her parents are together in Urban areas.

Guardian support & Grades in Rural & Urban areas

In [51]:
```
## lets focus over support for education now from parents / school
## using guardian field here to show impact of this over performance
## rural areas / urban areas

fig, ax  = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)
```

```
rural_df = rural_data[['guardian', 'G1', 'G2', 'G3']]
rural_df.groupby(['guardian']).sum().plot(kind='bar',ax=ax[0],figsize=(10,5
), title="Guardian support & Grades in Rural Areas", legend=True)

urban_df = urban_data[['guardian', 'G1', 'G2', 'G3']]
urban_df.groupby(['guardian']).sum().plot(kind='bar',ax=ax[1],figsize=(10,5
), title="Guardian support & Grades in Urban Areas ", legend=True)
```

Out[51]: <matplotlib.axes._subplots.AxesSubplot at 0x14ac72f8>

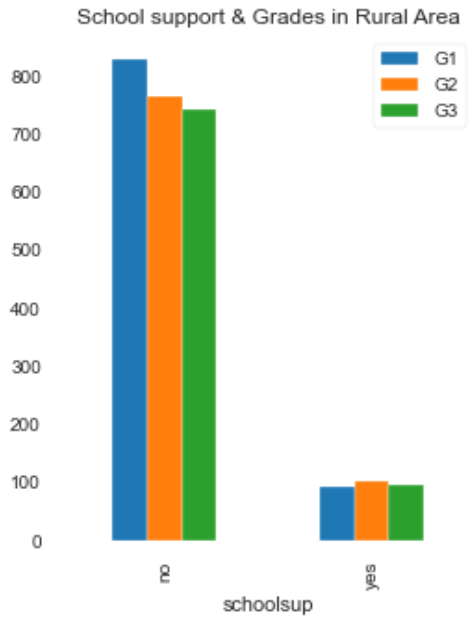


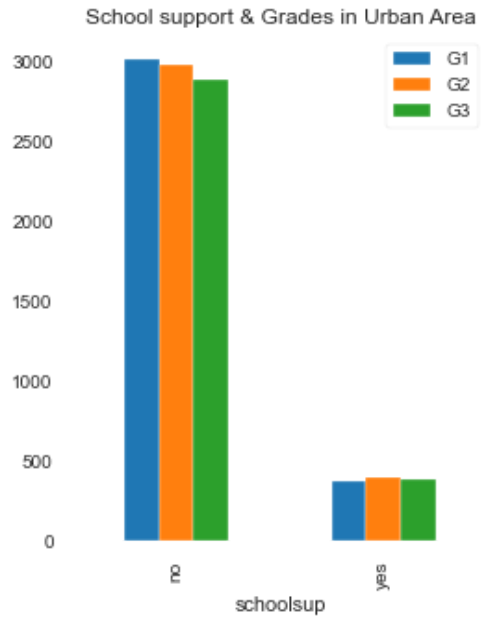
This analysis shows that Age vs Grades of Pstatus Apart & Together in Urban Areas. First Thing, which one can see that Students whose age is 15 has scores Maximum while his\her parents are apart in Urban areas. Second Thing, which one can see that Students whose age is 16 has scores Maximum while his\her parents are together in Urban areas.

School support & Grades in Rural & Urban areas

In [52]:
```
fig, ax  = plt.subplots(1, 2)
fig.subplots_adjust(hspace=0.5, wspace=0.5)

rural_df = rural_data[['schoolsup', 'G1', 'G2', 'G3']]
rural_df.groupby(['schoolsup']).sum().plot(kind='bar',ax=ax[0], figsize=(10,
5), title="School support & Grades in Rural Area")

urban_df = urban_data[['schoolsup', 'G1', 'G2', 'G3']]
urban_df.groupby(['schoolsup']).sum().plot(kind='bar',ax=ax[1], figsize=(10,
5), title="School support & Grades in Urban Area")
```

Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x15e49bb0>





This analysis shows that Age vs Grades of Pstatus Apart & Together in Urban Areas. First Thing, which one can see that Students whose age is 15 has scores Maximum while his\her parents are apart in Urban areas. Second Thing, which one can see that Students whose age is 16 has scores Maximum while his\her parents are together in Urban areas.

In [53]:
```
import plotly.graph_objs as go
from plotly.offline import init_notebook_mode,iplot

student_mat_dataframe = pd.read_csv("student-mat.csv") #read to csv file

trace = go.Scatter3d(
    x = student_mat_dataframe.age,
    y = student_mat_dataframe.Dalc,
    z = student_mat_dataframe.Walc,
    mode = 'markers',
    marker = dict(
        size = 10,
        color = student_mat_dataframe.age,
        colorscale = 'Rainbow'
    )
)
dataTrace = [trace]
layoutTrace = go.Layout(
    margin = dict(
    l = 0,
    r = 0,
    b = 0,
```

```
    t = 0 )
)
figTrace = go.Figure(data=dataTrace, layout=layoutTrace)
iplot(figTrace)
```

Conclusion :

Although there is less data for rural and urban areas but if one take a fine look over the analysis then one can easily understand that students are performing almost equal in terms of grades from both the regions.

Apart from it by looking at Dalc age wise distribution and failures / studytime trend line, one can easily understand that from 16th - 18th age of students are more likely aligned towards alcohol consumption and hence their performance is low than others in exams.

By looking trendlines of parents education and their professions, one can see the impact of these fields over past failures and studytime of students and Pstatus also impacts the performance which one can measure looking at failures / studytime trendlines.

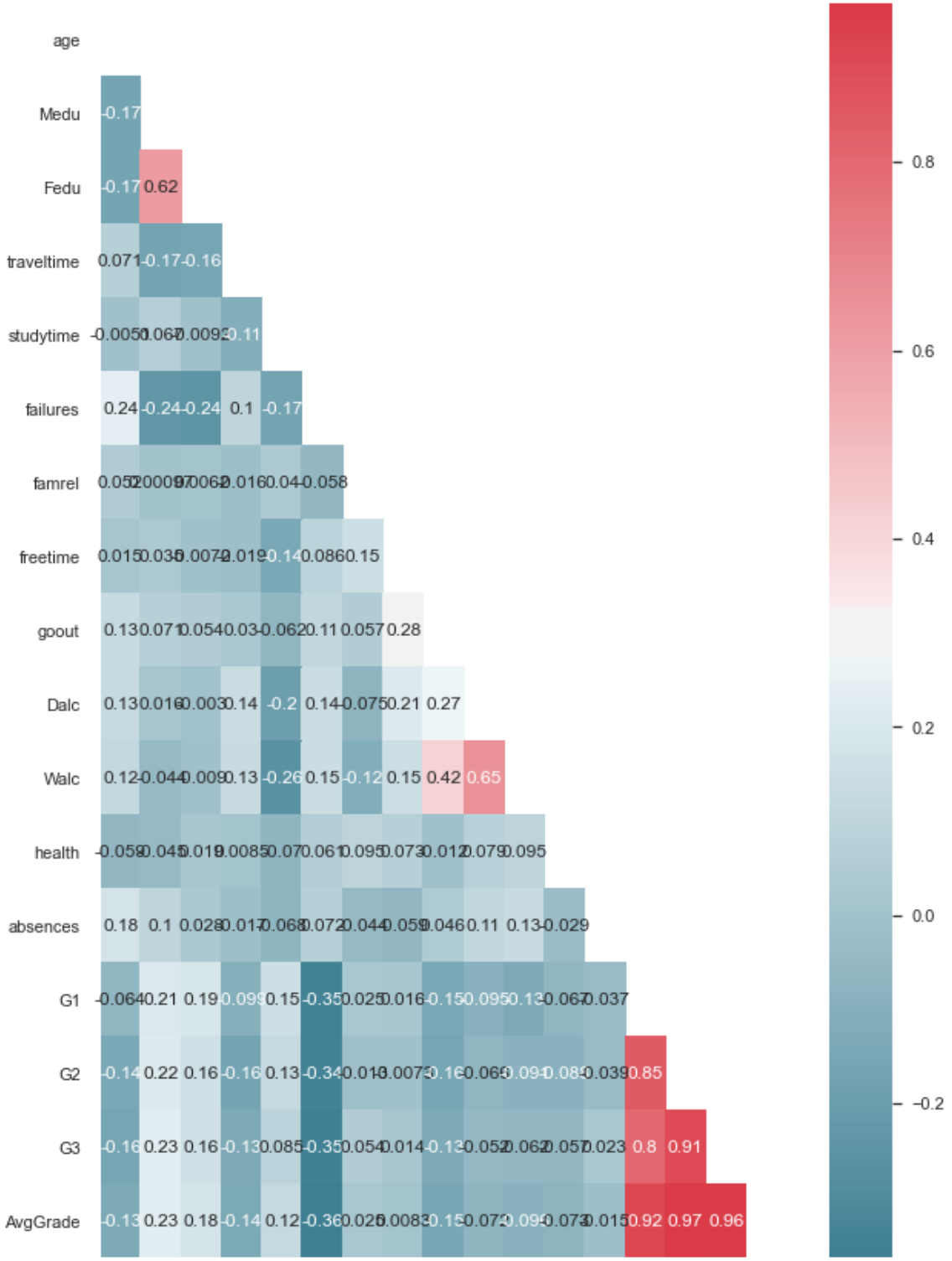
These are the analysis I could able to perform so far. This is my first analysis so I can be wrong somewhere, so please help me to improve myself.

Correlation

In [54]:
```
data=data.drop_duplicates(["school","sex","age","address","famsize","Pstatus","Medu","Fedu",
                           "Mjob","Fjob","reason","nursery","internet"])
data['AvgGrade'] = data[['G1', 'G2', 'G3']].mean(axis=1)
```

```
<ipython-input-54-fbef0d82a9d3>:3: SettingWithCopyWarning:


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: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
```

In [55]:
```
df=data
def heat_map(corrs_mat):
    sns.set(style="white")
    f, ax = plt.subplots(figsize=(10, 15))
    mask = np.zeros_like(corrs_mat, dtype=np.bool)
    mask[np.triu_indices_from(mask)] = True
    # Generate a custom diverging colormap
    cmap = sns.diverging_palette(220, 10, as_cmap=True)
    sns.heatmap(corrs_mat, mask=mask, annot=True, cmap=cmap, ax=ax)

variable_correlations = df.corr()
#variable_correlations
heat_map(variable_correlations)
```


age
Medu
Fedu
traveltime
studytime
failures
famrel
freetime
goout
Dalc
Walc
health
absences
G1
G2
G3
AvgGrade
0.8
0.6
0.4
0.2
0.0
−0.2

age Medu Fedu traveltime studytime failures famrel freetime goout Dalc Walc health absences G1 G2 G3 AvgGrade

Pivot tables

In [56]:
```
df = data
pivot = pd.pivot_table(df,
            values = ['AvgGrade', 'G1', 'G2', 'G3'''],
            index = ['school',
                     'sex',
                    'famsize',
                     'paid',
                    'guardian'],
                       columns= ['Mjob'],
                       aggfunc=[np.mean],
                       margins=True).fillna('')
pivot
```

Out[56]:

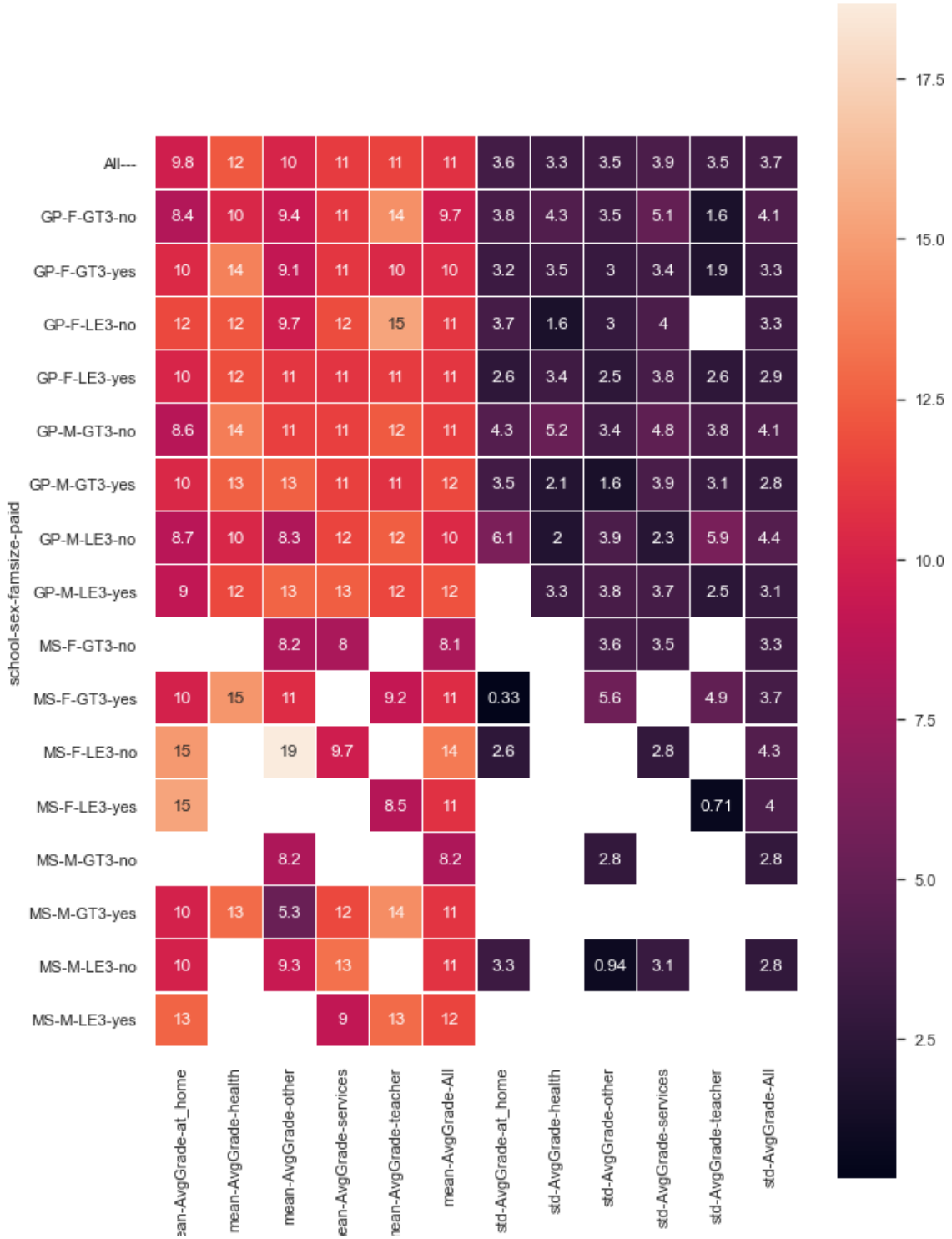
					mean					
					AvgGrade					
				Mjob	at_home	health	other	services	teacher	All
school	sex	famsize	paid	guardian						
GP	F	GT3	no	father	8.11111	6.33333	9.11111	10.3333		9.18750
				mother	7.76923	12.1667	9.74074	11.9048	14.5	9.93939
				other	11.5556		6.5			9.53333
			yes	father	9	16.1667	11.6667	12.8333		12.2666
				mother	10.6667	14.5	9.01852	10.4444	10.3333	10.2916
				other		6.33333	6.44444	11.8333		9.12500
		LE3	no	father	15.3333	13.3333	10.5333	12.3333		11.7777
				mother	10.3333	11	9.66667	14.1667	15.3333	11.4000
				other	10.6667		5.33333	6		7.33333
			yes	father			13	6.33333		9.66666
				mother	10.1667	11.8889	11	11.5556	11.1667	11.2592
				other			8.66667	11.3333		10.0000
	M	GT3	no	father	10		10.0303	14.8	10.3333	11.3157
				mother	8.2	13.6667	11.8571	9.77778	12.8571	11.0151
				other	9		13.6667	17.6667		13.5333
			yes	father		12.3333	11.8333	12.6667		12.1212
				mother	11.6667	12.7778	13.1905	10.9444	10.9667	11.7619
				other	7.66667				8.33333	8.00000
		LE3	no	father		10.6667	9.33333	9		9.66666

				mother	8.66667	10	8.14815	11.9444	12.5714	10.3974
				other					12	12.0000
			yes	father		11.6667	12	9.16667	9.33333	10.7142
				mother	9		13.1111	13.6111	12	12.6250
MS	F	GT3	no	father			11.6667	9.33333		10.5000
				mother			7.33333	7.33333		7.33333
			yes	father	10.3333		12.3333		12.6667	11.7777
				mother	9.83333		9.66667		5.66667	8.93333
				other		14.6667				14.6666
		LE3	no	father	16.6667					16.6666
				mother	13		18.6667	11.6667		14.4444
				other				7.66667		7.66666
			yes	mother	15.3333				8.5	10.7777
	M	GT3	no	father			10.6667			10.6666
				mother			7.55556			7.55555
				other			8			8.00000
			yes	father			5.33333		14.3333	9.83333
				mother	10	13		11.6667		11.5555
		LE3	no	father			9.33333			9.33333
				mother	10			13.1667		11.5833
			yes	father	12.6667					12.6666
				mother					13	13.0000
				other				9		9.00000
All					9.76271	12.2353	10.119	11.1961	11.4762	10.7246

43 rows × 24 columns

In [57]:
```
pivot = pd.pivot_table(df,
            values = ['AvgGrade'],
            index = ['school',
                     'sex',
                    'famsize',
                     'paid',
                    ],
                           columns= ['Mjob'],
                           aggfunc=[np.mean, np.std],
                           margins=True)
pivot
cmap = sns.cubehelix_palette(start = 1.5, rot = 1.5, as_cmap = True)
plt.subplots(figsize = (10, 15))
sns.heatmap(pivot,linewidths=0.2,annot=True,square=True )#annotations=True
```

Out[57]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x19f60868>
```


school-sex-famsize-paid
All---
GP-F-GT3-no
GP-F-GT3-yes
GP-F-LE3-no
GP-F-LE3-yes
GP-M-GT3-no
GP-M-GT3-yes
GP-M-LE3-no
GP-M-LE3-yes
MS-F-GT3-no
MS-F-GT3-yes
MS-F-LE3-no
MS-F-LE3-yes
MS-M-GT3-no
MS-M-GT3-yes
MS-M-LE3-no
MS-M-LE3-yes
mean-AvgGrade-at_home
mean-AvgGrade-health
mean-AvgGrade-other
mean-AvgGrade-services
mean-AvgGrade-teacher
mean-AvgGrade-All
std-AvgGrade-at_home
std-AvgGrade-health
std-AvgGrade-other
std-AvgGrade-services
std-AvgGrade-teacher
std-AvgGrade-All
17.5
15.0
12.5
10.0
7.5
5.0
2.5

None-None-Mjob

In [58]:
```
df = data
pivot = pd.pivot_table(df,
            values = ['AvgGrade'],
            index = ['school',
                     'sex',
                     'famsize',
                     'paid'],
                      columns= ['Fjob'],
                      aggfunc=[np.mean],
                      margins=True).fillna('')
pivot
```

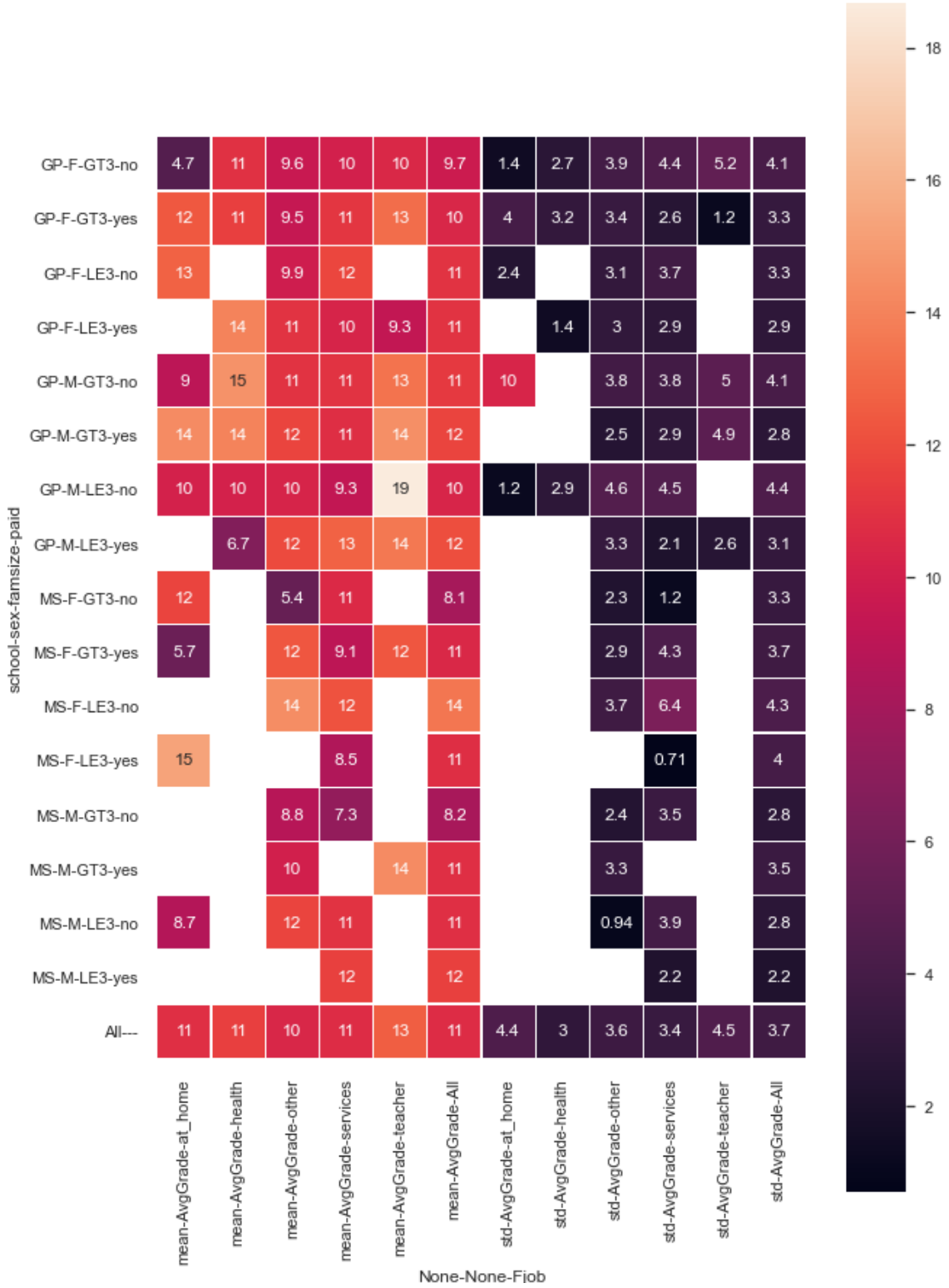
Out[58]:

				mean					
				AvgGrade					
			Fjob	at_home	health	other	services	teacher	All
school	sex	famsize	paid						
GP	F	GT3	no	4.66667	10.8889	9.58559	10.0714	10.4815	9.723077
			yes	12.4	11.4762	9.45614	11.0909	13.1667	10.432432
		LE3	no	12.75		9.87879	11.7619		11.000000
			yes		14	10.9524	10.2667	9.33333	11.000000
	M	GT3	no	9	14.6667	11.019	11	13.375	11.284314
			yes	14.3333	14	11.6667	10.8056	14.5	11.674797
		LE3	no	10.1667	10.1111	10.2982	9.26667	18.6667	10.377778
			yes		6.66667	11.9111	12.75	13.5556	12.043478
MS	F	GT3	no	11.6667		5.41667	10.5556		8.125000
			yes	5.66667		12.3333	9.11111	12.3333	10.518519
		LE3	no			14.4444	12.1667		13.533333
			yes	15.3333			8.5		10.777778
	M	GT3	no			8.83333	7.33333		8.190476
			yes			10		14.3333	10.866667
		LE3	no	8.66667		11.6667	11		10.833333
			yes				11.5556		11.555556
All				10.8167	11.4815	10.4217	10.7061	12.5714	10.724638

In [59]:
```
pivot = pd.pivot_table(df,
            values = ['AvgGrade'],
            index = ['school',
                     'sex',
                     'famsize',
                     'paid'],
                      columns= ['Fjob'],
                      aggfunc=[np.mean, np.std],
```

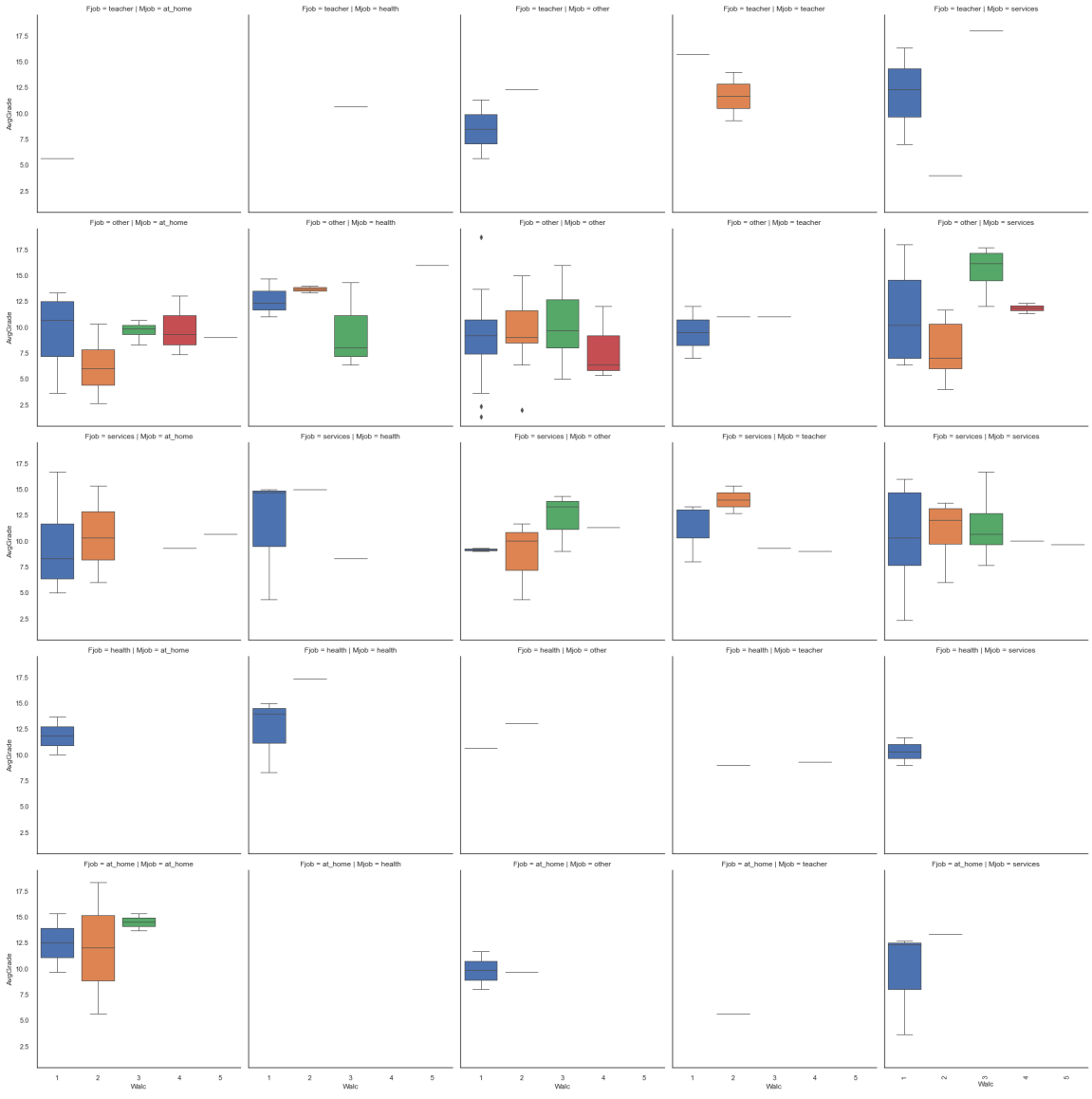
```
                            margins=True)
cmap = sns.cubehelix_palette(start = 1.5, rot = 1.5, as_cmap = True)
plt.subplots(figsize = (10, 15))
sns.heatmap(pivot,linewidths=0.2,annot=True,square=True)
```

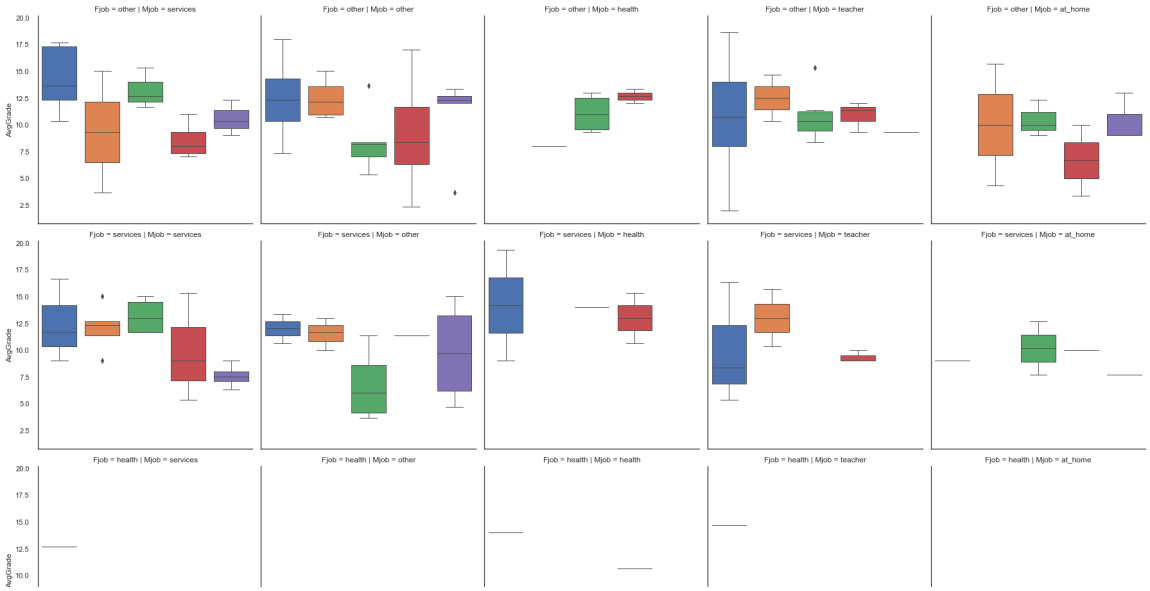
Out[59]: <matplotlib.axes._subplots.AxesSubplot at 0x1c800430>



Simple Box Plots

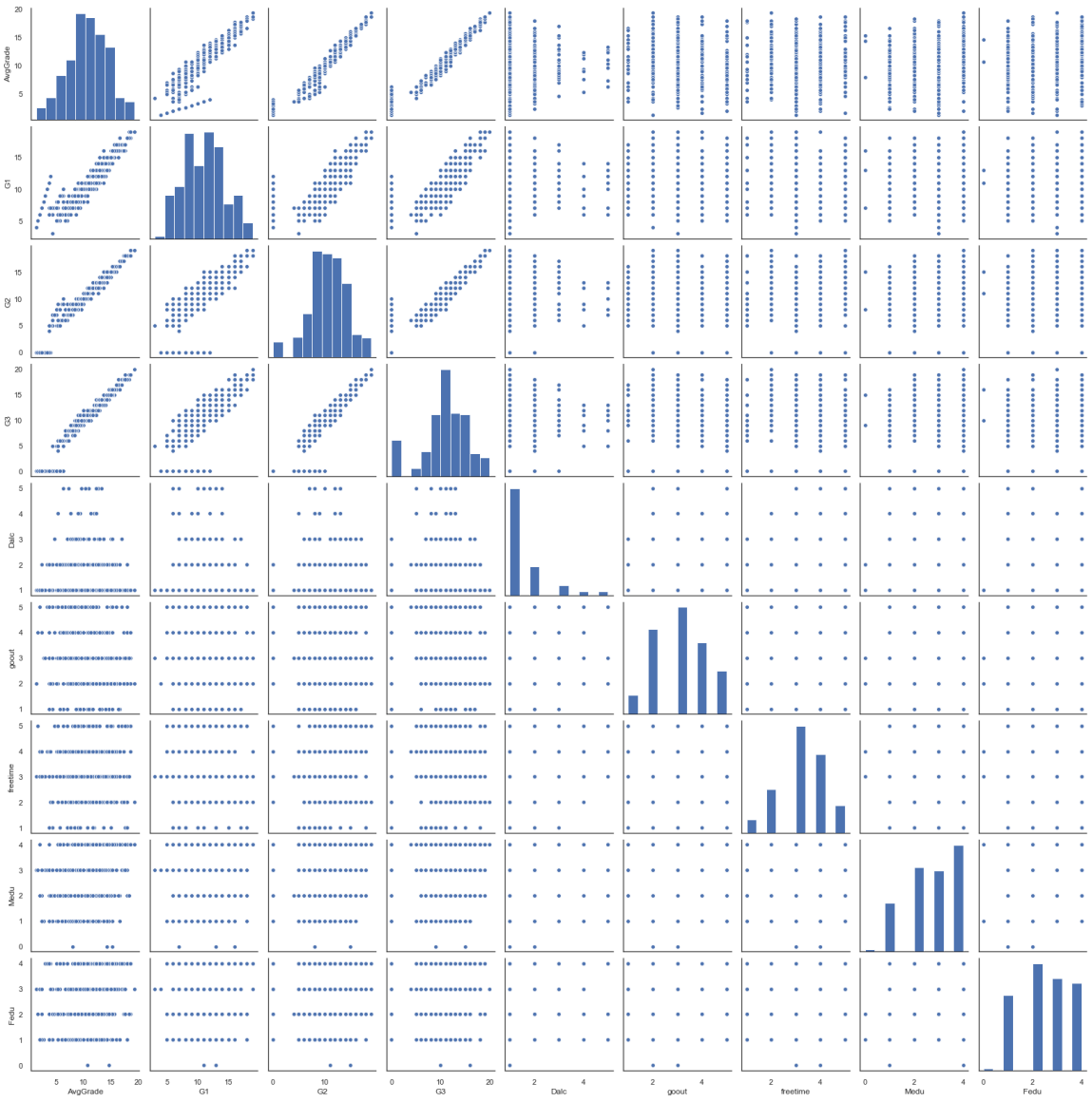
In [60]:
```
InputFile2_reduced=df
for i in set(InputFile2_reduced['sex']):
    aa= InputFile2_reduced[InputFile2_reduced['sex'].isin([i])]
    g = sns.catplot(x='Walc', y="AvgGrade",data=aa,
                    saturation=1, kind="box",
                    ci=None, aspect=1, linewidth=1, row='Fjob', col = 'Mjob')
    locs, labels = plt.xticks()
    plt.setp(labels, rotation=90)
```





In [61]:
```
df_small = df[['AvgGrade',
               'G1',
               'G2',
               'G3',
               'Dalc',
               'goout',
               'freetime',
               'Medu',
               'Fedu']]
sns.pairplot(df_small)
```

Out[61]: <seaborn.axisgrid.PairGrid at 0x1a059c70>



In [62]:
```
#data = df

sns.set(style="white", palette="muted", color_codes=True)
f, axes = plt.subplots(4, 4, figsize=(20,20))
sns.despine(left=True)
sns.distplot(df['AvgGrade'],  kde=False, color="b", ax=axes[0, 0])
sns.distplot(df['G3'],        kde=False, color="b", ax=axes[0, 1])
sns.distplot(df['G2'],        kde=False, color="b", ax=axes[0, 2])
sns.distplot(df['G1'],        kde=False, color="b", ax=axes[0, 3])
sns.distplot(df['studytime'], kde=False, color="b", ax=axes[1, 0])
sns.distplot(df['freetime'],  kde=False, color="b", ax=axes[1, 1])
```

```
sns.distplot(df['goout'],      kde=False, color="b", ax=axes[1, 2])
sns.distplot(df['absences'],   kde=False, color="b", ax=axes[1, 3])
sns.distplot(df['Dalc'],       kde=False, color="b", ax=axes[2, 0])
sns.distplot(df['Walc'],       kde=False, color="b", ax=axes[2, 1])
sns.distplot(df['health'],     kde=False, color="b", ax=axes[2, 2])
sns.distplot(df['famrel'],     kde=False, color="b", ax=axes[2, 3])
sns.distplot(df['traveltime'],kde=False, color="b", ax=axes[3, 0])
sns.distplot(df['age'],        kde=False, color="b", ax=axes[3, 1])
sns.distplot(df['Medu'],       kde=False, color="b", ax=axes[3, 2])
sns.distplot(df['Fedu'],       kde=False, color="b", ax=axes[3, 3])
plt.tight_layout()
```

Feature importance

In [ ]:
```
from sklearn.linear_model import SGDClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.calibration import CalibratedClassifierCV
#import xgboost as xgb
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import StratifiedKFold
from sklearn.feature_selection import SelectFromModel
from sklearn.linear_model import LogisticRegression
from sklearn import svm

mod_df=df

df_copy = pd.get_dummies(mod_df)

df1 = df_copy
y = np.asarray(df1['AvgGrade'], dtype="|S6")
df1 = df1.drop(['AvgGrade'],axis=1)
X = df1.values
Xtrain, Xtest, ytrain, ytest = train_test_split(X, y, test_size=0.50)

radm = RandomForestClassifier()
radm.fit(Xtrain, ytrain)

clf = radm
indices = np.argsort(radm.feature_importances_)[::-1]

# Print the feature ranking
print('Feature ranking:')

for f in range(df1.shape[1]):
    print('%d. feature %d %s (%f)' % (f+1 ,
                                      indices[f],
                                      df1.columns[indices[f]],
                                      radm.feature_importances_[indices[f
]]))
```

Feature ranking:

1. feature 32 G3 (0.099127)
2. feature 31 G2 (0.090746)
3. feature 30 G1 (0.083324)
4. feature 29 absences (0.047501)
5. feature 2 age (0.047210)
6. feature 8 Mjob (0.042698)
7. feature 10 reason (0.039888)
8. feature 9 Fjob (0.038109)
9. feature 24 freetime (0.036717)
10. feature 27 Walc (0.036440)
11. feature 28 health (0.036254)
12. feature 25 goout (0.035556)
13. feature 7 Fedu (0.032512)
14. feature 6 Medu (0.029738)
15. feature 13 studytime (0.028424)

16. feature 23 famrel (0.022782)
17. feature 14 failures (0.021496)
18. feature 4 famsize (0.020987)
19. feature 12 traveltime (0.019854)
20. feature 16 famsup (0.018525)
21. feature 22 romantic (0.018305)
22. feature 15 schoolsup (0.016997)
23. feature 0 school (0.016006)
24. feature 18 activities (0.015686)
25. feature 21 internet (0.015028)
26. feature 26 Dalc (0.014583)
27. feature 1 sex (0.014086)
28. feature 17 paid (0.013742)
29. feature 19 nursery (0.013234)
30. feature 3 address (0.012410)
31. feature 11 guardian (0.007684)
32. feature 20 higher (0.007477)
33. feature 5 Pstatus (0.006875)

In [ ]:
```
import warnings
warnings.filterwarnings('ignore')
from sklearn.decomposition import PCA
from sklearn.model_selection import cross_val_score
from sklearn.feature_selection import RFECV, SelectKBest
from sklearn.tree import DecisionTreeRegressor
from sklearn.linear_model import LinearRegression, Lasso
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier, Ext
raTreesClassifier
from sklearn.tree import DecisionTreeClassifier, ExtraTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB, BernoulliNB
from sklearn.neighbors import KNeighborsClassifier

classifiers = [('RandomForestClassifierG', RandomForestClassifier(n_jobs=-1,
criterion='gini')),
               ('RandomForestClassifierE', RandomForestClassifier(n_jobs=-1,
criterion='entropy')),
               ('AdaBoostClassifier', AdaBoostClassifier()),
               ('ExtraTreesClassifier', ExtraTreesClassifier(n_jobs=-1)),
               ('KNeighborsClassifier', KNeighborsClassifier(n_jobs=-1)),
               ('DecisionTreeClassifier', DecisionTreeClassifier()),
               ('ExtraTreeClassifier', ExtraTreeClassifier()),
               ('LogisticRegression', LogisticRegression()),
               ('GaussianNB', GaussianNB()),
               ('BernoulliNB', BernoulliNB()),
              ]
allscores = []

x, Y = mod_df.drop('AvgGrade', axis=1), np.asarray(mod_df['AvgGrade'], dtype
="|S6")

for name, classifier in classifiers:
    scores = []
    for i in range(20): # 20 runs
        roc = cross_val_score(classifier, x, Y)
        scores.extend(list(roc))
    scores = np.array(scores)
    print(name, scores.mean())
```

```
    new_data = [(name, score) for score in scores]
    allscores.extend(new_data)
```

RandomForestClassifierG 0.187486720641

RandomForestClassifierE 0.178953321325

AdaBoostClassifier 0.111638803935

ExtraTreesClassifier 0.160233050953

KNeighborsClassifier 0.152225891389

DecisionTreeClassifier 0.431037095584

ExtraTreeClassifier 0.097145971318

LogisticRegression 0.0702088297653

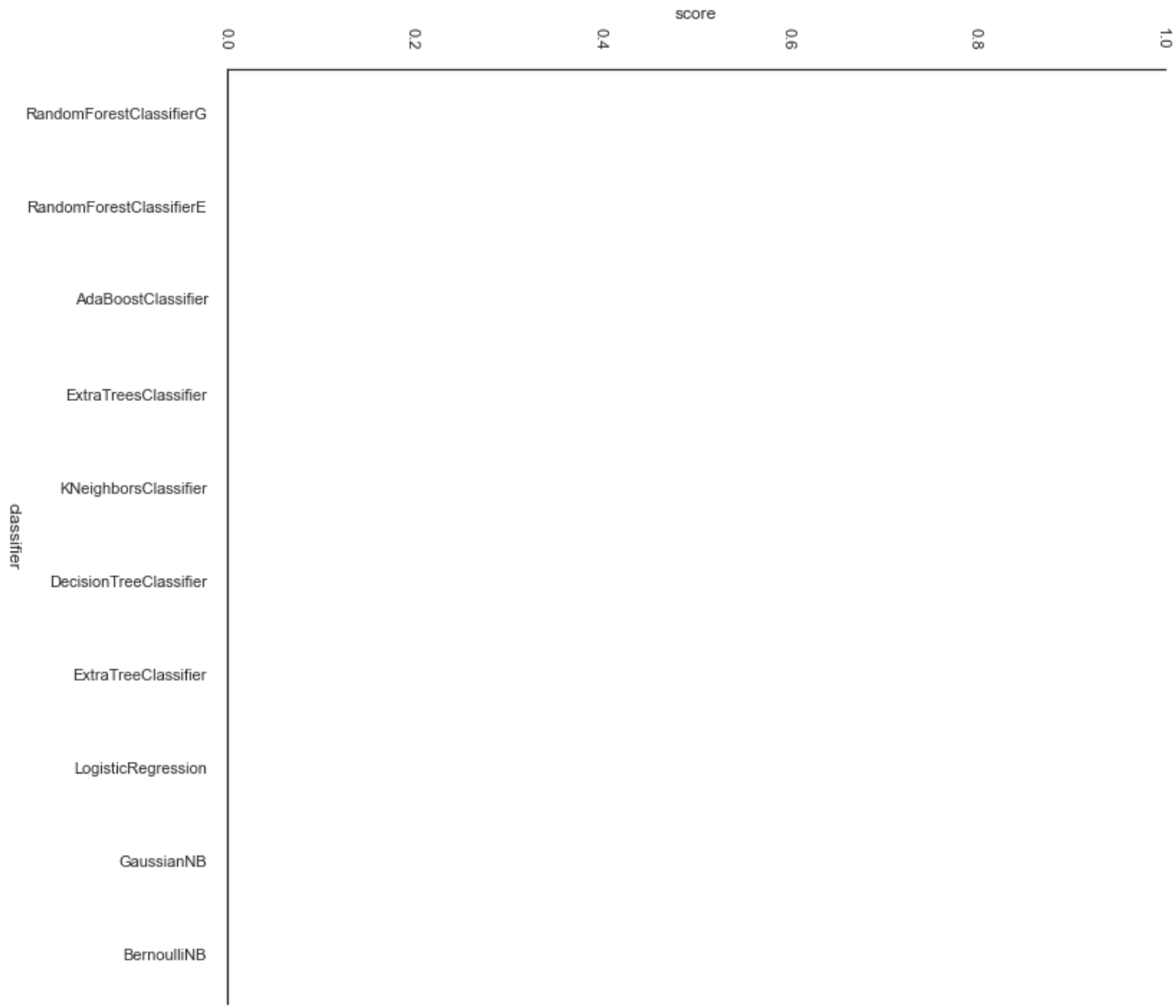
GaussianNB 0.10938420549

BernoulliNB 0.049166651031

In [65]:
```
temp = pd.DataFrame(allscores, columns=['classifier', 'score'])
#sns.violinplot('classifier', 'score', data=temp, inner=None, linewidth=0.3)
plt.figure(figsize=(15,10))
sns.factorplot(x='classifier',
               y="score",
               data=temp,
               saturation=1,
               kind="box",
               ci=None,
               aspect=1,
               linewidth=1,
               size = 10)
locs, labels = plt.xticks()
plt.setp(labels, rotation=90)
```

Out[65]:
```
[None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None,
 None]
```

<Figure size 1080x720 with 0 Axes>


score
0.0
0.2
0.4
0.6
0.8
1.0
classifier
RandomForestClassifierG
RandomForestClassifierE
AdaBoostClassifier
ExtraTreesClassifier
KNeighborsClassifier
DecisionTreeClassifier
ExtraTreeClassifier
LogisticRegression
GaussianNB
BernoulliNB

None of these classifiers have good prediction capability.

IQR Score

First we calculate IQR score to filter out the outliers by keeping only valid values.

In [66]:
```
from scipy.stats import ttest_1samp, t
ub=pd.read_csv("student-mat.csv")
Q1 = ub['Walc'].quantile(0.25)
Q3 = ub['Walc'].quantile(0.75)
IQR = Q3 - Q1
IQR
```

Out[66]: 2.0

In [67]:
```
alc_new = ub['Walc'][ub['Walc'].between(Q1-1.5*IQR, Q3+1.5*IQR)]

print(f"{len(alc_new)}/{len(ub['Walc'])} observations remain in Walc variable.")
```

```
395/395 observations remain in Walc variable.
```

In [68]:
```
alc_new.describe().to_frame().style.background_gradient(cmap='coolwarm')
```

Out[68]:

	Walc
count	395.000000
mean	2.291139
std	1.287897
min	1.000000
25%	1.000000
50%	2.000000
75%	3.000000
max	5.000000

In [69]:
```
sample = alc_new.sample(25, random_state=42)
sample.head()
```

Out[69]:
```
78     1
371    3
248    3
55     1
390    5
Name: Walc, dtype: int64
```

Hypothesis

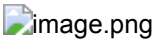
Can we infer that in the age 20, the average of Consumption for Alcohol is 2 days or more ?

Null Hypotheis: $(H_0)

Alternate Hypotheis: $(H_a)

Level of significance: $\alpha = 0.05$

degree of freedom: `len(sample) - 1`



In [70]:
```
t_stat, _ = ttest_1samp(sample.values, popmean=2.5
                        )

alpha = 0.05
df = len(sample) - 1
cv = t.ppf(1.0-alpha/2, df) # Critical value

df_ttest = pd.DataFrame({"Test Statistic": [abs(t_stat)],
                         "Critical Value": [cv]},
                        index=["Sample Data"]
                        )
df_ttest.style.background_gradient(cmap='coolwarm')
```

Out[70]:

	Test Statistic	Critical Value
Sample Data	1.846372	2.063899

Since our test statistic value is less than the critical value, we have good evidence to not reject the null-hypothesis at the 0.05 significance level.

Thus, we can infer the average of Consumption for Alcohol is 2 days or more in the age 20 is 2.5.

In [71]:
```
data =pd.read_csv("student-mat.csv")
```

In [72]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 395 entries, 0 to 394
Data columns (total 33 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   school      395 non-null    object
```

```
 1   sex          395 non-null    object
 2   age          395 non-null    int64
 3   address      395 non-null    object
 4   famsize      395 non-null    object
 5   Pstatus      395 non-null    object
 6   Medu         395 non-null    int64
 7   Fedu         395 non-null    int64
 8   Mjob         395 non-null    object
 9   Fjob         395 non-null    object
 10  reason       395 non-null    object
 11  guardian     395 non-null    object
 12  traveltime   395 non-null    int64
 13  studytime    395 non-null    int64
 14  failures     395 non-null    int64
 15  schoolsup    395 non-null    object
 16  famsup       395 non-null    object
 17  paid         395 non-null    object
 18  activities   395 non-null    object
 19  nursery      395 non-null    object
 20  higher       395 non-null    object
 21  internet     395 non-null    object
 22  romantic     395 non-null    object
 23  famrel       395 non-null    int64
 24  freetime     395 non-null    int64
 25  goout        395 non-null    int64
 26  Dalc         395 non-null    int64
 27  Walc         395 non-null    int64
 28  health       395 non-null    int64
 29  absences     395 non-null    int64
 30  G1           395 non-null    int64
 31  G2           395 non-null    int64
 32  G3           395 non-null    int64
dtypes: int64(16), object(17)
memory usage: 75.7+ KB
```

Does the study time differ by gender?

In [ ]:
```
#student["studytime"] = factor(student["studytime"], levels=c(1,2,3,4), labe
ls=c("1-2","2-5","5-10",">10"))
chisq.test(student["studytime"], student["sex"])
```

```
	Pearson's Chi-squared test
```

data: student*studytimeandstudent*sex

X-squared = 50.634, df = 3, p-value = 5.854e-11

yes, at the level of significance of 0.05, as the p-value = 5.854e-11 is less than 0.05

Does the study time depend on being in a romantic relationship?

In [ ]:
```
chisq.test(table(student["studytime"], student["Pstatus"]))
```

```
	Pearson's Chi-squared test
```

data: table(student*studytime*, *student*Pstatus).

X-squared = 10.665, df = 3, p-value = 0.01368

the p-value = 0.01368 is less that 0.05 so studytime and Pstatus are dependent

Does the number of past class failures depend on gender?

In [ ]:
```
t.test(student["failures"]~student["sex"])
```

```
	Welch Two Sample t-test
```

data: student*failures*by*student*sex

t = -0.87636, df = 374.61, p-value = 0.3814

alternative hypothesis: true difference in means is not equal to 0

```
95 percent confidence interval:
```

-0.2144090 0.0822103

sample estimates:

```
mean in group F mean in group M 
      0.3028846       0.3689840 
```

The p-value = 0.3814 is more that 0.05 so failures and sex are independent

Does the number of absences depend on gender?

In [ ]:
```
t.test(student["absences"]~student["sex"])
```

```
	Welch Two Sample t-test
```

data: student*absences*by*student*sex

t = 1.3611, df = 354.35, p-value = 0.1743

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-0.4769038 2.6208261

sample estimates:

mean in group F mean in group M

6.216346 5.144385

the p-value = 0.1743 is more that 0.05 so absences and sex are independent