Creating Plots on Data Aware Grids

Using FacetGrid, factorplot and lmplot

Tidy data

Seaborn's grid plots require data in "tidy format" i,.e one observation per row of data

	INSTNM	OPEID	REGION	SAT_AVG_ALL	PCTPELL	PCTFLOAN	ADM_RATE_ALL	UG	AVGFACSAL	COMPL_RPY_5YR_RT	DEBT_MDN
0	Alabama A & M University	100200	5	850.0	0.7249	0.8159	0.653841	4380.0	7017.0	0.477631579	14600
1	University of Alabama at Birmingham	105200	5	1147.0	0.3505	0.5218	0.604275	10331.0	10221.0	0.673230442	14250
2	Amridge University	2503400	5	NaN	0.7455	0.8781	NaN	98.0	3217.0	0.636363636	11082
3	University of Alabama in Huntsville	105500	5	1221.0	0.3179	0.4589	0.811971	5220.0	9514.0	0.762222222	15000
4	Alabama State University	100500	5	844.0	0.7567	0.7692	0.463858	4348.0	7940.0	0.43006993	15274

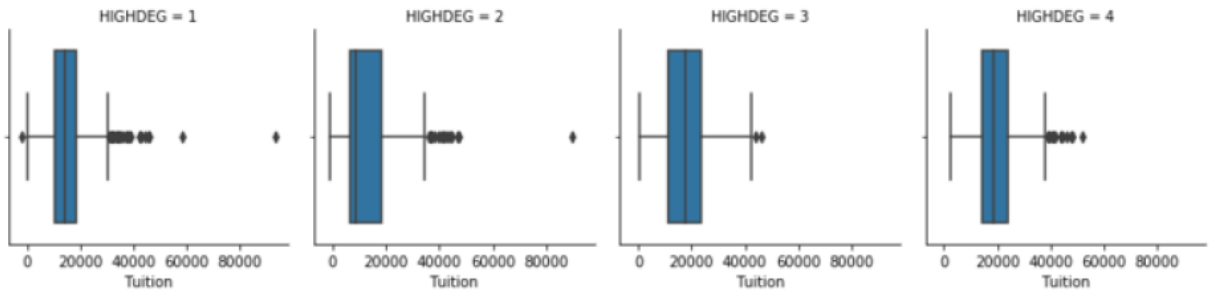
FacetGrid

- The `FacetGrid` is foundational for many data aware grids

- It allows the user to control how data is distributed across columns, rows and hue
- Once a `FacetGrid` is created, the plot type must be mapped to the grid

FacetGrid Categorical Example

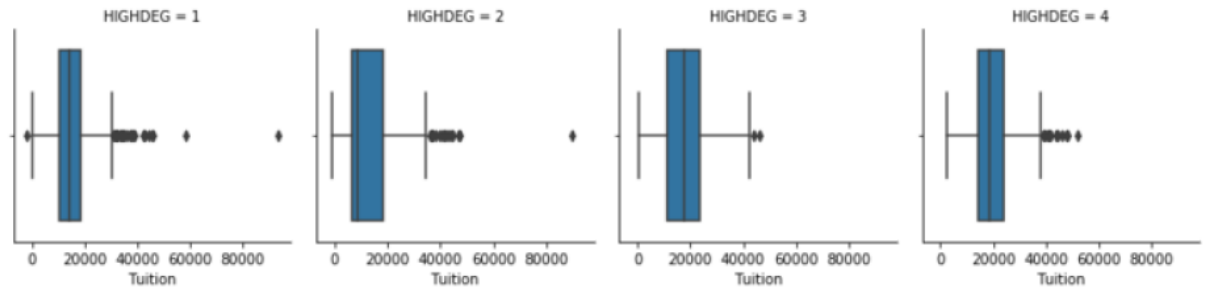
```
g = sns.FacetGrid(df, col="HIGHDEG")
g.map(sns.boxplot, 'Tuition',
      order=['1', '2', '3', '4'])
```



factorplot()

- The `factorplot` is a simpler way to use a `FacetGrid` for categorical data
- Combines the facetting and mapping process into 1 function.

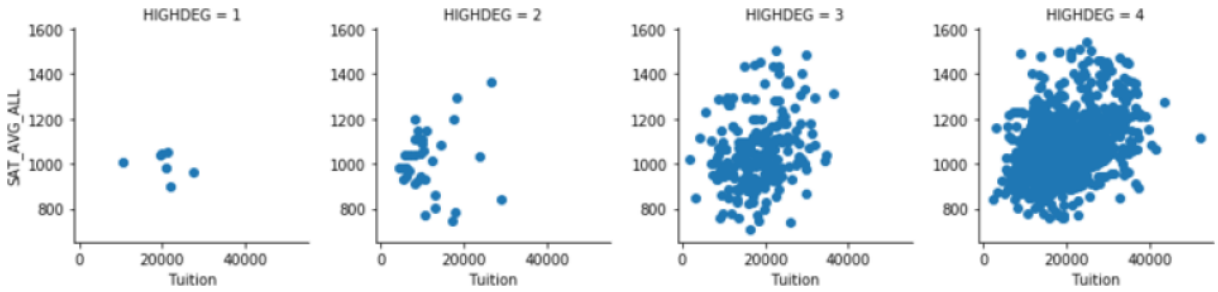
```
sns.factorplot(x="Tuition", data=df,
               col="HIGHDEG", kind='box')
```



FacetGrid for regression

`FacetGrid()` can also be used for scatter or regression plot.

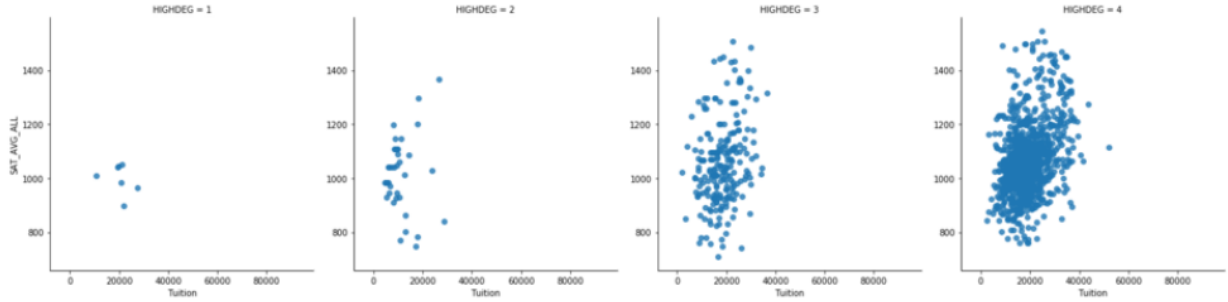
```
g = sns.FacetGrid(df, col="HIGHDEG")
g.map(plt.scatter, 'Tuition', 'SAT_AVG_ALL')
```



lmplot

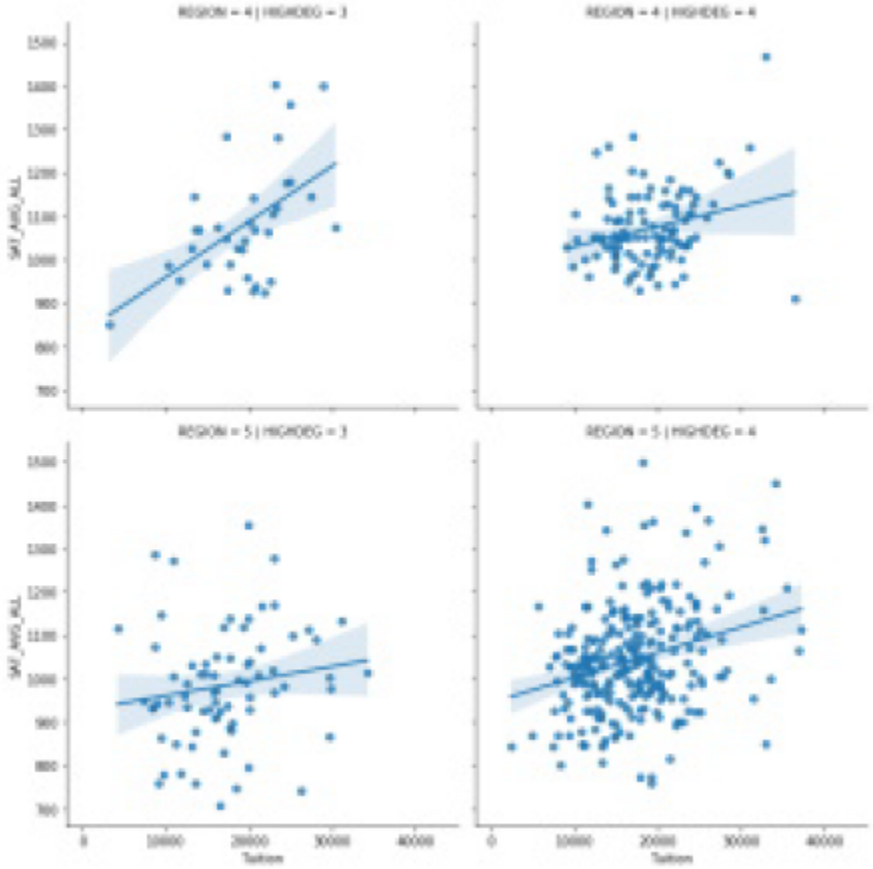
`lmplot` plots scatter and regression plots on a `FacetGrid`

```
sns.lmplot(data=df, x="Tuition", y="SAT_AVG_ALL",
           col="HIGHDEG", fit_reg=False)
```



lmplot with regression

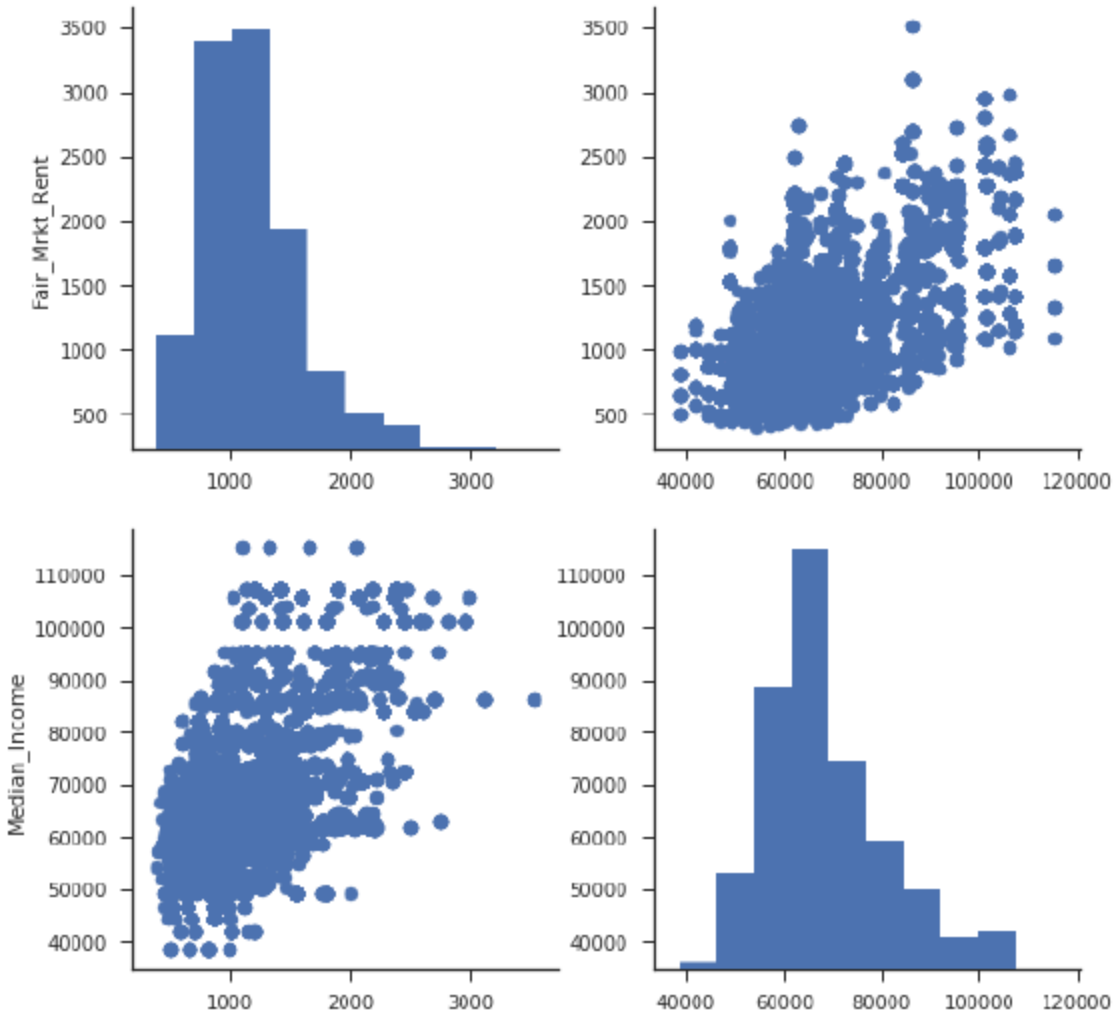
```
sns.lmplot(data=df, x="Tuition", y="SAT_AVG_ALL",
           col="HIGHDEG", row='REGION')
```



Using PairGrid and pairplot

Pairwise relationships

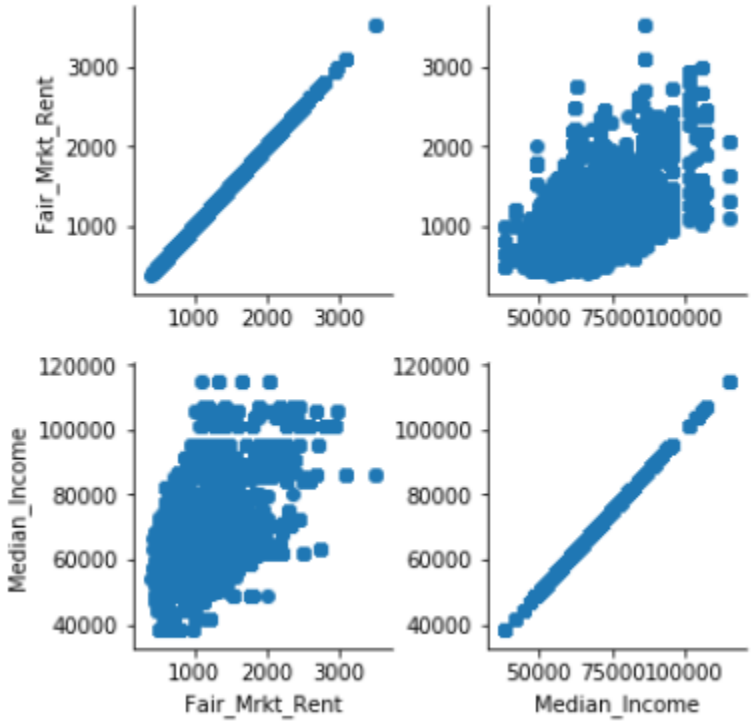
`PairGrid` shows pairwise relationships between data elements



Creating a PairGrid

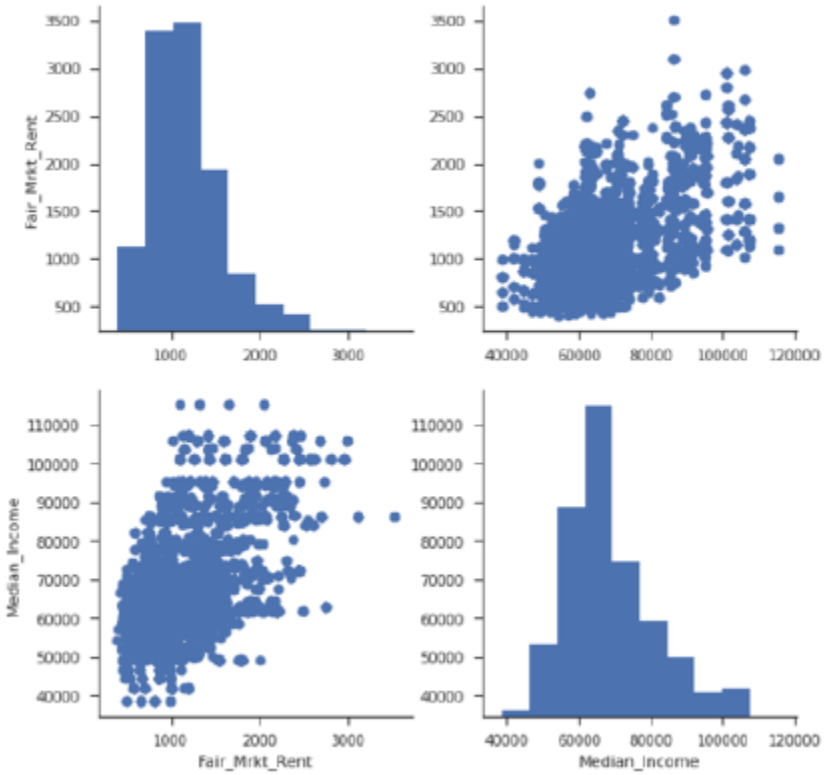
The `PairGrid` follows similar API to FacetGrid

```
g = sns.PairGrid(df, vars=["Fair_Mrkt_Rent",
                 "Median_Income"])
g = g.map(plt.scatter)
```



Customizing the PairGrid diagonals

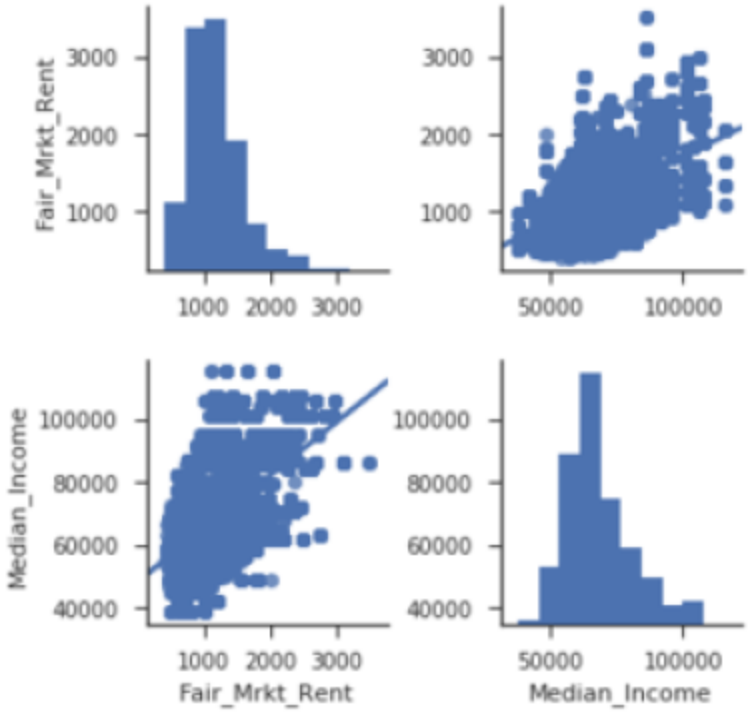
```
g = sns.PairGrid(df, vars=["Fair_Mrkt_Rent",
                 "Median_Income"])
g = g.map_diag(plt.hist)
g = g.map_offdiag(plt.scatter)
```



Pairplot

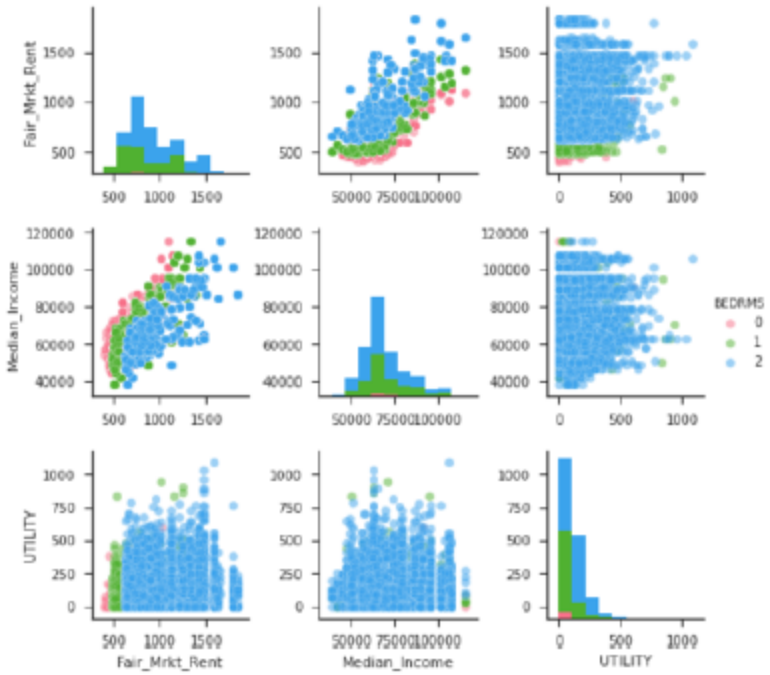
`pairplot` is a shortcut for the `PairGrid`

```
sns.pairplot(df, vars=["Fair_Mrkt_Rent",
            "Median_Income"], kind='reg',
            diag_kind='hist')
```



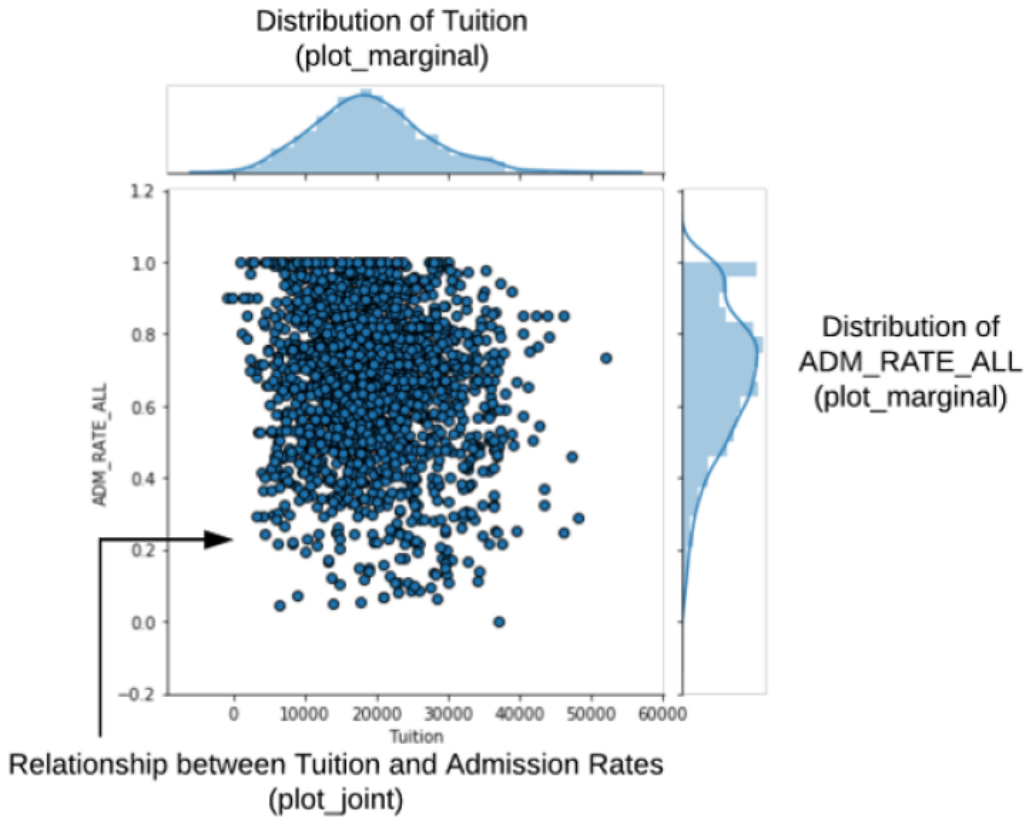
Customizing a pairplot

```
sns.pairplot(df.query('BEDRMS < 3'),
             vars=["Fair_Mrkt_Rent",
             "Median_Income", "UTILITY"],
             hue='BEDRMS', palette='husl',
             plot_kws={'alpha': 0.5})
```



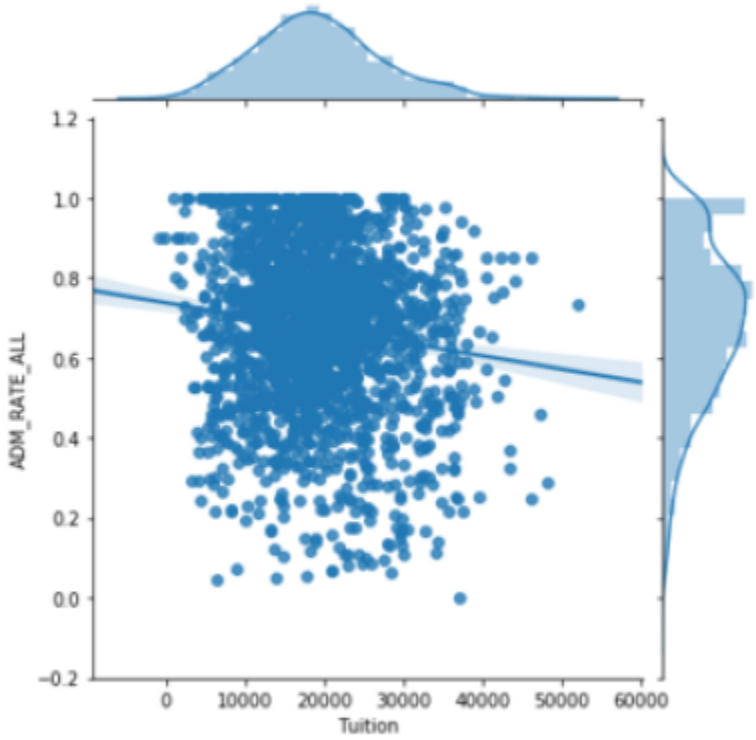
Using JointGrid and jointplot

JointGrid() Overview



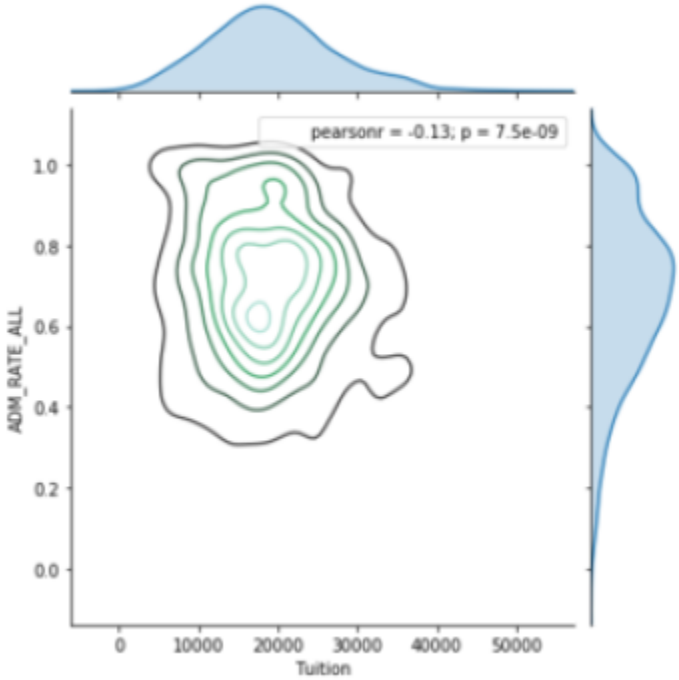
Basic JointGrid

```
g = sns.JointGrid(data=df, x="Tuition",
      y="ADM_RATE_ALL")
g.plot(sns.regplot, sns.distplot)
```



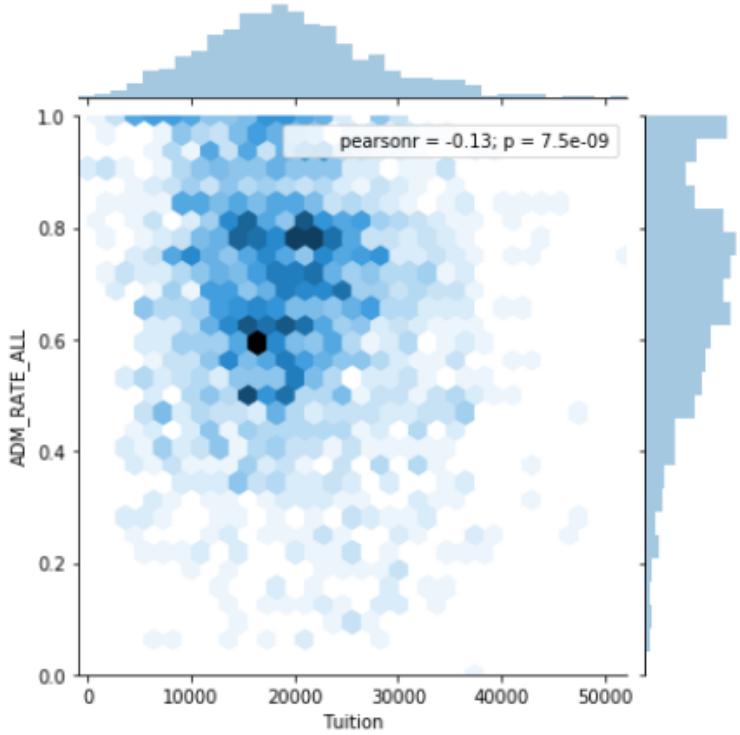
Advanced JointGrid

```
g = sns.JointGrid(data=df, x="Tuition",
                  y="ADM_RATE_ALL")
g = g.plot_joint(sns.kdeplot)
g = g.plot_marginals(sns.kdeplot, shade=True)
g = g.annotate(stats.pearsonr)
```



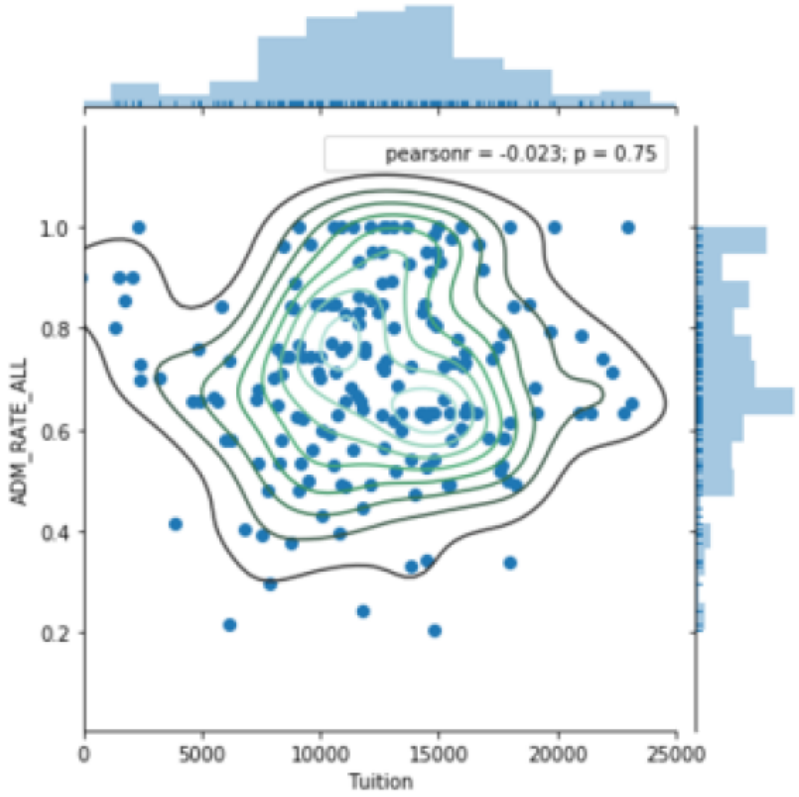
jointplot()

```
sns.jointplot(data=df, x="Tuition",
              y="ADM_RATE_ALL", kind='hex')
```

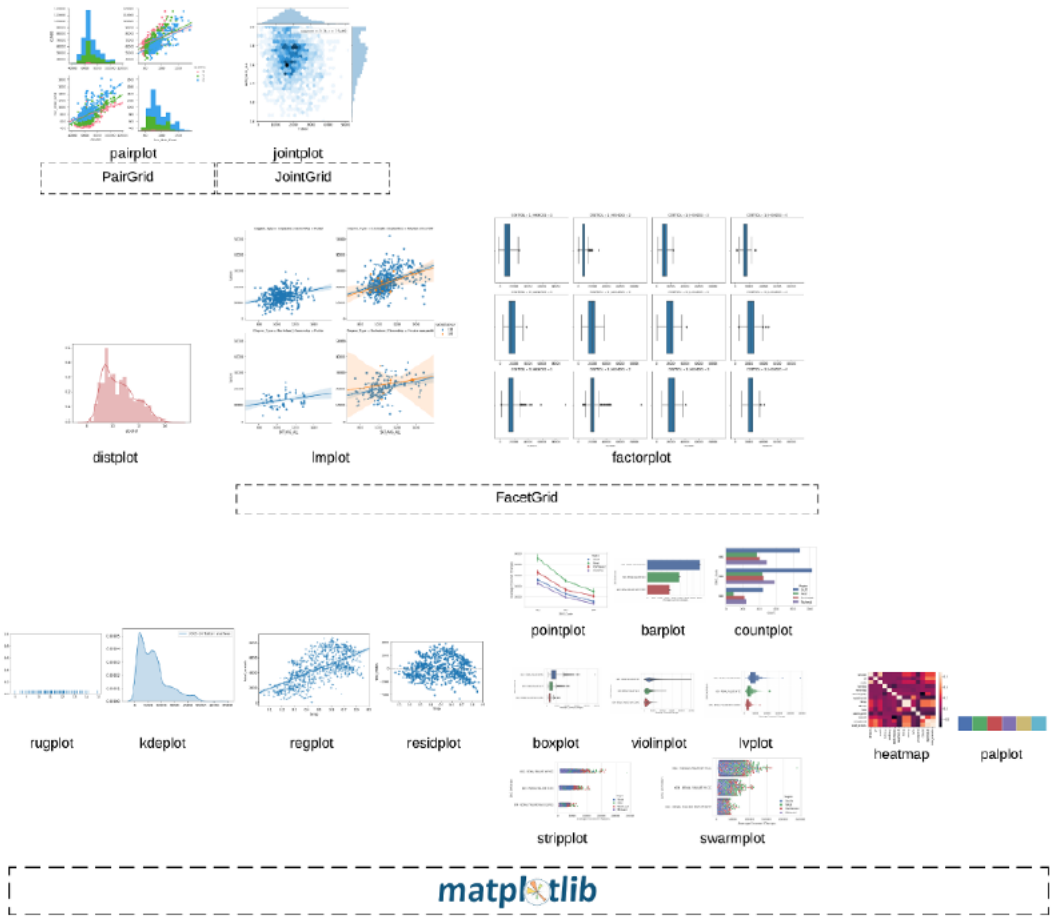


Customizing a jointplot

```
g = (sns.jointplot(x="Tuition",
                   y="ADM_RATE_ALL", kind='scatter',
                   xlim=(0, 25000),
                   marginal_kws=dict(
                   bins=15,rug=True),
                   data=df.query('UG < 2500 &
                   Ownership == "Public"'))
  .plot_joint(sns.kdeplot))
```

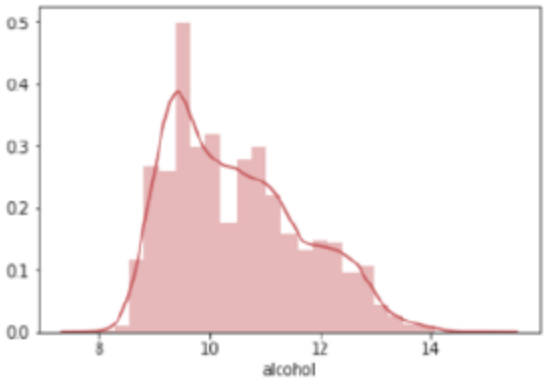


Selecting Seaborn Plots



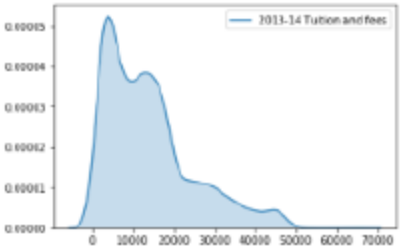
Univariate Distribution Analysis

- `distplot()` is the best place to start for this analysis
- `rugplot()` and `kdeplot()` can be useful alternatives



distplot

rugplot



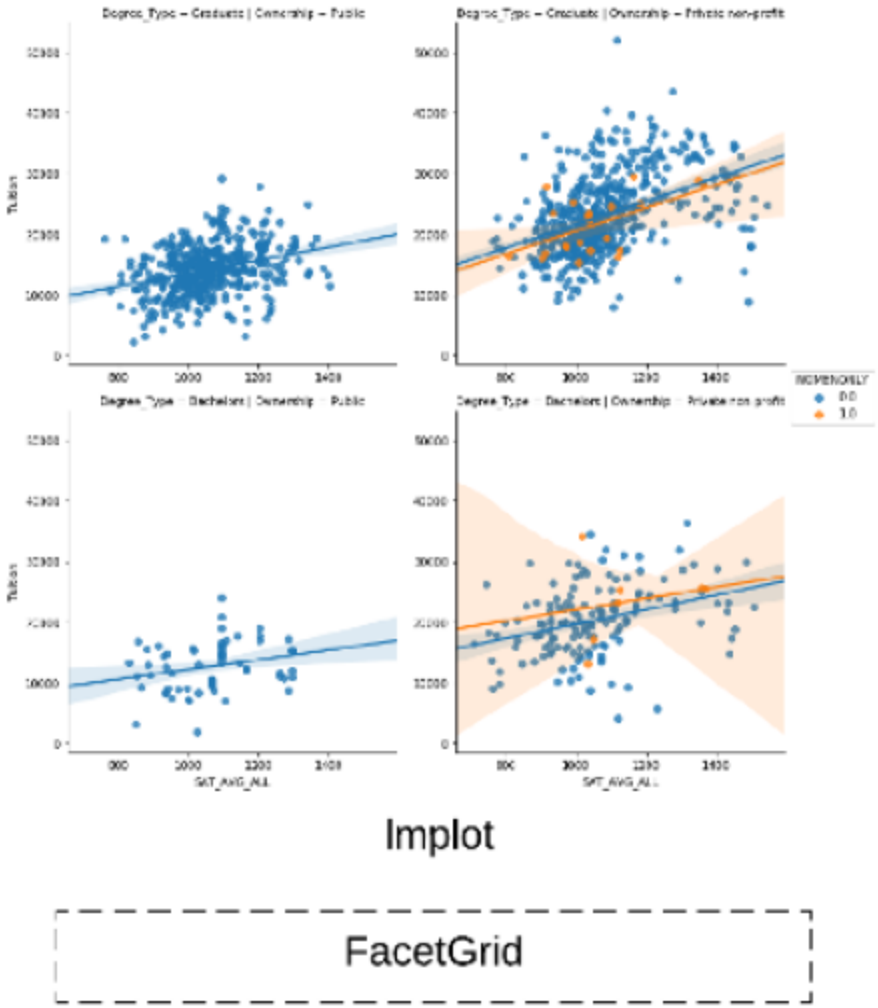
kdeplot

plt.hist()

matplotlib

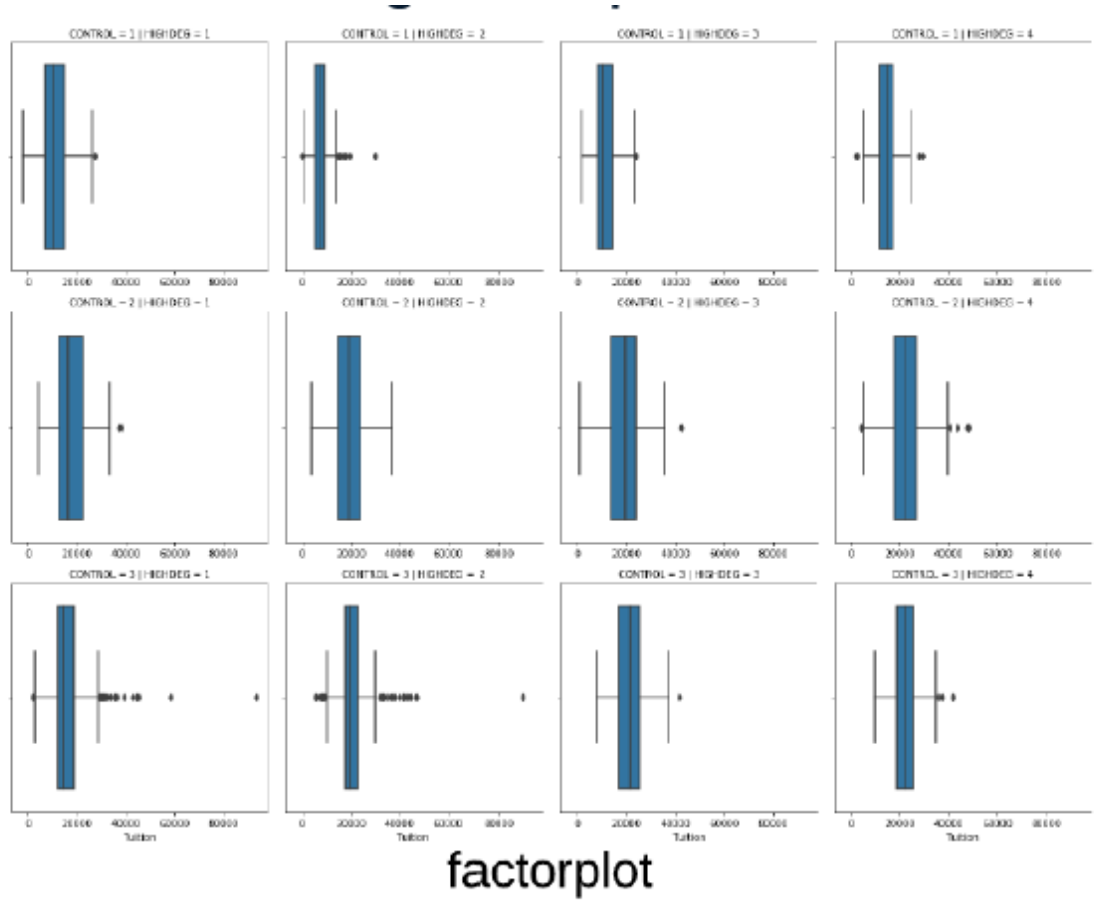
Regression Analysis

`lmplot()` performs regression analysis and supports facetting



Categorical Plots

Explore data with the categorical plots and facet with

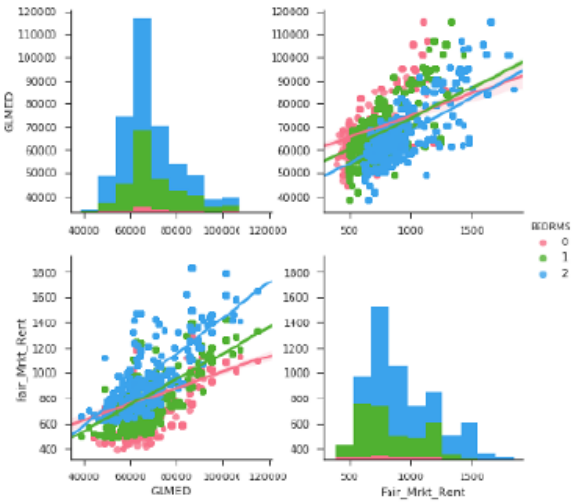


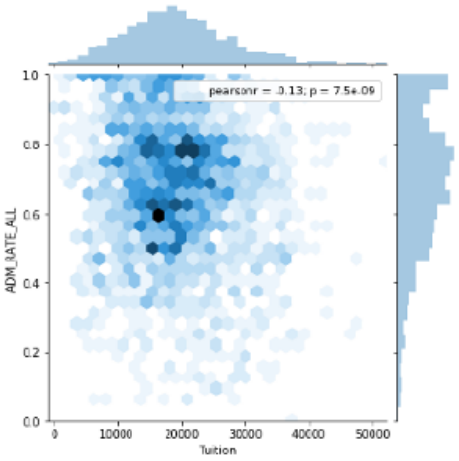
factorplot

FacetGrid

pairplot() and jointplot()

- Perform regression analysis with `lmplot`
- Analyze distributions with `distplot`





pairplot

jointplot

PairGrid

JointGrid