msalloum / **econ128**

Branch: **master** ▾

econ128 / Labs / Lab1 / **Lab1_PartB_Statistical_Graphs.ipynb**

Find file **Copy path**

msalloum adding labs 1, 4, 5, 6

`d160664` on Aug 8

1 contributor

```
758 lines (757 sloc)   573 KB
```

A Gallery of Statistical Graphs in Matplotlib

Inspiration for these examples was taken from http://nbviewer.ipython.org/5357268 (http://nbviewer.ipython.org/5357268)

Also see these same examples with Matplotlib defaults (http://nbviewer.ipython.org/urls/raw.github.com/cs109/content/master/lec_03_statistical_graphs_mpl_default.ipynb)

In [1]:

```
#brewer2mpl makes it easier to use color tables from colorbrewer2.org in matplotlib
!pip install brewer2mpl
```

```
Requirement already satisfied (use --upgrade to upgrade): brewer2mpl in /Users/beaumont/anaconda/l
ib/python2.7/site-packages
Cleaning up...
```

In [2]:

```
%matplotlib inline
from urllib import urlopen

import brewer2mpl
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [3]:

```
# Set up some better defaults for matplotlib
from matplotlib import rcParams

#colorbrewer2 Dark2 qualitative color table
dark2_colors = brewer2mpl.get_map('Dark2', 'Qualitative', 7).mpl_colors

rcParams['figure.figsize'] = (10, 6)
rcParams['figure.dpi'] = 150
rcParams['axes.color_cycle'] = dark2_colors
rcParams['lines.linewidth'] = 2
rcParams['axes.facecolor'] = 'white'
rcParams['font.size'] = 14
rcParams['patch.edgecolor'] = 'white'
rcParams['patch.facecolor'] = dark2_colors[0]
rcParams['font.family'] = 'StixGeneral'


def remove_border(axes=None, top=False, right=False, left=True, bottom=True):
    """
    Minimize chartjunk by stripping out unnecesasry plot borders and axis ticks

    The top/right/left/bottom keywords toggle whether the corresponding plot border is drawn
    """
    ax = axes or plt.gca()
    ax.spines['top'].set_visible(top)
    ax.spines['right'].set_visible(right)
    ax.spines['left'].set_visible(left)
    ax.spines['bottom'].set_visible(bottom)

    #turn off all ticks
    ax.yaxis.set_ticks_position('none')
    ax.xaxis.set_ticks_position('none')

    #now re-enable visibles
    if top:
        ax.xaxis.tick_top()
    if bottom:
        ax.xaxis.tick_bottom()
    if left:
        ax.yaxis.tick_left()
    if right:
        ax.yaxis.tick_right()
```

Example Data

In [4]:

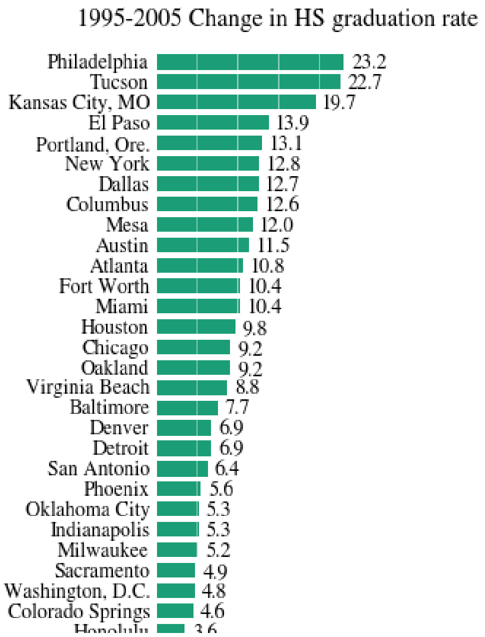
```
file = urlopen('https://raw.githubusercontent.com/vincentarelbundock/Rdatasets/master/csv/ggplot2/
diamonds.csv')
diamonds = pd.read_csv(file)

file = urlopen('http://www.columbia.edu/~cjd11/charles_dimaggio/DIRE/resources/R/titanic.csv')
titanic = pd.read_csv(file)
```

In [5]:
```
change = [23.2, 22.7, 19.7, 13.9, 13.1, 12.8, 12.7,
        12.6, 12.0, 11.5, 10.8, 10.4, 10.4, 9.8, 9.2,
        9.2, 8.8, 7.7, 6.9, 6.9, 6.4, 5.6, 5.3, 5.3, 5.2, 4.9,
        4.8, 4.6, 3.6, 3.1, 0.7, -.3, -.7, -1.2, -1.5, -1.7,
        -1.7, -1.8, -2, -2.3, -2.4, -3.6, -3.7,
        -4.9, -6.5, -6.6, -11.6, -14.8, -17.6, -23.1]
city = ['Philadelphia', 'Tucson', 'Kansas City, MO',
        'El Paso', 'Portland, Ore.', 'New York', 'Dallas',
        'Columbus', 'Mesa', 'Austin', 'Atlanta', 'Fort Worth',
        'Miami', 'Houston', 'Chicago', 'Oakland', 'Virginia Beach',
        'Baltimore', 'Denver', 'Detroit', 'San Antonio', 'Phoenix',
        'Oklahoma City', 'Indianapolis', 'Milwaukee', 'Sacramento',
        'Washington, D.C.', 'Colorado Springs', 'Honolulu', 'Nashville',
        'Jacksonville', 'Louisville', 'Seattle',
        'Memphis', 'Fresno', 'Boston', 'Mineappolis',
        'San Jose', 'Tulsa', 'Charlotte', 'San Diego', 'Los Angeles',
        'Long Beach', 'Cleveland', 'San Francisco', 'Albuquerque',
        'Arlington, TX', 'Omaha', 'Wichita', 'Las Vegas']

grad = pd.DataFrame({'change' : change, 'city': city})
```

Bar Chart

In [6]:
```
plt.figure(figsize=(3, 8))

change = grad.change[grad.change > 0]
city = grad.city[grad.change > 0]
pos = np.arange(len(change))

plt.title('1995-2005 Change in HS graduation rate')
plt.barh(pos, change)

#add the numbers to the side of each bar
for p, c, ch in zip(pos, city, change):
    plt.annotate(str(ch), xy=(ch + 1, p + .5), va='center')

#cutomize ticks
ticks = plt.yticks(pos + .5, city)
xt = plt.xticks()[0]
plt.xticks(xt, [' '] * len(xt))

#minimize chartjunk
remove_border(left=False, bottom=False)
plt.grid(axis = 'x', color ='white', linestyle='-')

#set plot limits
plt.ylim(pos.max() + 1, pos.min() - 1)
plt.xlim(0, 30)
```

Out[6]:
```
(0, 30)
```

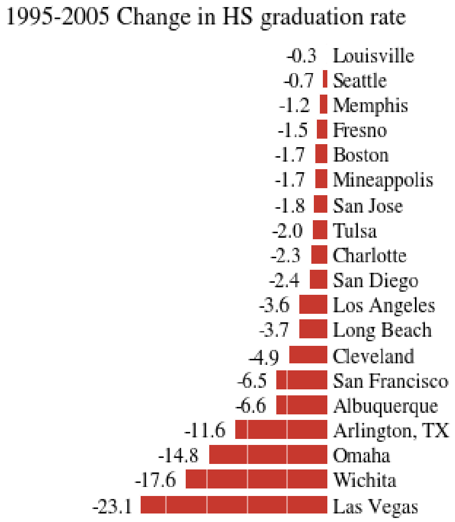


In [7]:

```
change = grad.change[grad.change < 0].values
city = grad.city[grad.change < 0].values

pos = np.arange(len(change))
red = (0.78, 0.22, 0.18) # RGB triplet

plt.figure(figsize=(3, 6), dpi=200)
plt.barh(pos, change, color=red)
plt.yticks(pos + .5, city)

#add the numbers to the side of each bar
for p, c, ch in zip(pos, city, change):
    plt.annotate(str(ch), xy=(ch - 1, p + .5), va='center', ha='right')

#cutomize ticks
plt.gca().yaxis.tick_right()
ticks = plt.yticks(pos + .5, city)
xt = plt.xticks()[0]
plt.xticks(xt, [' '] * len(xt))

#Remove chartjunk
remove_border(left=False, bottom=False)
plt.grid(axis = 'x', color ='white', linestyle='-')

plt.ylim(pos.max() + 1, pos.min()- .5)
plt.xlim(-30, 0)
plt.title('1995-2005 Change in HS graduation rate')
```

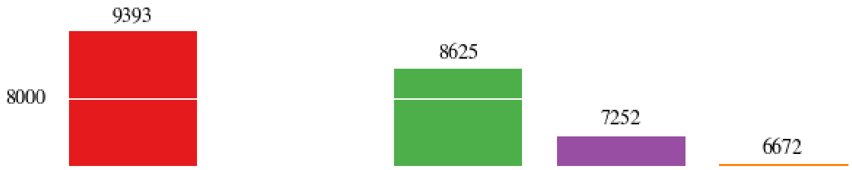
Out[7]:

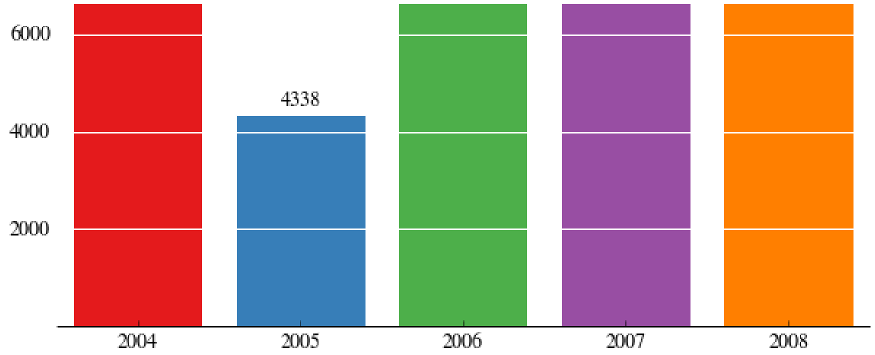
```
<matplotlib.text.Text at 0x10b3b9290>
```



In [8]:

```
years = np.arange(2004, 2009)
heights = np.random.random(years.shape) * 7000 + 3000

box_colors = brewer2mpl.get_map('Set1', 'qualitative', 5).mpl_colors

plt.bar(years - .4, heights, color=box_colors)
plt.grid(axis='y', color='white', linestyle='-', lw=1)
plt.yticks([2000, 4000, 6000, 8000])

fmt = plt.ScalarFormatter(useOffset=False)
plt.gca().xaxis.set_major_formatter(fmt)
plt.xlim(2003.5, 2008.5)
remove_border(left=False)

for x, y in zip(years, heights):
    plt.annotate("%i" % y, (x, y + 200), ha='center')
```



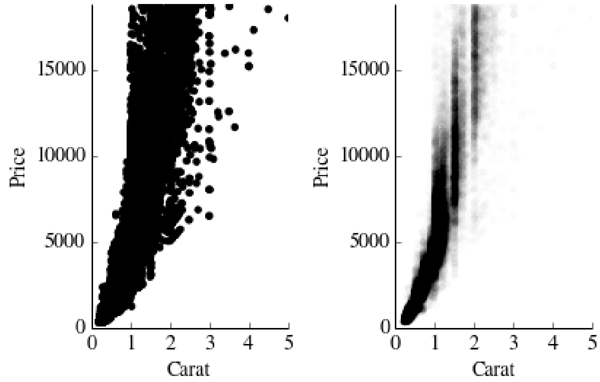


Dot Plots

Scatterplots

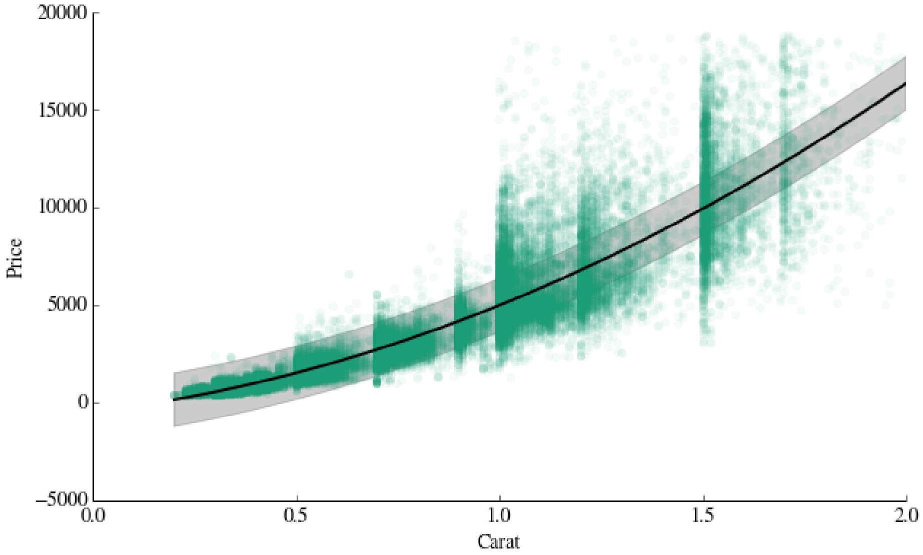
In [9]:
```
plt.figure(tight_layout=True, figsize=(6, 4))
plt.subplot(121)
plt.scatter(diamonds.carat, diamonds.price, color='k')
plt.ylim(0, diamonds.price.max())
plt.xlim(0, 5)
plt.xlabel("Carat")
plt.ylabel("Price")
remove_border()

plt.subplot(122)
plt.scatter(diamonds.carat, diamonds.price, color='k', alpha=.01)
plt.ylim(0, diamonds.price.max())
plt.xlim(0, 5)

plt.xlabel("Carat")
plt.ylabel("Price")
remove_border()
```

```
/Users/beaumont/anaconda/lib/python2.7/site-packages/matplotlib/figure.py:1595: UserWarning: This
 figure includes Axes that are not compatible with tight_layout, so its results might be incorrec
t.
  warnings.warn("This figure includes Axes that are not "
```



Trend Lines

In [10]:
```
# the raw data
x = diamonds.carat[diamonds.carat < 2]
y = diamonds.price[diamonds.carat < 2]
plt.plot(x, y, 'o', mec='none', alpha=.05)

#fit and overplot a 2nd order polynomial
params = np.polyfit(x, y, 2)
xp = np.linspace(x.min(), 2, 20)
yp = np.polyval(params, xp)
plt.plot(xp, yp, 'k')

#overplot an error band
sig = np.std(y - np.polyval(params, x))
plt.fill_between(xp, yp - sig, yp + sig,
plt.fill_between(xp, yp - sig, yp + sig,
                 color='k', alpha=0.2)

plt.xlabel("Carat")
plt.ylabel("Price")
plt.xlim(0, 2)
remove_border()
```



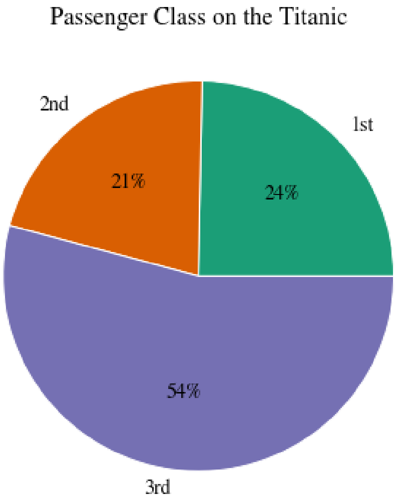
Bubble Charts

Pie Charts

In [11]:
```
t = titanic.groupby(['pclass']).size()
print t

plt.subplot(aspect=True)
plt.pie(t, labels=t.index.values, colors = dark2_colors[0:3], autopct='%i%%')
plt.title("Passenger Class on the Titanic")
```

```
pclass
1st       323
2nd       277
3rd       709
dtype: int64
```

Out[11]:
```
<matplotlib.text.Text at 0x10b3cf7d0>
```



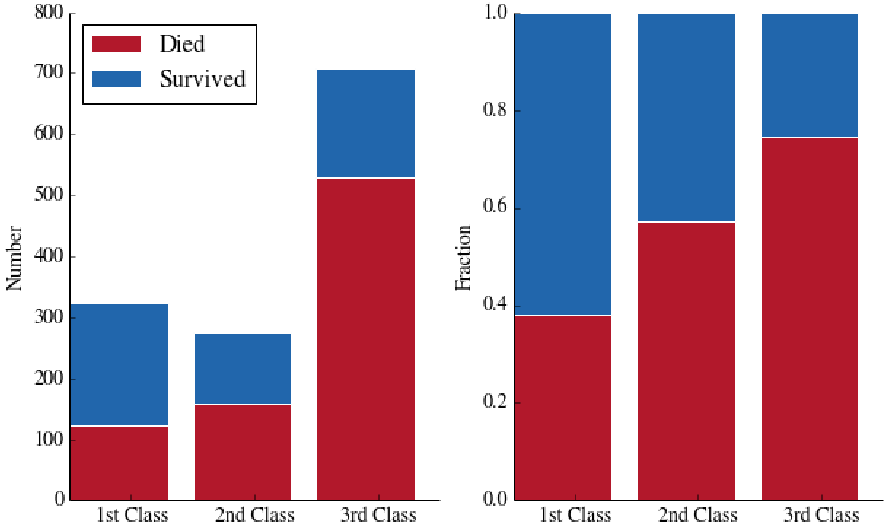
Donut Charts

Stacked Bar Chart

In [12]:

```
tclass = titanic.groupby(['pclass', 'survived']).size().unstack()
print tclass

red, blue = '#B2182B', '#2166AC'

plt.subplot(121)
plt.bar([0, 1, 2], tclass[0], color=red, label='Died')
plt.bar([0, 1, 2], tclass[1], bottom=tclass[0], color=blue, label='Survived')
plt.xticks([0.5, 1.5, 2.5], ['1st Class', '2nd Class', '3rd Class'], rotation='horizontal')
plt.ylabel("Number")
plt.xlabel("")
plt.legend(loc='upper left')
remove_border()

#normalize each row by transposing, normalizing each column, and un-transposing
tclass = (1. * tclass.T / tclass.T.sum()).T

plt.subplot(122)
plt.bar([0, 1, 2], tclass[0], color=red, label='Died')
plt.bar([0, 1, 2], tclass[1], bottom=tclass[0], color=blue, label='Survived')
plt.xticks([0.5, 1.5, 2.5], ['1st Class', '2nd Class', '3rd Class'], rotation='horizontal')
plt.ylabel("Fraction")
plt.xlabel("")
remove_border()

plt.show()
```

```
survived    0    1
pclass
1st       123  200
2nd       158  119
3rd       528  181
```



Small Multiples

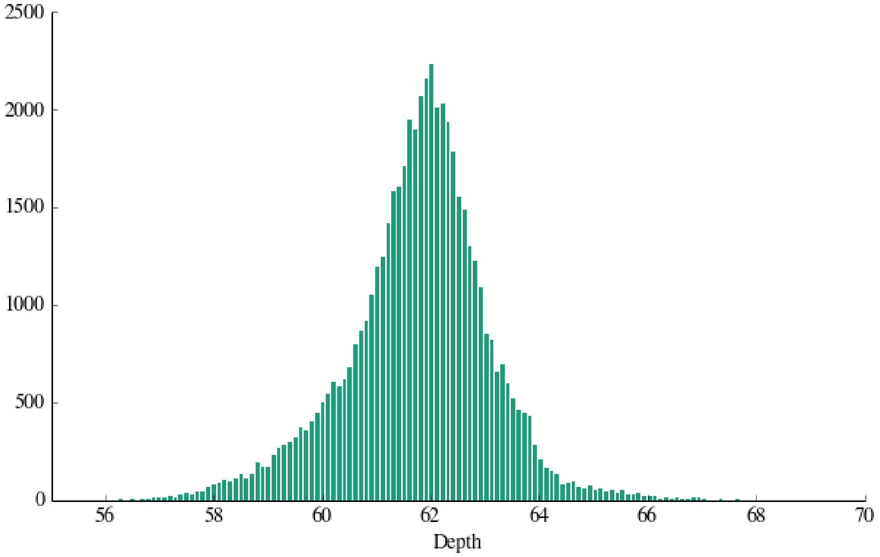
Waterfall Chart

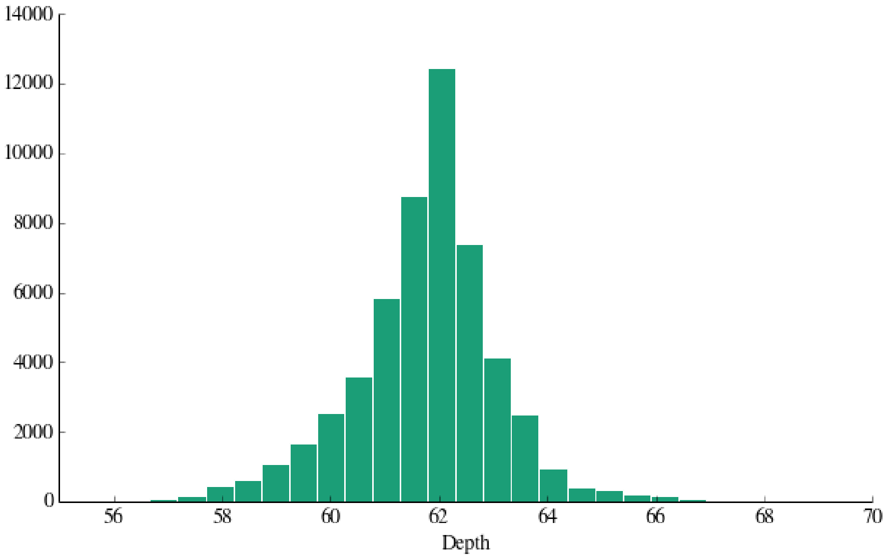
Stacked Area Chart

Histogram

In [13]:

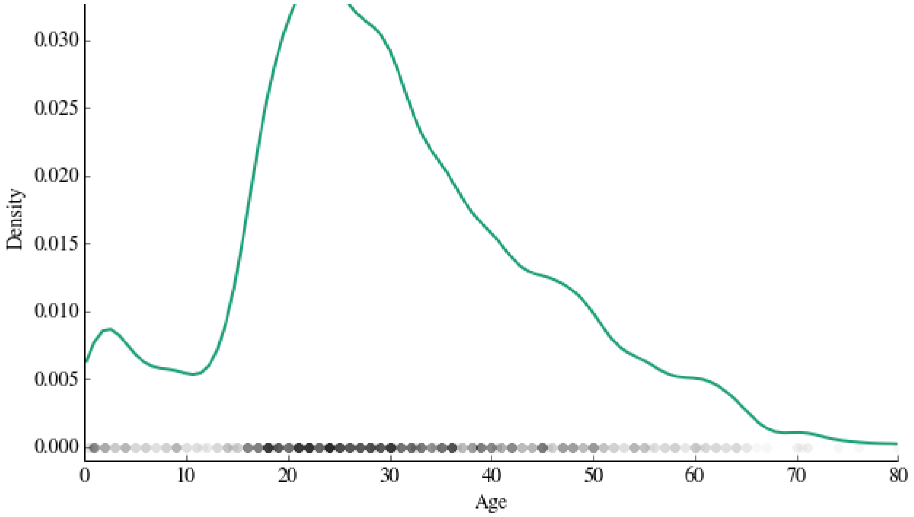
```
plt.hist(diamonds.depth, bins=np.linspace(50, 70, 200))
plt.xlabel("Depth")
remove_border()
plt.xlim(55, 70)
plt.show()
plt.hist(diamonds.depth, bins=np.linspace(50, 70, 40))
plt.xlabel("Depth")
remove_border()
plt.xlim(55, 70)
plt.show()
```





Density Plots

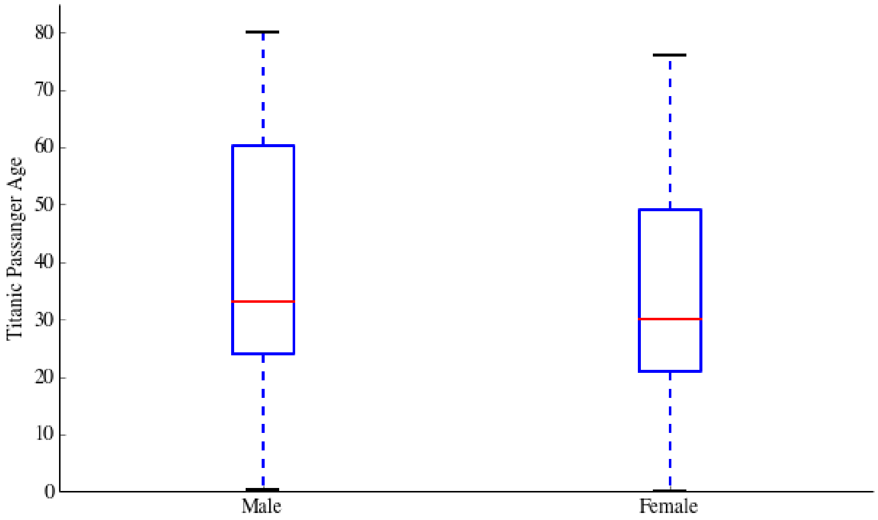
In [14]:
```
#KernelDensity objects estimate the (log of the) density of points
#see http://scikit-learn.org/stable/modules/density.html
from sklearn.neighbors.kde import KernelDensity


age = titanic.age.dropna().values  # drop missing values, turn to normal numpy array
age = age.reshape(-1, 1)  # scikit-learn expects data matrices of shape [ndata, ndim]

kde = KernelDensity(bandwidth=2).fit(age)
x = np.linspace(age.min(), age.max(), 100).reshape(-1, 1)
density = np.exp(kde.score_samples(x))

plt.plot(x, density)
plt.plot(age, age * 0, 'ok', alpha=.03)
plt.ylim(-.001, .035)

plt.xlabel("Age")
plt.ylabel("Density")
remove_border()
```



Box and Whisker Plots

In [15]:

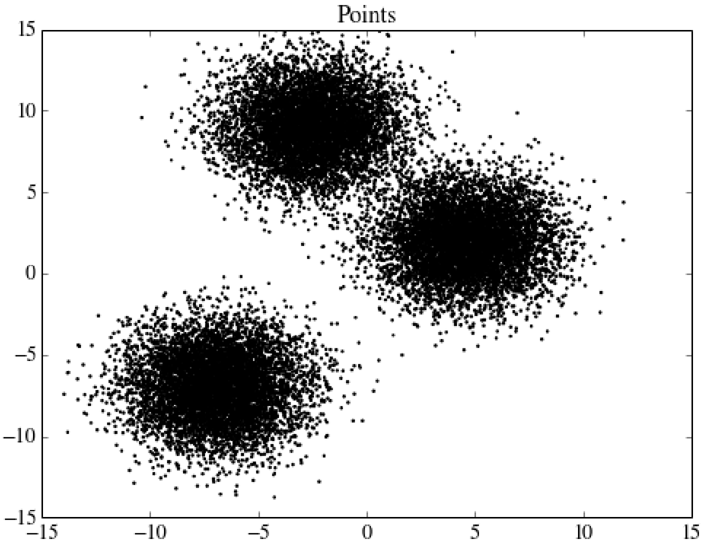
```
male_age = titanic.age[titanic.sex == 'male']
female_age = titanic.age[titanic.sex == 'female']

plt.boxplot([male_age, female_age])
plt.ylabel("Titanic Passanger Age")
plt.xticks([1, 2], ["Male", "Female"])
plt.ylim(0, 85)
remove_border()
```

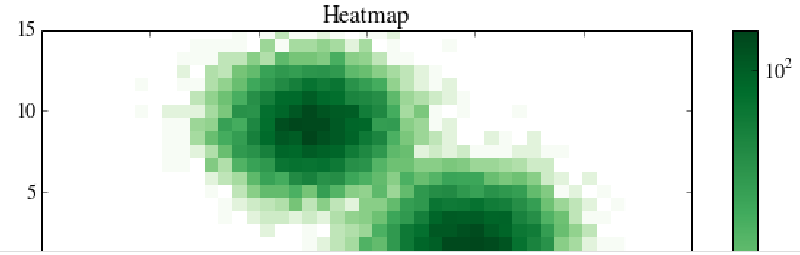


Heat Maps (2D Density Plots)

In [16]:

```
from sklearn.datasets import make_blobs
from matplotlib.colors import LogNorm

X, _ = make_blobs(n_samples=20000, centers=3, random_state=42, cluster_std=2)

plt.scatter(X[:, 0], X[:, 1], 2, color='k')
plt.title("Points")
plt.xlim(-15, 15)
plt.ylim(-15, 15)
plt.gca().set_position([.125, .125, .62, .775])
plt.show()

plt.hist2d(X[:, 0], X[:, 1], bins=40, cmap='Greens', norm=LogNorm())
ax = plt.gca()
plt.title("Heatmap")
plt.colorbar()
plt.xlim(-15, 15)
plt.ylim(-15, 15)
plt.show()
plt.show()
```


Points


Heatmap
10^2