1 </br>

Introduction

In [27]:
```
import sys
print(sys.version)

import pandas as pd
import numpy as np
print('numpy version: ', np.__version__) # 1.15.4
print('pandas version: ', pd.__version__) # 0.23.4


import bokeh
from bokeh.io import output_notebook
import holoviews as hv
hv.extension('bokeh')

print('bokeh version: ', bokeh.__version__) # 1.0.1
print('holoviews version: ', hv.__version__) # 1.10.9

import hvplot.pandas
print('hvplot version: ', hvplot.__version__) # 0.2.1

import sklearn
print('sklearn version: ', sklearn.__version__) # 0.20.1
```

```
3.7.1 | packaged by conda-forge | (default, Nov 13 2018, 19:01:41) [MSC v.1900 64 bit (AMD64)]
numpy version:  1.15.4
pandas version:  0.23.4
```

```
bokeh version:  1.0.1
holoviews version:  1.10.9
hvplot version:  0.2.1
sklearn version:  0.20.1
```

In [28]:
```
import pathlib
import feather

from holoviews.operation.timeseries import rolling
from bokeh.themes.theme import Theme
from bokeh.plotting import figure, show
```

In [29]:
```
DATA_DIR = pathlib.Path.cwd() / 'data'

THEME_ONE = Theme(
    json={
        'attrs' : {
            'Axis': {
                'minor_tick_line_color': 'white',
                'axis_label_text_font': 'arial',
                'axis_label_text_font_style': 'normal',
            },
            'Title': {
                'align':'center',
            }
        }
})
```

page 2

In [30]:
```
df_wages = feather.read_dataframe(DATA_DIR / 'Wage.feather')

print('shape:', df_wages.shape)
print(df_wages.info())
```

```
shape: (3000, 11)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3000 entries, 0 to 2999
Data columns (total 11 columns):
year          3000 non-null int32
age           3000 non-null int32
maritl        3000 non-null category
race          3000 non-null category
education     3000 non-null category
region        3000 non-null category
jobclass      3000 non-null category
health        3000 non-null category
health_ins    3000 non-null category
logwage       3000 non-null float64
wage          3000 non-null float64
dtypes: category(7), float64(2), int32(2)
memory usage: 92.1 KB
None
```

In [31]:
```
%%opts Scatter [width=300 height=400 ] (size=3 color='lightgray')
hv.renderer('bokeh').theme = THEME_ONE

## Wage/Age
tbl_age_wage = hv.Table(df_wages, [('age','Age')],  [('wage','Wage')])
grouped = df_wages[['age','wage']].sort_values(['age','wage']).groupby('age').mean()
rc = rolling(hv.Curve(grouped), rolling_window=15).options(color='blue')
## Wage/Year
tbl_wage_year = hv.Table(df_wages, [('year','Year')],  [('wage','Wage')])
# no smoothing, below is a little jagged
wage_year_curve = hv.Curve(df_wages[['year','wage']].sort_values(['year','wage']).groupby('year').mean()).options(color
='blue')
# from 2003 mean to 2009 mean
wage_range = df_wages[['year','wage']].groupby(['year']).mean().agg({'wage': ['min', 'max']}).T
wage_year_points = [(df_wages.year.min(), wage_range['min'][0]) , (df_wages.year.max(), wage_range['max'][0])]
wage_year_line = hv.Curve(wage_year_points).options(color='blue')
## Wage/Edu boxplot
df_wages['edu'] = df_wages['education'].str[:1]
wage_edu_box = (hv.BoxWhisker(df_wages.sort_values(['edu']), [('edu', 'Education Level')], ('wage', 'Wage'))
                .options(color_index='edu', box_color=hv.Cycle('Set1'), outlier_fill_alpha=0, whisker_line_dash='dashe
d'))

left_range = df_wages.year.min() * .9999
right_range = df_wages.year.max() * 1.0001
(tbl_age_wage.to.scatter().options(fill_alpha=0)
    * rc + tbl_wage_year.to.scatter().redim.range(Year=(left_range, right_range)).options(fill_alpha=0)
    * wage_year_line + wage_edu_box)
```

Out[31]:

(https

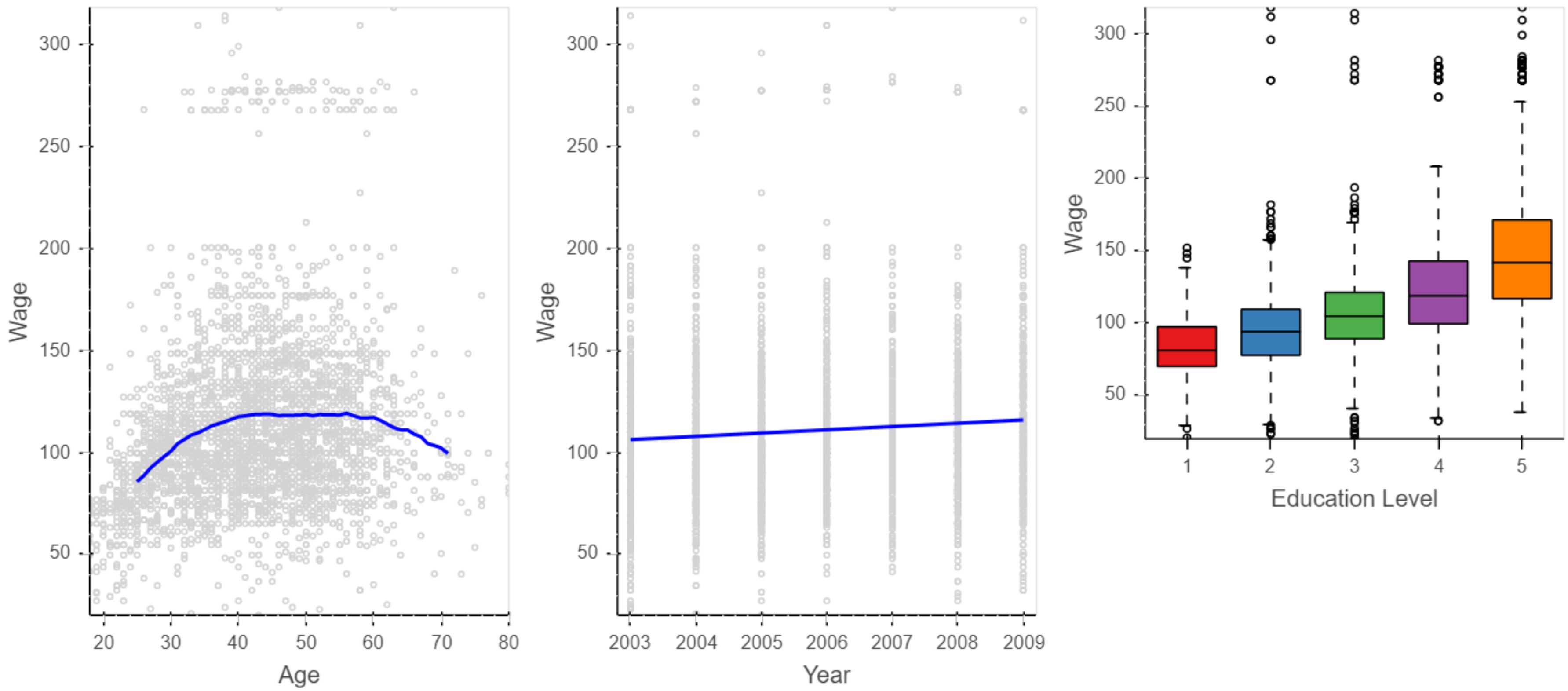


FIGURE 1.1 *`Wage` data, which contains income survey information for males from the central Atlantic region of the United States. Left: `wage` as a function of `age`. On average, `wage` increases with `age` until about 60 years of age, at which point it begins to decline. Center: `wage` as a function of `year`. There is a slow but steady increase of approximately $10,000 in the average `wage` between 2003 and 2009. Right: Boxplots displaying `wage` as a function of `education`, with 1 indicating the lowest level (no high school diploma) and 5 the highest level (an advanced graduate degree). On average, `wage` increases with the level of `education`.*

Figure 1.2 Notes:

- skip any particular set of x/y tick values, w/bokeh can zoom in & tick values "auto-update"

page 3

In [32]:
```
df_smarket = feather.read_dataframe(DATA_DIR / 'Smarket.feather')
print('shape:', df_smarket.shape)
print(df_smarket.info())
```

```
shape: (1250, 9)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1250 entries, 0 to 1249
Data columns (total 9 columns):
Year         1250 non-null float64
Lag1         1250 non-null float64
Lag2         1250 non-null float64
Lag3         1250 non-null float64
Lag4         1250 non-null float64
Lag5         1250 non-null float64
Volume       1250 non-null float64
Today        1250 non-null float64
Direction    1250 non-null category
dtypes: category(1), float64(8)
memory usage: 79.5 KB
None
```

In [33]:
```
%%opts BoxWhisker [width=300 height=400 yrotation=90 fontsize={'title': '14pt', 'ylabel': '15px', 'xlabel': '17px'} ]
hv.renderer('bokeh').theme = THEME_ONE

y_label = 'Percentage change in S&P'
df_smarket_sorted = df_smarket.sort_values(['Direction'])
boxes = []
for lag, label in [('Lag1', 'Yesterday'),('Lag2', 'Two Days Previous'),('Lag3', 'Three Days Previous')]:
    boxes.append(hv.BoxWhisker(df_smarket_sorted, [('Direction', "Today's Direction")], (lag, y_label), label=label,)
                 .options(box_color=hv.Cycle('Category10'), outlier_fill_alpha=0, box_line_width=1.5, whisker_line_dash
='dashed'))

boxes[0] + boxes[1] + boxes[2]
```

Out[33]:

(https

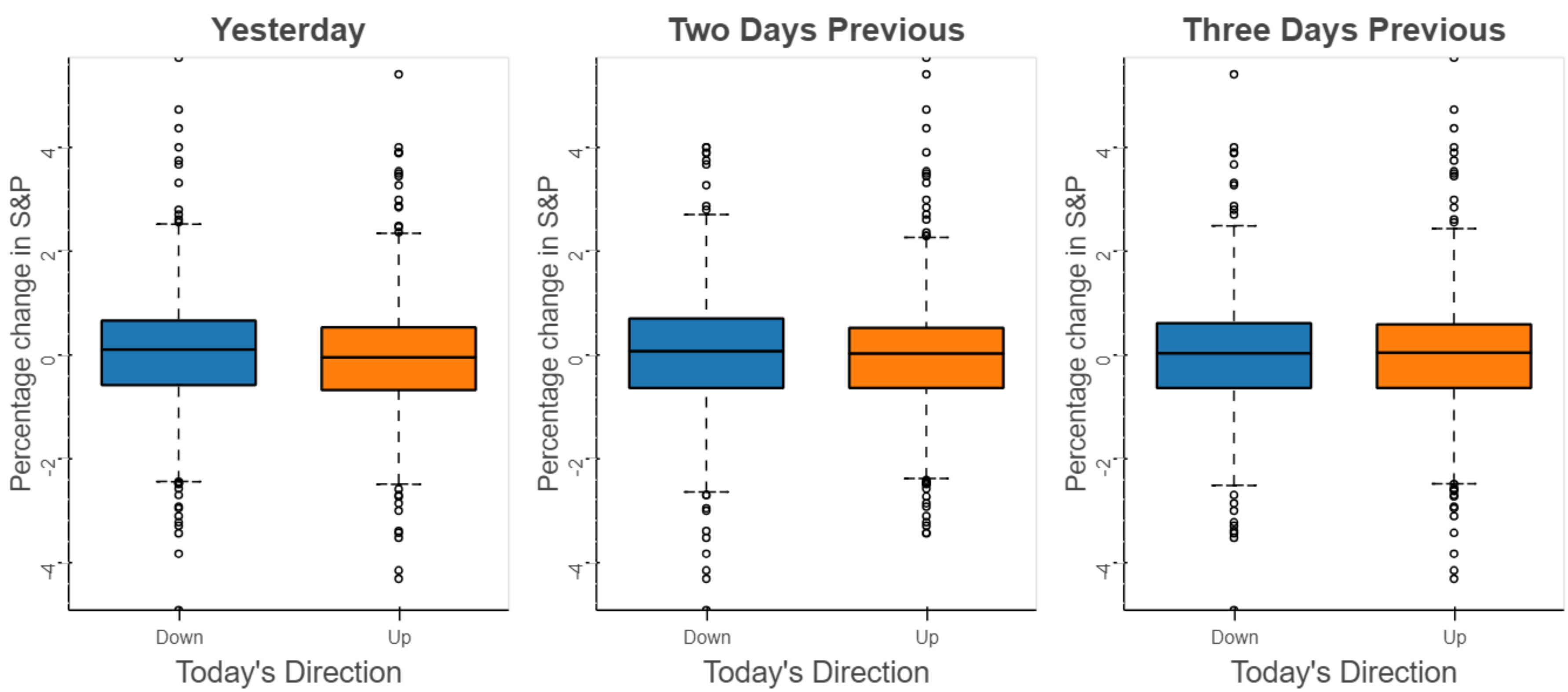


FIGURE 1.2 *Left: Boxplots of the previous day's percentage change in the S&P index for the days for which the market increased or decreased, obtained from the `Smarket` data.* Center and Right: *Same as left panel, but the percentage changes for 2 and 3 days previous are shown*

page 4

In [34]:

```
# implement R code from page 178 in py
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis

df_smarket['Direction'] = df_smarket['Direction'].astype('category')
year_filter = df_smarket['Year'] < 2005
train = df_smarket[year_filter]
smarket_2005 = df_smarket[~year_filter]
print('test shape', smarket_2005.shape)

x_cols = ['Lag1','Lag2']
y_col = 'Direction'
qda_sm = QuadraticDiscriminantAnalysis()
qda_sm.fit(train[x_cols], train[y_col])
qda_class = qda_sm.predict(smarket_2005[x_cols])
pd.crosstab(qda_class, smarket_2005.Direction, rownames=['qda_class'])
probs = qda_sm.predict_proba(smarket_2005[x_cols])
np.mean(qda_class == smarket_2005[y_col],)

df_probs = pd.DataFrame(probs, columns=['Down', 'Up'])

# pivot columns to rows
df_probs_box = df_probs.stack().to_frame().reset_index(level=1)
df_probs_box.columns=['direction', 'value']
df_probs_box.head()
```

```
test shape (252, 9)
```

Out[34]:

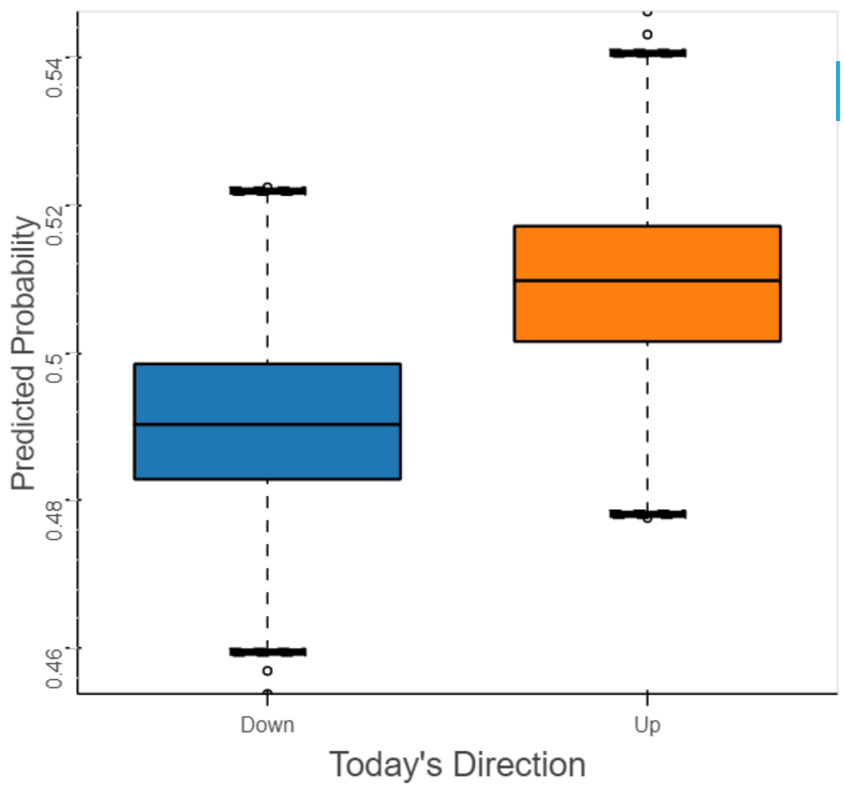
	direction	value
0	Down	0.487324
0	Up	0.512676
1	Down	0.475901
1	Up	0.524099
2	Down	0.463691

In [35]:
```
%%opts BoxWhisker [width=450 height=400 yrotation=90 fontsize={ 'title': '14pt', 'ylabel': '15px', 'xlabel': '17px'}]
hv.renderer('bokeh').theme = THEME_ONE

y_label = 'Predicted Probability'
box = (hv.BoxWhisker(df_probs_box
                     ,kdims= [('direction', "Today's Direction")]
                     ,vdims=[('value', 'Predicted Probability')]
                     ).options(color_index='direction'
                               ,box_color=hv.Cycle('Category10')
                               ,outlier_fill_alpha=0
                               ,box_line_width=1.5
                               ,whisker_line_dash='dashed')
)

box
```

Out[35]:



(htt|

FIGURE 1.3 *We fit a quadratic discriminant analysis model to the subset of the `Smarket` data corresponding to the 2001–2004 time period, and predicted the probability of a stock market decrease using the 2005 data. On average, the predicted probability of decrease is higher for the days in which the market does decrease. Based on these results, we are able to correctly predict the direction of movement in the market 60% of the time.*

Figure 1.3 Notes:

- the above doesn't so much resemble Figure 1.3 from ISLR, though the output generated by `QuadraticDiscriminantAnalysis` in `sklearn` above does essentially match that of R from chapter 4 (and the boxplot generated from that R *does* match above `BoxWhisker`).
  - **TODO**: investigate possible reasons for discrepancies.

---

page 5

In [36]:
```
df_nci60 = feather.read_dataframe(DATA_DIR / 'NCI60.feather')
print('shape', df_nci60.shape)
nci_labs = df_nci60.labs
nci_data = df_nci60[df_nci60.columns[:-1]].values
print(df_nci60.info())
df_nci60.head()
```

```
shape (64, 6831)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 64 entries, 0 to 63
Columns: 6831 entries, data.1 to labs
dtypes: category(1), float64(6830)
memory usage: 3.3 MB
None
```

Out[36]:

	data.1	data.2	data.3	data.4	data.5	data.6	data.7	data.8	data.9	data.10	...	data.6822	data.682
0	0.300000	1.180000	0.550000	1.140000	-0.265000	-7.000000e-02	0.350000	-0.315000	-0.450000	-0.654980	...	0.000000	0.030000
1	0.679961	1.289961	0.169961	0.379961	0.464961	5.799610e-01	0.699961	0.724961	-0.040039	-0.285019	...	-0.300039	-0.25003
2	0.940000	-0.040000	-0.170000	-0.040000	-0.605000	0.000000e+00	0.090000	0.645000	0.430000	0.475019	...	0.120000	-0.74000
3	0.280000	-0.310000	0.680000	-0.810000	0.625000	-1.387779e-17	0.170000	0.245000	0.020000	0.095019	...	-0.110000	-0.16000
4	0.485000	-0.465000	0.395000	0.905000	0.200000	-5.000000e-03	0.085000	0.110000	0.235000	1.490019	...	-0.775000	-0.51500

5 rows × 6831 columns

In [37]:
```
from sklearn.decomposition import PCA
from sklearn.preprocessing import scale
from sklearn.cluster import AgglomerativeClustering # hierarchical clustering

# tried a number of diff cluster algo/param combos
nci_data = scale(nci_data)
pca = PCA(n_components=4)
pca.fit(nci_data)
out = pca.fit_transform(nci_data) * (1,-1,1,1)
```

In [38]:
```
%%opts Scatter [width=450 height=400 show_legend=False yrotation=90 ] (size=8)

hclust = AgglomerativeClustering(n_clusters=4, linkage='ward').fit(nci_data)
df_hclust = pd.DataFrame(np.hstack([out, hclust.labels_[:,np.newaxis]]), columns=['x','y','a','b','label'])
df_hclust['label'] = df_hclust.label.astype(int).astype(str)

# extents defined as (xmin, ymin, xmax, ymax).
scatter_clusters = hv.Scatter(df_hclust, extents=(-60, -65, 75, 30)).options(color_index='label',
                                                                    cmap={'0':'#EB282B', '1':'#29479E', '2':'#377FC0', '3':'#4EB96
6'})

out_with_types = np.append(out, np.array(df_nci60.labs.values).reshape(out.shape[0],1), axis=1)
df_types = pd.DataFrame(out_with_types, columns=['x','y','a','b','label'])
scatter_types = hv.Scatter(df_types,  extents=(-60, -65, 65, 30)).options(color_index='label', cmap='Category20')

scatter_clusters.redim.label(x='Z1', y='Z2') + scatter_types.redim.label(x='Z1', y='Z2')
```

Out[38]:

(https

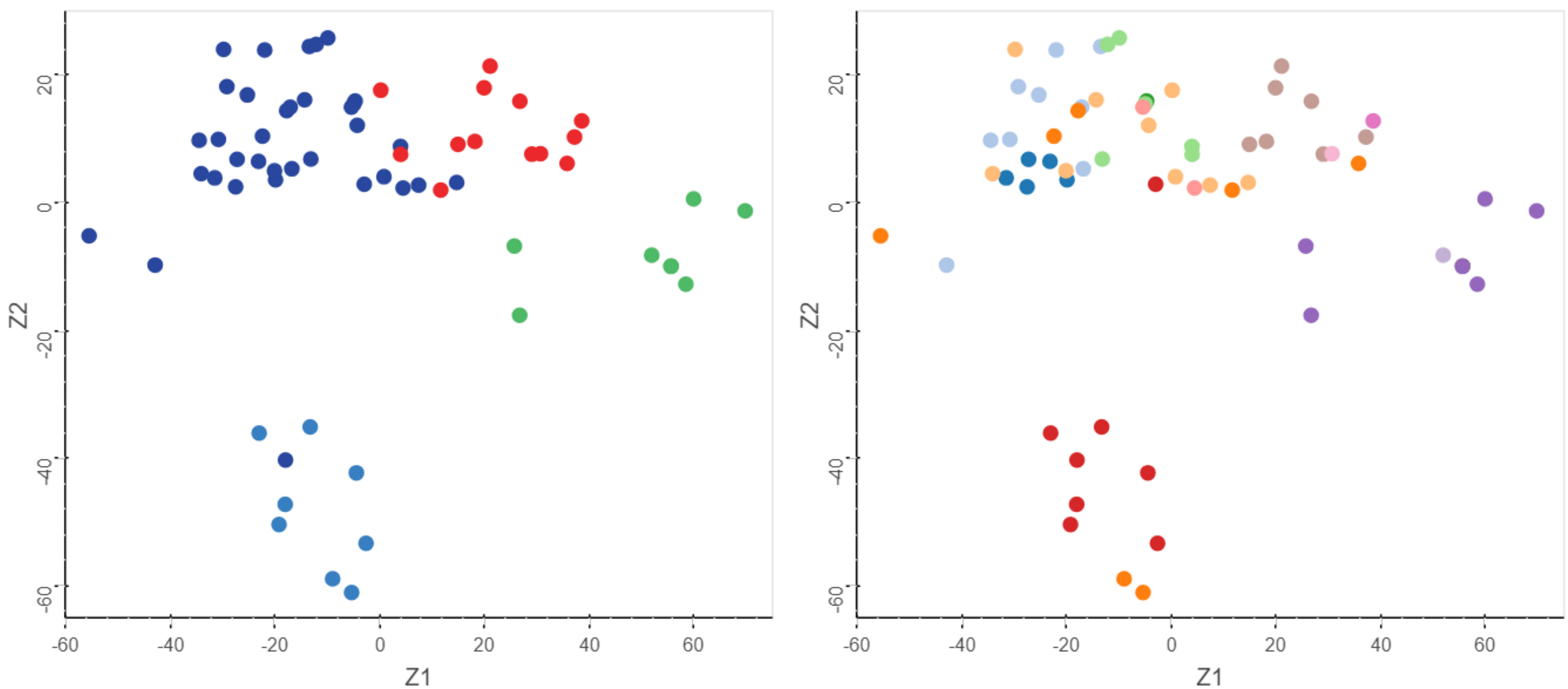


FIGURE 1.4 Left: *Representation of the `NCI60` gene expression data set in a two-dimensional space, `Z1` and `Z2`. Each point corresponds to one of the 64 cell lines. There appear to be four groups of cell lines, which we have represented using different colors.* Right: *Same as left panel except that we have represented each of the 14 different types of cancer using a different colored symbol. Cell lines corresponding to the same cancer type tend to be nearby in the two-dimensional space.*

Figure 1.4 Notes:

- Left plot
  - tried a number of diff approaches, with various settings and unable to reproduce the exact same four clusters as from ISLR
- Right plot
  - even less luck here, unable ot find a way of setting marker shapes tied to underlying data with current Holoviews/bokec
  - see `bokeh` scatter plot **below** instead

In [39]:

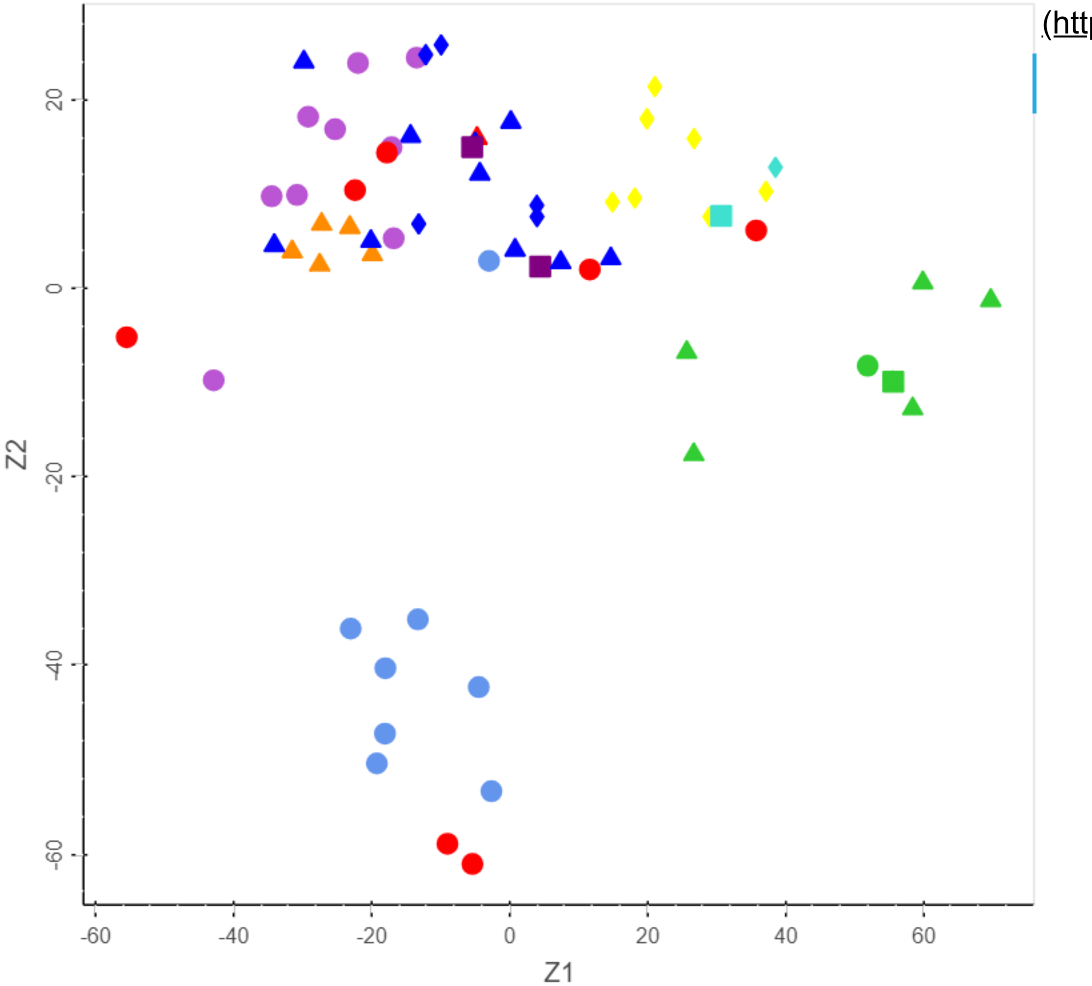
```
pf = figure(plot_width=550, plot_height=500)
df_types = pd.DataFrame(out_with_types, columns=['x','y','a','b','label'])

d3 = {'CNS': ['triangle', 'darkorange'],
     'RENAL': ['circle', 'mediumorchid'],
     'BREAST': ['circle', 'red'],
     'NSCLC': ['triangle', 'blue'],
     'MELANOMA': ['circle', 'cornflowerblue'],
     'PROSTATE': ['square', 'purple'],
     'UNKNOWN': ['triangle', 'red'],
     'OVARIAN': ['diamond', 'blue'],
     'LEUKEMIA': ['triangle', 'limegreen'],
     'K562B-repro': ['circle', 'limegreen'],
     'K562A-repro': ['square', 'limegreen'],
     'COLON': ['diamond', 'yellow'],
     'MCF7A-repro': ['diamond', 'turquoise'],
     'MCF7D-repro': ['square', 'turquoise'],
}

df_types['marker'] = df_types.label.apply(lambda x: d3[x][0])
df_types['color'] = df_types.label.apply(lambda x: d3[x][1])

pf.xaxis.axis_label = 'Z1'
pf.yaxis.axis_label = 'Z2'
pf.xgrid.grid_line_color = None
pf.ygrid.grid_line_color = None
pf.yaxis.major_label_orientation = 'vertical'
pf.scatter(x=df_types.x.values, y=df_types.y.values, size=10,
           marker=list(df_types.marker.values),
           fill_color=list(df_types.color.values),
           line_color=list(df_types.color.values))

show(pf)
```



In [ ]: