Data manipulation and plotting

- pip install bokeh
- pip install lightning-python

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib
import plotly
```

```
/Users/tomas/miniconda2/envs/py27_nb/lib/python2.7/site-packages/mat
plotlib/font_manager.py:273: UserWarning:

Matplotlib is building the font cache using fc-list. This may take a
 moment.
```

Intro python data manipulation

In [3]:

```
# Create demo data
mean, cov = [0, 1], [(1, .5), (.5, 1)]
data = np.random.multivariate_normal(mean, cov, 200)
df = pd.DataFrame(data, columns=["x", "y"])
df['z'],df['zz'] = 1,1
df['w']=np.random.uniform(1,0,200)
df.head()
```

Out[3]:

	x	y	z	zz	w
0	1.488354	0.179012	1	1	0.770852
1	0.139258	0.135126	1	1	0.347032
2	1.107870	1.658317	1	1	0.361021
3	0.366891	0.275043	1	1	0.604694
4	-0.700017	0.689024	1	1	0.291650

In [4]:

```
df['grp'] = ['A' if x > 4 else 'B' if  x > 1 else 'C' for x in df.sum(axis=1)]
df['grpX'] = ['A' if x > 1 else 'B' if  x > .5 else 'C' for x in df.x]

df.head()
```

Out[4]:

	x	y	z	zz	w	grp	grpX
0	1.488354	0.179012	1	1	0.770852	A	A
1	0.139258	0.135126	1	1	0.347032	B	C
2	1.107870	1.658317	1	1	0.361021	A	A
3	0.366891	0.275043	1	1	0.604694	B	C
4	-0.700017	0.689024	1	1	0.291650	B	C

In [5]:

```
#df = DataFrame({'d': np.random.randint(-20, 20, 100)})
bins = [-1, -.5, 0, .5, 1, 1.5]
df['labels'] = np.digitize(df['y'], bins) - 3
df['labels'].value_counts()
```

Out[5]:

```
 3    59
 2    43
 1    33
 0    30
-1    19
-2    11
-3     5
Name: labels, dtype: int64
```

Group functions and calculations

In [6]:

```
def S(array):
    s = np.sum(array)
    return s

def test_add():
    def inner(group):
        return S(group)
    inner.__name__ = 'grpRes'
    return inner
```

In [7]:

```
print(df.head())
print(df.shape)
print(df.grpX.value_counts())

# predefined numpy function
foo = df.groupby(['grpX'])['z'].agg([np.sum])

# user defined function
bar = df.groupby(['grpX'])['zz'].apply(test_add())

print(foo.head())
print(bar.head())
pd.DataFrame(bar)
```

```
          x         y  z  zz         w grp grpX  labels
0  1.488354  0.179012  1   1  0.770852   A    A       0
1  0.139258  0.135126  1   1  0.347032   B    C       0
2  1.107870  1.658317  1   1  0.361021   A    A       3
3  0.366891  0.275043  1   1  0.604694   B    C       0
4 -0.700017  0.689024  1   1  0.291650   B    C       1
(200, 8)
C    139
B     31
A     30
Name: grpX, dtype: int64
      sum
grpX
A      30
B      31
C     139
grpX
A     30
B     31
C    139
Name: zz, dtype: int64
```

Out[7]:

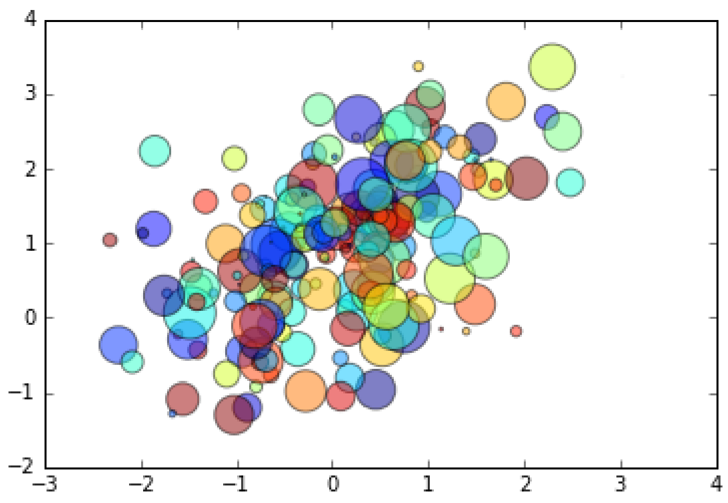
	zz
grpX	
A	30
B	31
C	139

Plotting

Scatter plot

In [8]:

```
import matplotlib.pyplot as plt
%matplotlib inline
colors = np.random.rand(len(df.x))
area = np.pi * (15 * np.random.rand(len(df.x)))**2  # 0 to 15 point radiuses
plt.scatter(df.x,df.y,s = area, c=colors, alpha=0.5)
plt.show()
```



In [9]:

```
from lightning import Lightning
lgn = Lightning(ipython=True, host='http://public.lightning-viz.org')
n=1000
#cp = [asarray(color_palette('Blues', 100)[random.choice(range(100))])*255 for i
 in range(n)]
ap = np.random.rand(n)
sp = np.random.rand(n)*15+8
lgn.scatter(df['x'],df['y'], values=df['zz'],alpha=ap,
size=sp,colormap='YlOrRd')
```

Lightning initialized

```
Connected to server at http://public.lightning-viz.org

/Users/tomas/miniconda2/envs/py27_nb/lib/python2.7/site-packages/IPy
thon/kernel/__init__.py:13: ShimWarning:

The `IPython.kernel` package has been deprecated. You should import
 from ipykernel or jupyter_client instead.
```

Out[9]:

BOKEH

Drawing

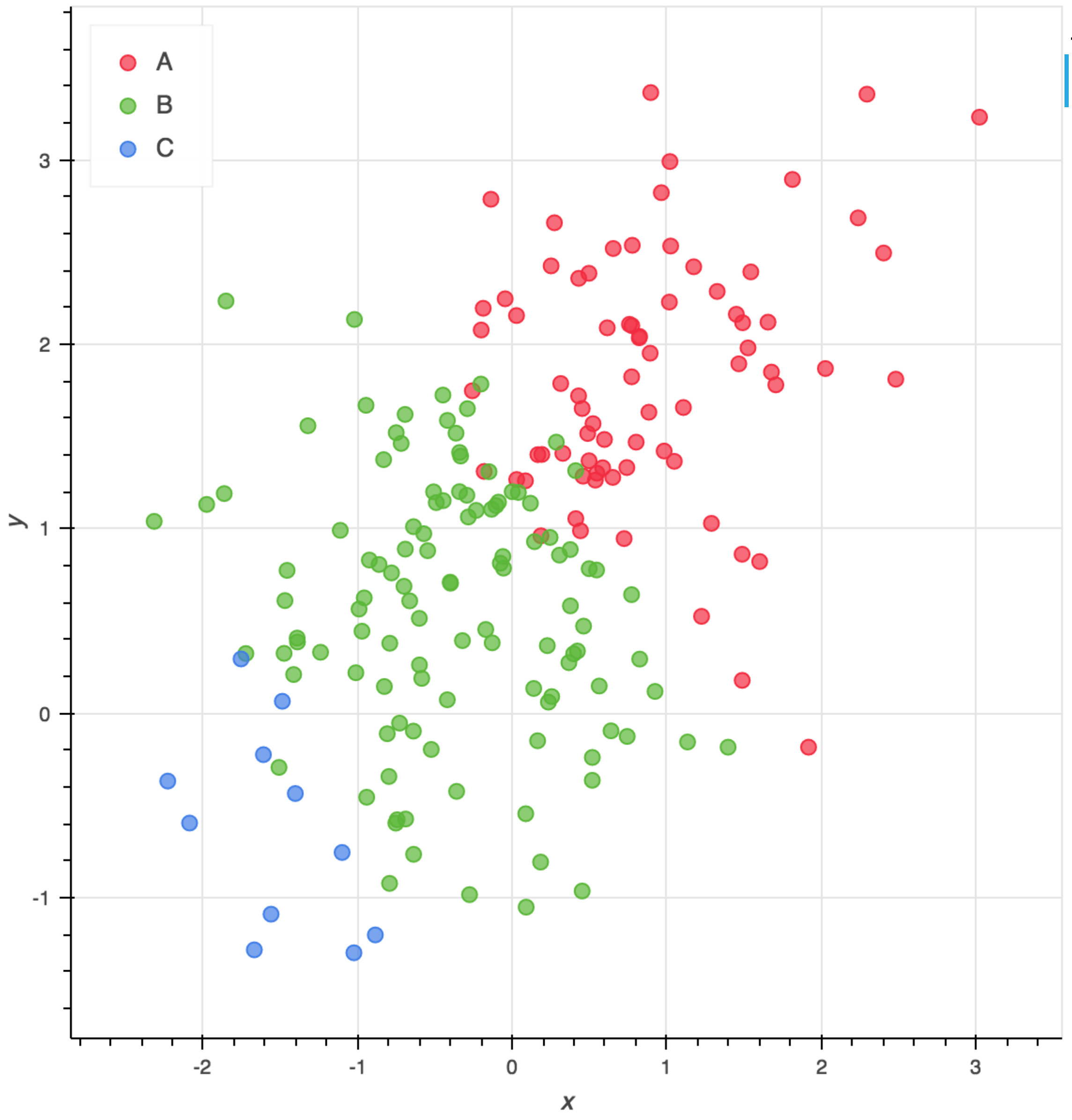
In [10]:

```
from bokeh.io import output_notebook, show
output_notebook()
# http://nbviewer.jupyter.org/github/bokeh/bokeh-notebooks/blob/master/tutorial/
01%20-%20charts.ipynb
```

(http://bokeh.pydata.org) BokehJS successfully loaded.

In [11]:

```
from bokeh.charts import Scatter

p = Scatter(df, x='x', y='y', color='grp', legend='top_left')
show(p)
```



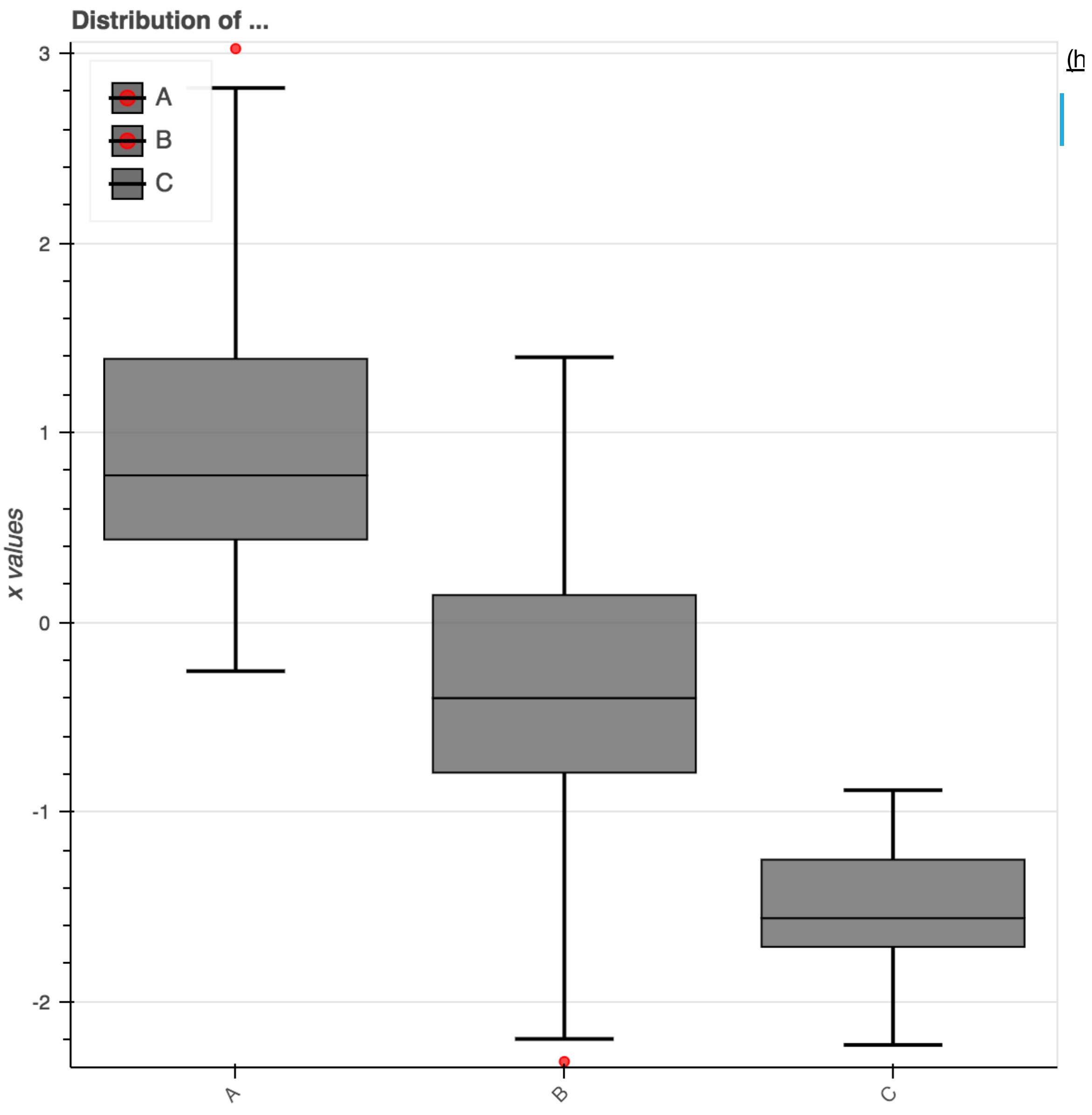
(h

In [12]:

```
# Possible tools are: box_select, box_zoom, click, crosshair,
# help, hover, lasso_select, pan, poly_select, previewsave,
# reset, resize, save, tap, wheel_zoom,
# xpan, xwheel_zoom, ypan or ywheel_zoom
TOOLS = 'pan,box_zoom,reset,'
```

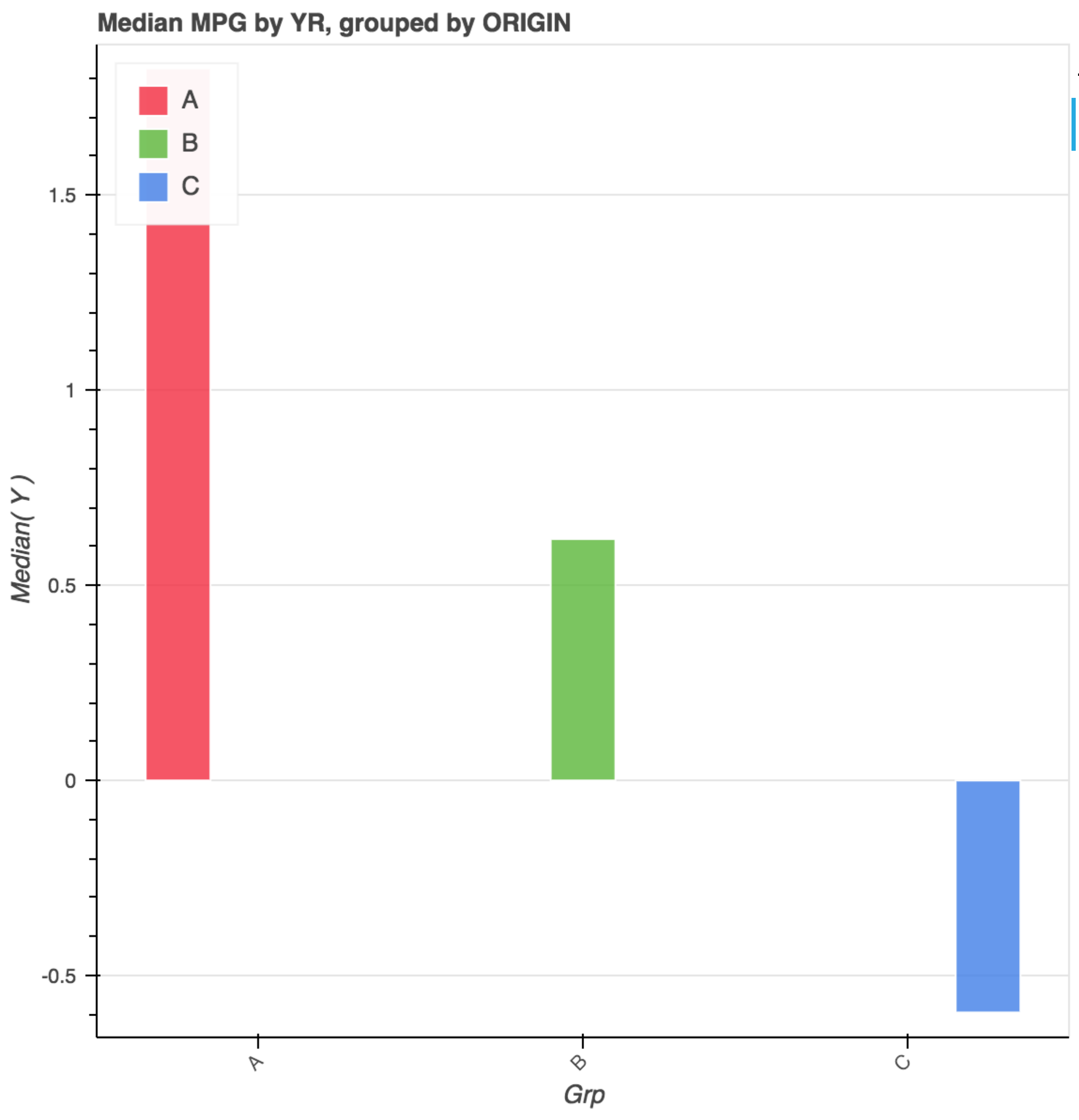
In [13]:

```
from bokeh.charts import BoxPlot
p = BoxPlot(
    df, label='grp', values='x', tools='crosshair',
    xlabel='', ylabel='x values', title='Distribution of ...'
)
show(p)
```



In [14]:

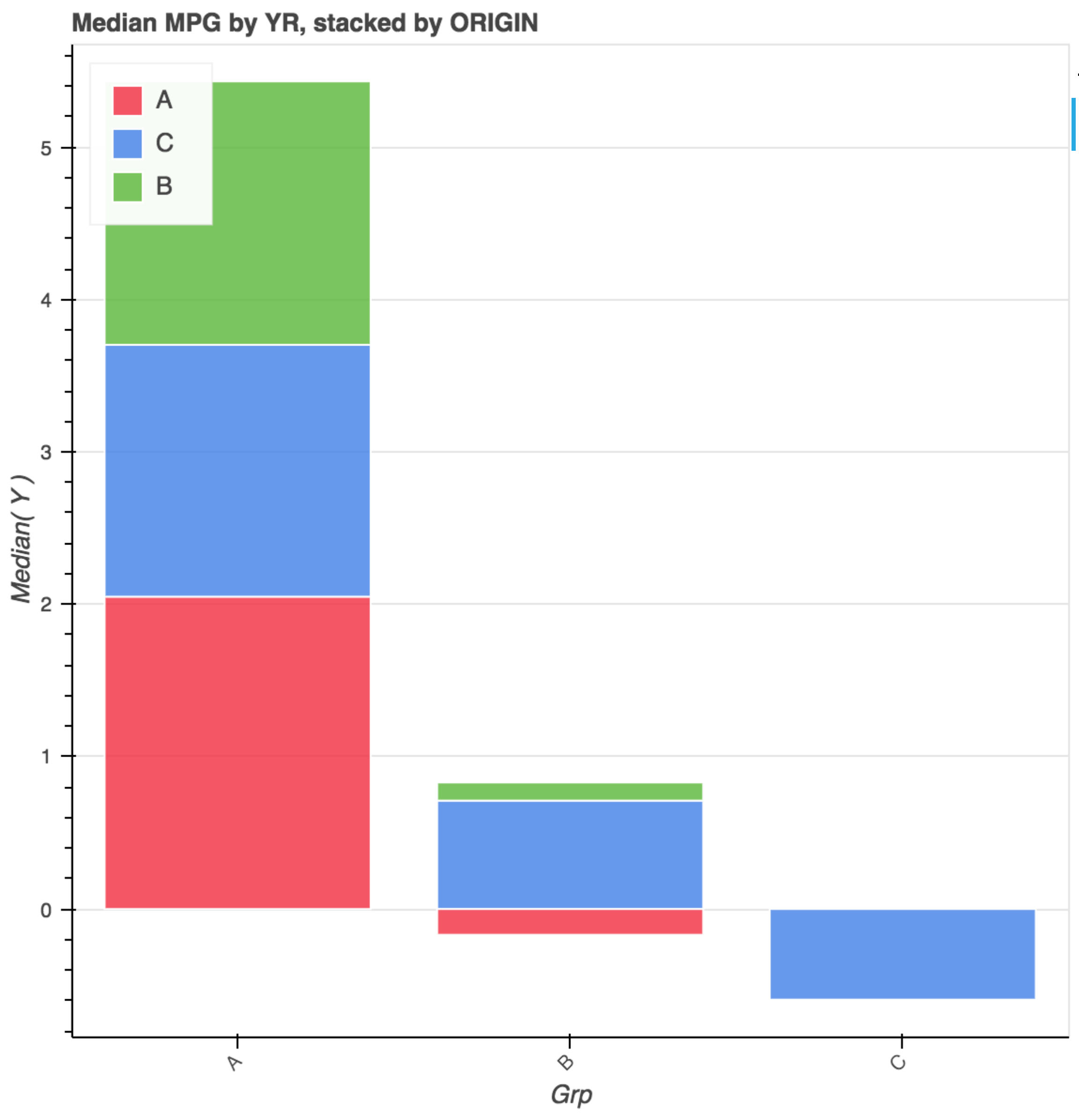
```
from bokeh.charts import Bar
p = Bar(
    df, label='grp', values='y', agg='median',
    group='grp', # Use the group feature
    title="Median MPG by YR, grouped by ORIGIN", legend='top_left', tools='cross
hair'
)
show(p)
```



(h

```
In [15]:
```

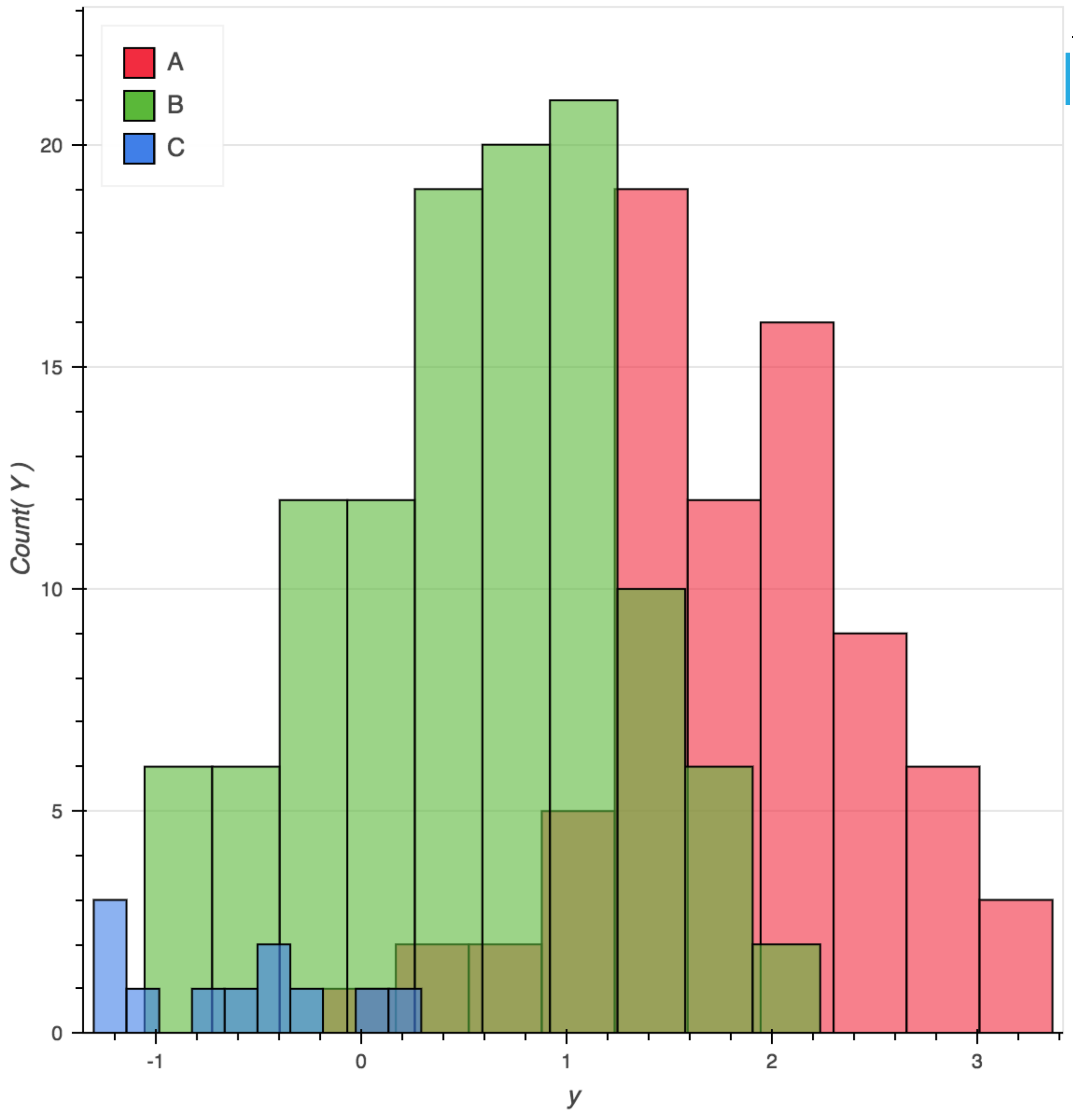
```
from bokeh.charts import Bar
p = Bar(
    df, label='grp', values='y', agg='median',
    stack='grpX', # Use the stack feature
    title="Median MPG by YR, stacked by ORIGIN", legend='top_left', tools='cross
hair'
)
show(p)
```



(h

In [16]:

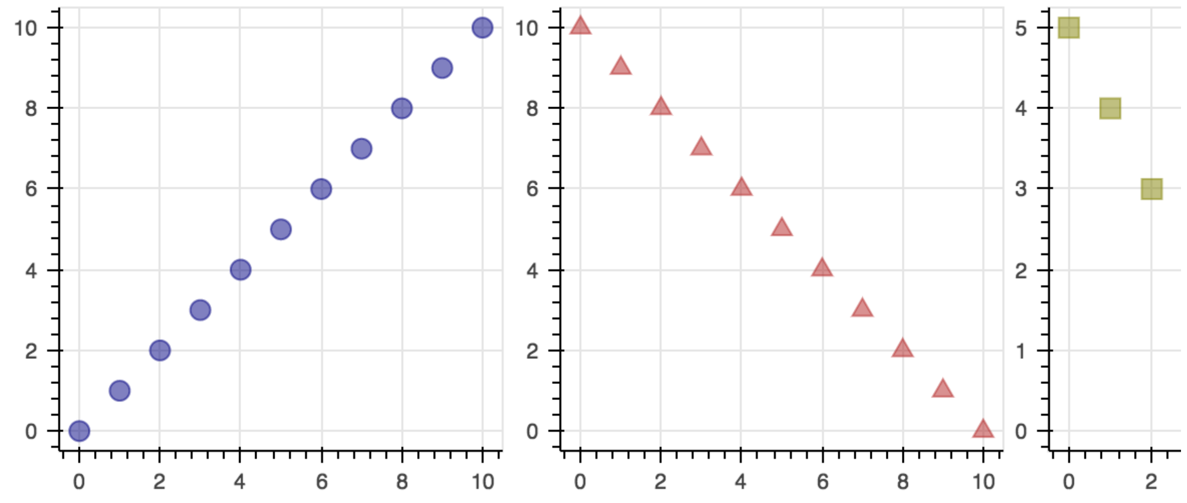
```
from bokeh.charts import Histogram
hist = Histogram(df, values='y', color='grp', bins=10, legend=True)
show(hist)
```



(h

In [17]:

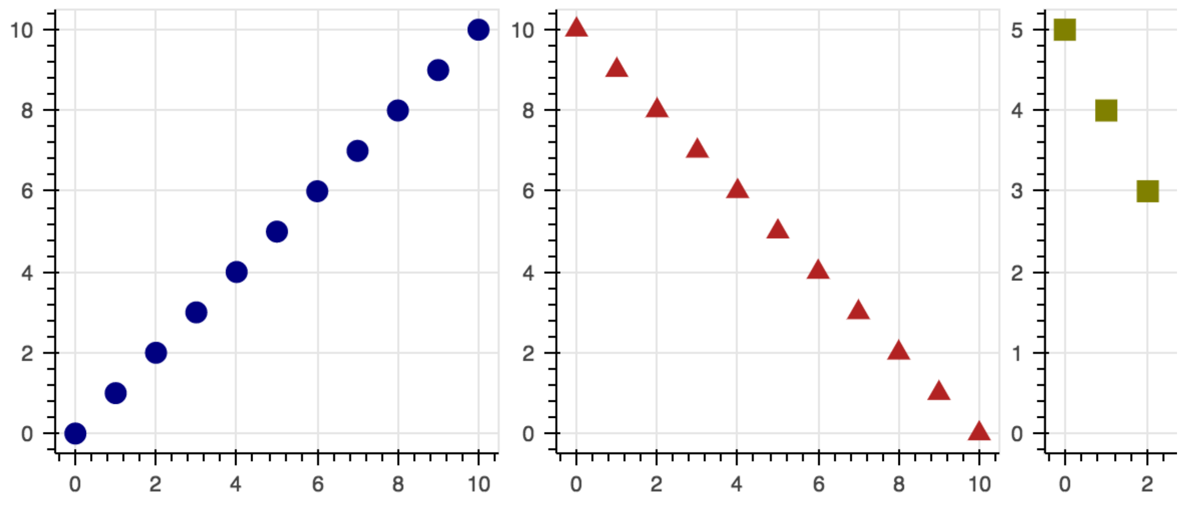
```
from bokeh.plotting import figure
from bokeh.io import gridplot

x = list(range(11))
y0, y1, y2 = x, [10-i for i in x], [abs(i-5) for i in x]

# create a new plot
s1 = figure(width=250, plot_height=250)
s1.circle(x, y0, size=10, color="navy", alpha=0.5)

# create another one
s2 = figure(width=250, height=250)
s2.triangle(x, y1, size=10, color="firebrick", alpha=0.5)

# create and another
s3 = figure(width=250, height=250)
s3.square(x, y2, size=10, color="olive", alpha=0.5)

# put all the plots in an HBox
p = gridplot([[s1, s2, s3]], toolbar_location=None)

# show the results
show(p)
```



Linked panning

In [18]:

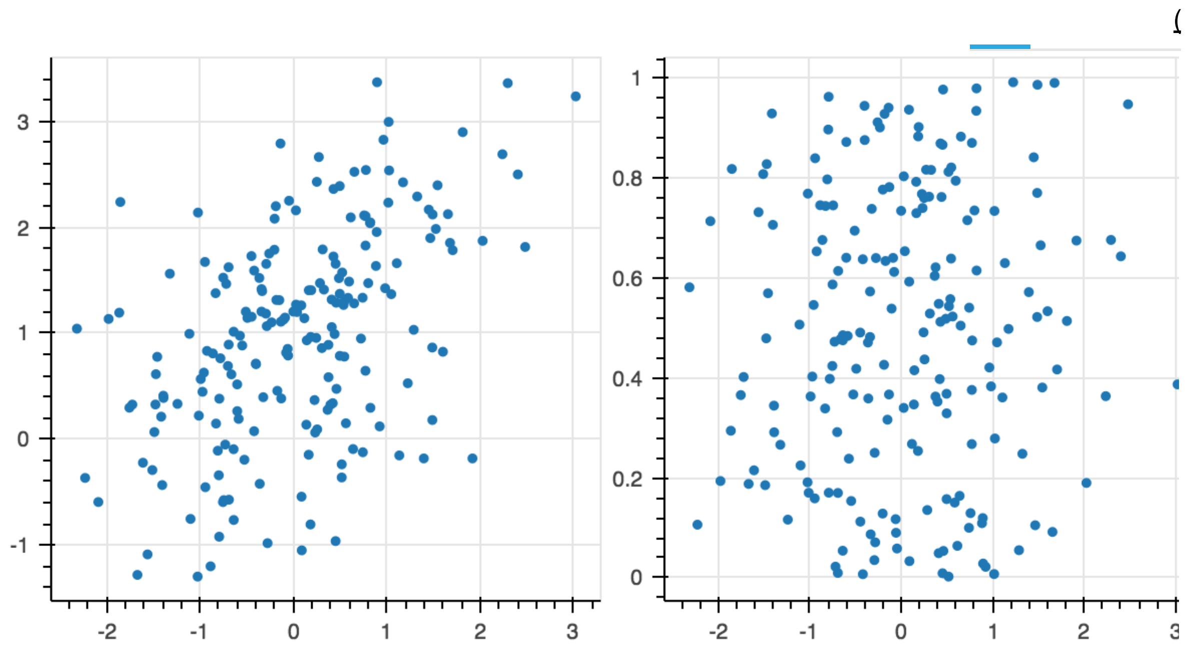
```
plot_options = dict(width=250, plot_height=250, title=None, tools='pan')

# create a new plot
s1 = figure(**plot_options)
s1.circle(x, y0, size=10, color="navy")

# create a new plot and share both ranges
s2 = figure(x_range=s1.x_range, y_range=s1.y_range, **plot_options)
s2.triangle(x, y1, size=10, color="firebrick")

# create a new plot and share only one range
s3 = figure(x_range=s1.x_range, **plot_options)
s3.square(x, y2, size=10, color="olive")

p = gridplot([[s1, s2, s3]])

# show the results
show(p)
```



Linked brushing

In [19]:

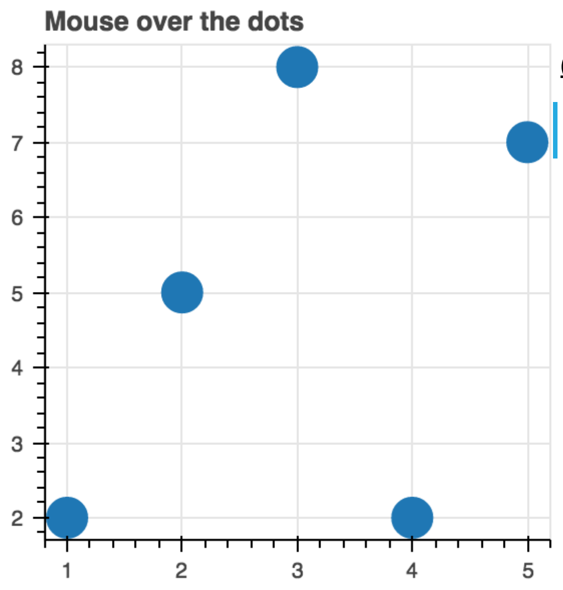
```
from bokeh.models import ColumnDataSource

#x = list(range(-20, 21))
#y0, y1 = [abs(xx) for xx in x], [xx**2 for xx in x]

x=df.x
y0=df.y
y1=df.w
# create a column data source for the plots to share
source = ColumnDataSource(data=dict(x=x, y0=y0, y1=y1))

TOOLS = "box_select,lasso_select,help"

# create a new plot and add a renderer
left = figure(tools=TOOLS, width=300, height=300)
left.circle('x', 'y0', source=source)

# create another new plot and add a renderer
right = figure(tools=TOOLS, width=300, height=300)
right.circle('x', 'y1', source=source)

p = gridplot([[left, right]])

show(p)
```



Hover Tools

In [20]:

```
from bokeh.models import HoverTool

source = ColumnDataSource(
        data=dict(
            x=[1, 2, 3, 4, 5],
            y=[2, 5, 8, 2, 7],
            desc=['A', 'b', 'C', 'd', 'E'],
        )
    )

hover = HoverTool(
        tooltips=[
            ("index", "$index"),
            ("(x,y)", "($x, $y)"),
            ("desc", "@desc"),
        ]
    )

p = figure(plot_width=300, plot_height=300, tools=[hover], title="Mouse over the
 dots")

p.circle('x', 'y', size=20, source=source)

# Also show custom hover
#from utils import get_custom_hover

#show(gridplot([[p, get_custom_hover()]]))
show(p)
```



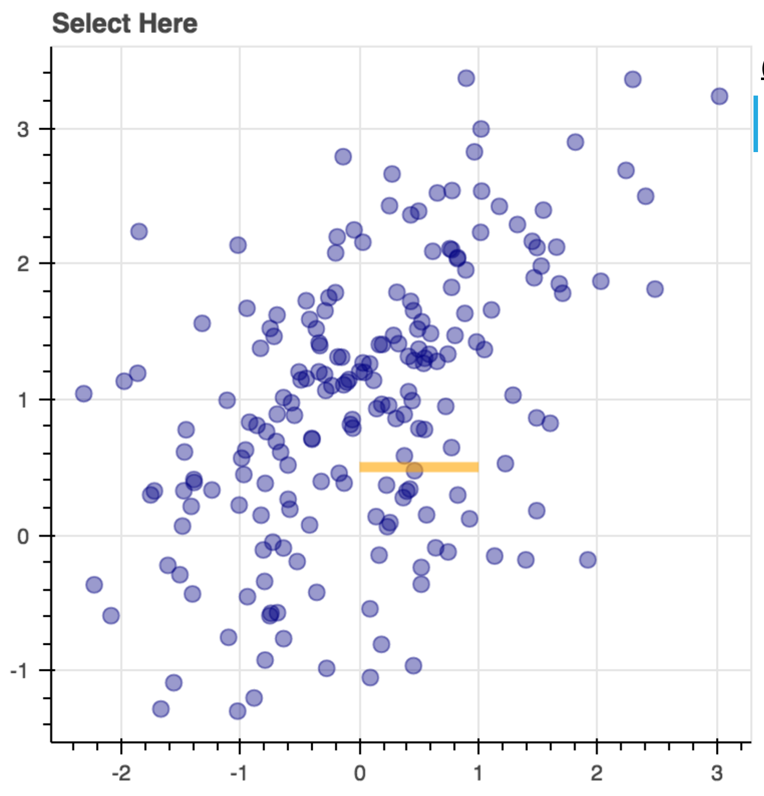
(http://bokeh.pydata.org/)

Calbacks for selections

In [21]:

```
from bokeh.models import TapTool, CustomJS, ColumnDataSource
from random import random


#x = [random() for x in range(500)]
#y = [random() for y in range(500)]
x=df.x
y=df.y
color = ["navy"] * len(x)

s = ColumnDataSource(data=dict(x=x, y=y, color=color))
p = figure(plot_width=400, plot_height=400, tools="lasso_select", title="Select
 Here")
p.circle('x', 'y', color='color', size=8, source=s, alpha=0.4)

s2 = ColumnDataSource(data=dict(ym=[0.5, 0.5]))
p.line(x=[0,1], y='ym', color="orange", line_width=5, alpha=0.6, source=s2)

s.callback = CustomJS(args=dict(s2=s2), code="""
    var inds = cb_obj.get('selected')['1d'].indices;
    var d = cb_obj.get('data');
    var ym = 0

    if (inds.length == 0) { return; }

    for (i = 0; i < d['color'].length; i++) {
        d['color'][i] = "navy"
    }
    for (i = 0; i < inds.length; i++) {
        d['color'][inds[i]] = "firebrick"
        ym += d['y'][inds[i]]
    }

    ym /= inds.length
    s2.get('data')['ym'] = [ym, ym]

    cb_obj.trigger('change');
    s2.trigger('change');
""")

show(p)
```

```
/Users/tomas/miniconda2/envs/py27_nb/lib/python2.7/site-packages/bok
eh/util/deprecation.py:34: BokehDeprecationWarning:


Supplying a user-defined data source AND iterable values to glyph me
thods is deprecated.

See https://github.com/bokeh/bokeh/issues/2056 for more information.
```



(http://bokeh.pydata.org/)