DSC 640: Weeks 11 – 12

Author: Yousof Rahimian

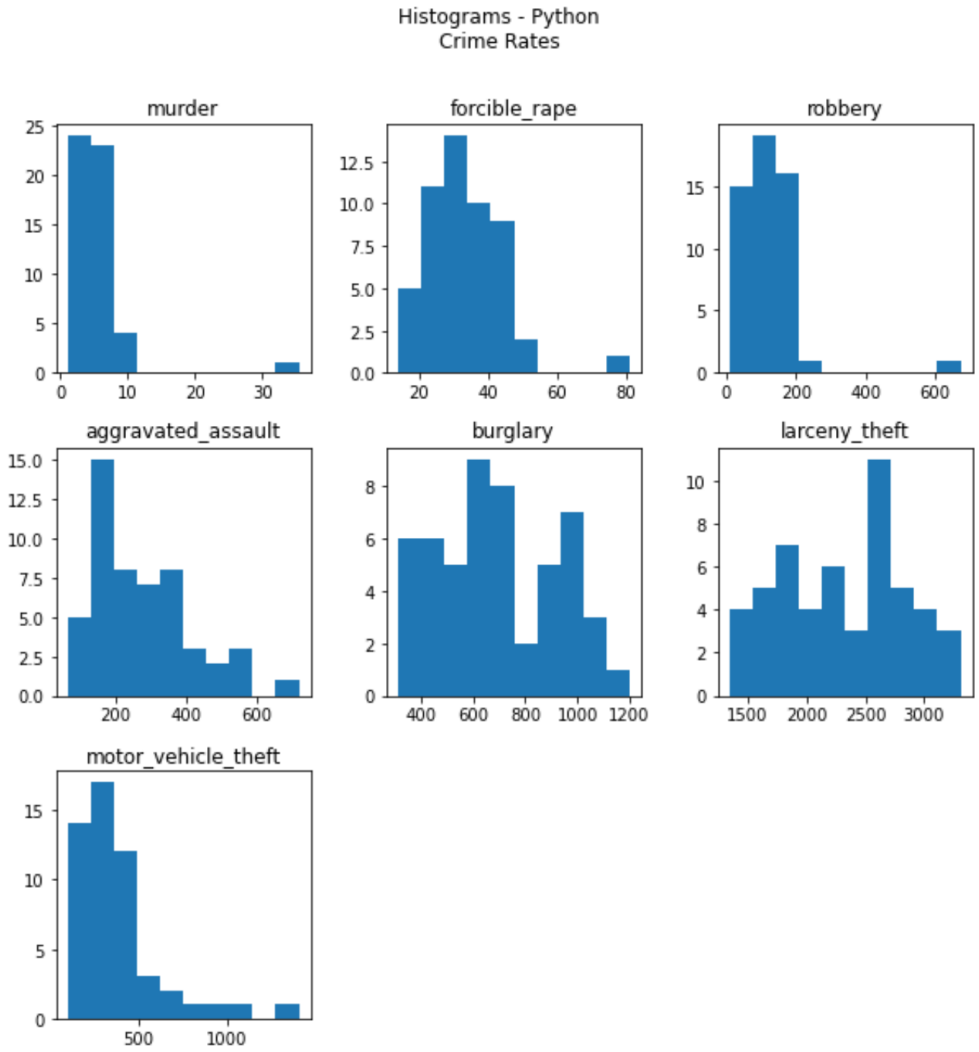
Date: March 3, 2023

Exercise 6.2 Histograms, Box Plots & Bullet Charts

Histograms

Python

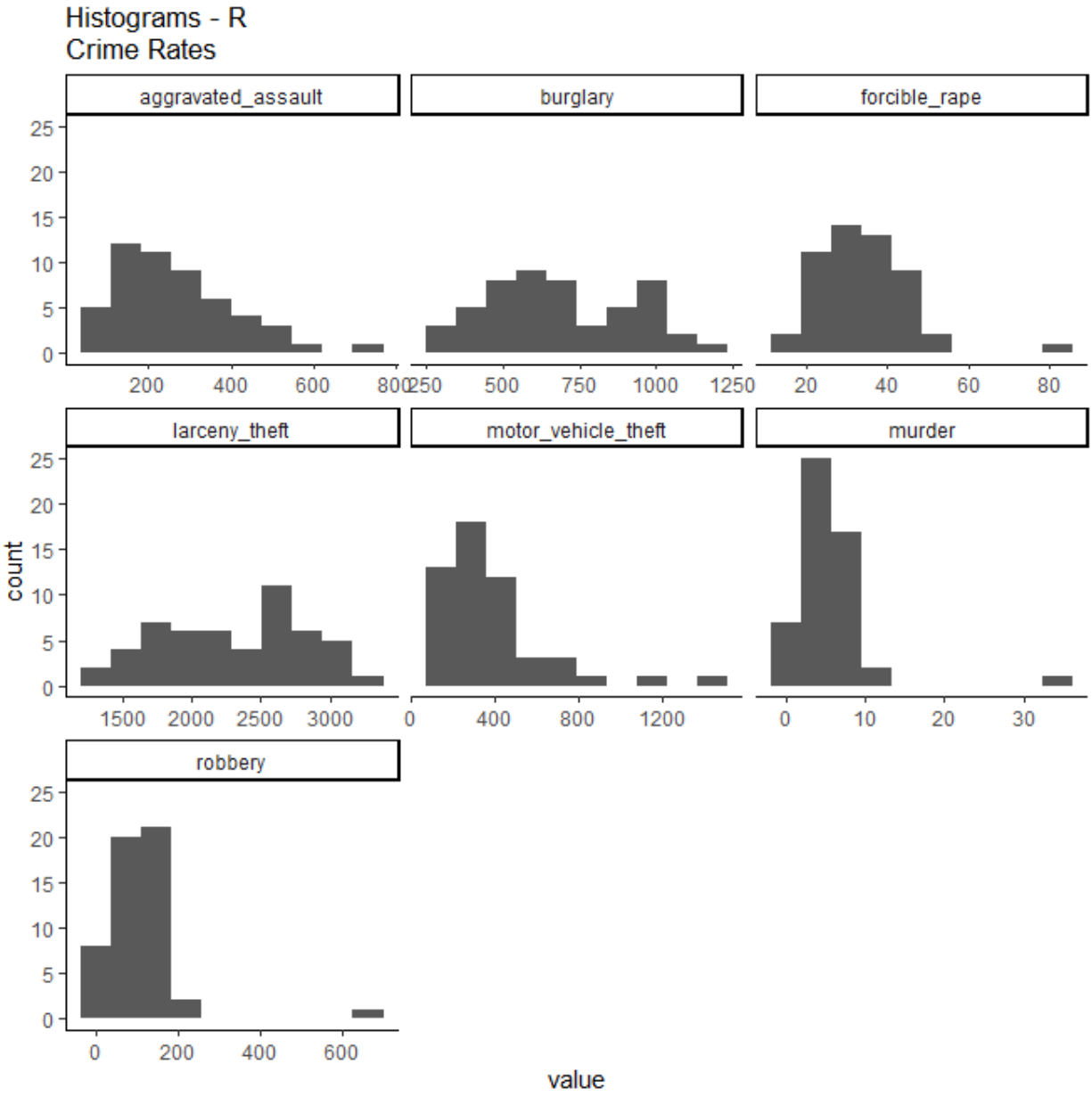
```
axes = crimerate_df.hist(bins = 10, figsize = (10,10), grid = False)
plt.suptitle("Histograms - Python\nCrime Rates")
```



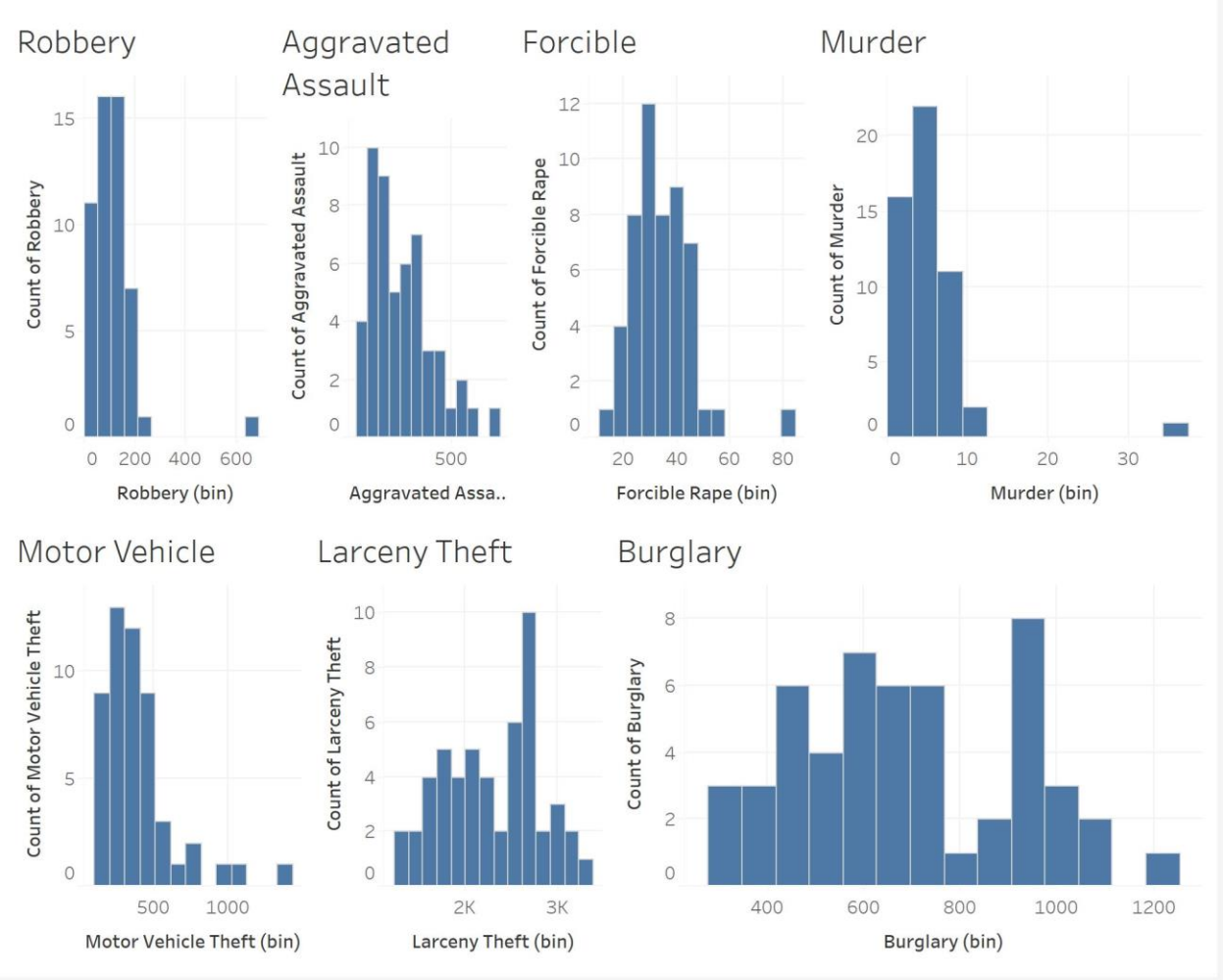
R

```{r}
#| label: justcrimes
just_crimes_df <- subset(crime_df, select = -c(state) )
just_crimes_df
```

```{r}
#| label: histogram
fig <- ggplot(gather(just_crimes_df), aes(value)) +
  geom_histogram(bins = 10) +
  facet_wrap(~key, scales = 'free_x') +
  ggtitle("Histograms - R\nCrime Rates")
fig
```



Tableau



Box Plots

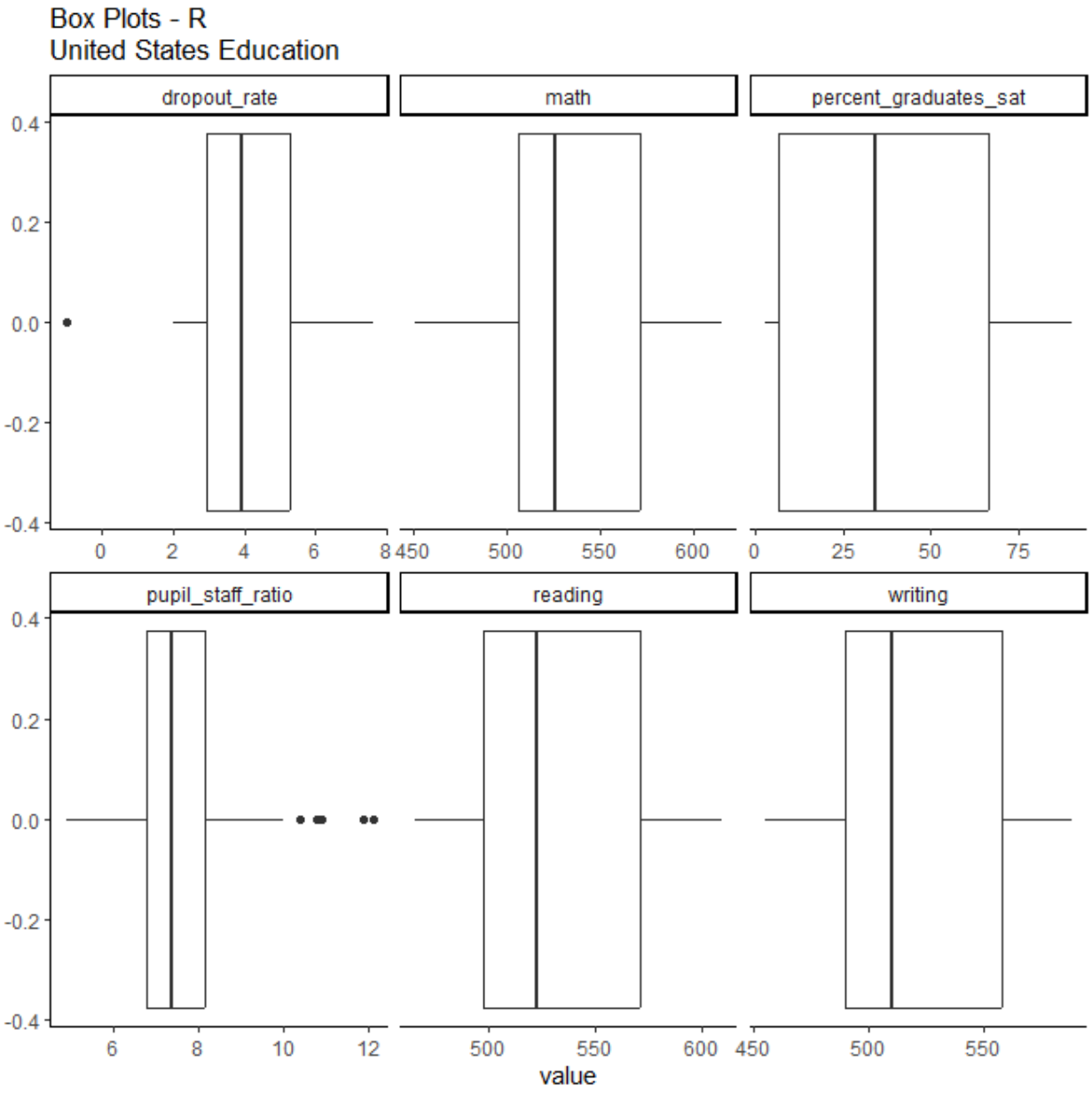
Python

```
states_education = education_df[education_df['state'] !='United States']
states_education.head()
```

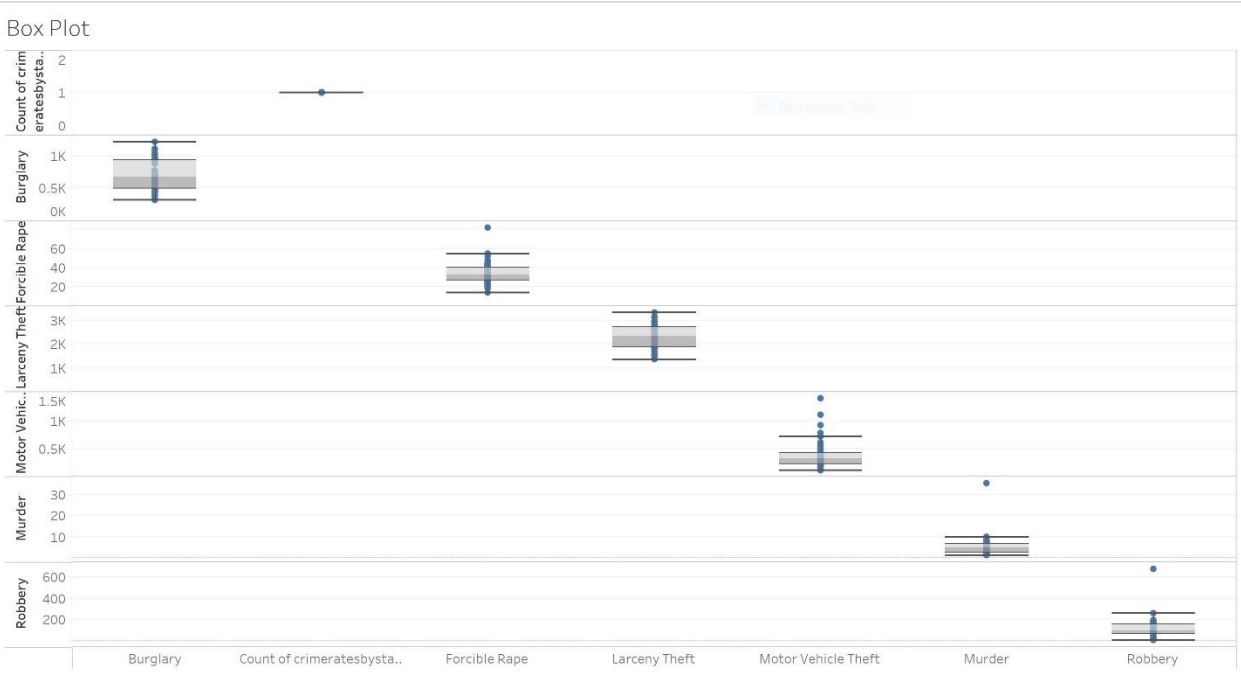
```
fig = plt.figure()
bp = states_education.boxplot(grid = False)
plt.title("Box Plot - Python\nUnited States Education")
plt.xticks([1, 2, 3, 4, 5, 6], ['Reading', 'Math', 'Writing', 'Percent Graduates SAT', 'Pupil to Staff Ratio', 'Dropout Rate'
plt.xticks(rotation = 45, ha = 'right', rotation_mode = 'anchor')
plt.tight_layout()
plt.show()
```

R

```
#| label: boxplot
fig <- ggplot(gather(just_scores_df), aes(value)) +
  geom_boxplot() +
  facet_wrap(~key, scales = 'free_x') +
  ggtitle("Box Plots - R\nUnited States Education")
fig
```



Tableau



Bullet Chart

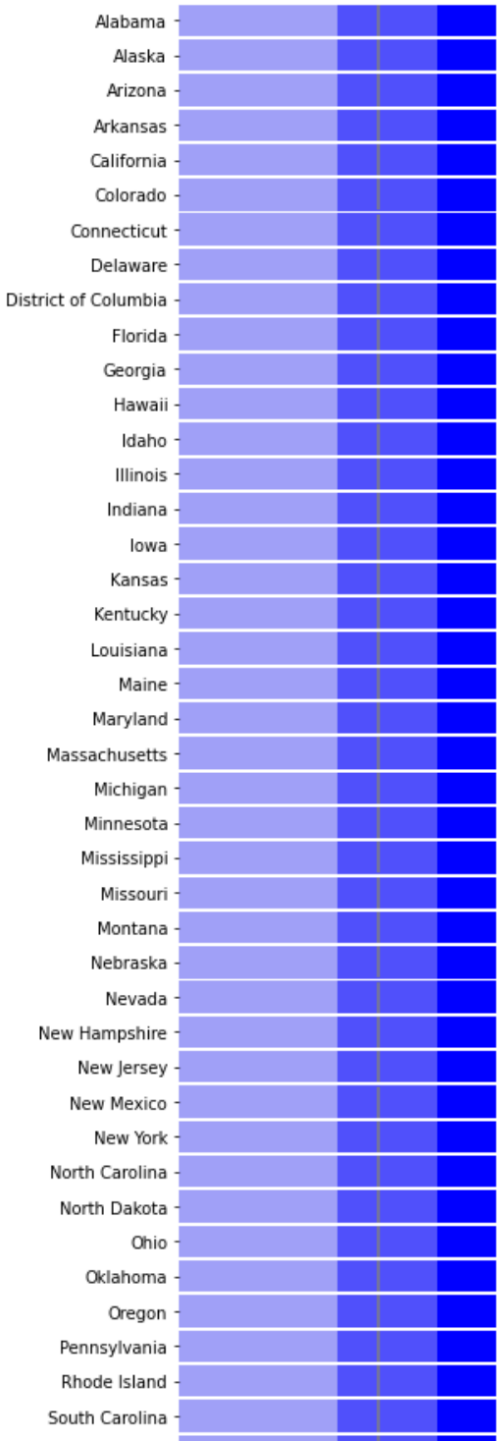
Python

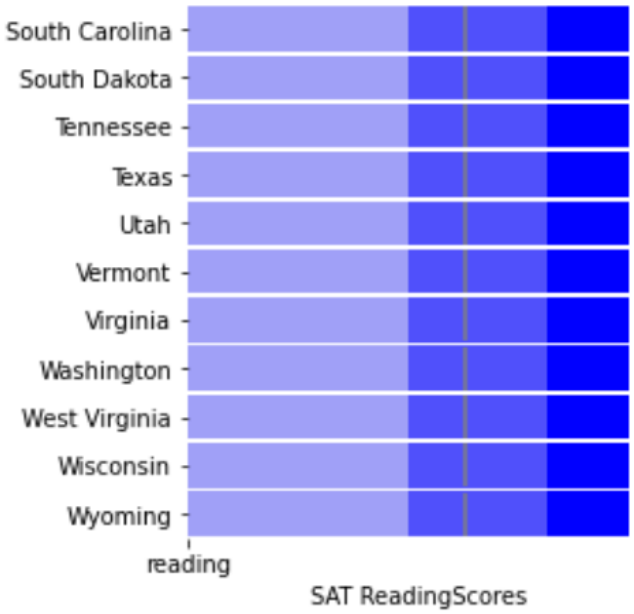
```python
def bulletgraph(data=None, limits=None, labels=None, axis_label=None, title=None,
                size=(15, 20), palette=None, formatter=None, target_color="gray",
                bar_color="black", label_color="gray"):
    """ Build out a bullet graph image
        Args:
            data = List of labels, measures and targets
            limits = list of range values
            labels = list of descriptions of the limit ranges
            axis_label = string describing x axis
            title = string title of plot
            size = tuple for plot size
            palette = a seaborn palette
            formatter = matplotlib formatter object for x axis
            target_color = color string for the target line
            bar_color = color string for the small bar
            label_color = color string for the limit label text
        Returns:
            a matplotlib figure
    """
    # Determine the max value for adjusting the bar height
    # Dividing by 10 seems to work pretty well
    h = limits[-1] / 10

    # Use the green palette as a sensible default
    if palette is None:
        palette = sns.light_palette("blue", len(limits), reverse=False)

    # Must be able to handle one or many data sets via multiple subplots
    if len(data) == 1:
        fig, ax = plt.subplots(figsize=size, sharex=True)
    else:
        fig, axarr = plt.subplots(len(data), figsize=size, sharex=True)

    # Add each bullet graph bar to a subplot
    for idx, item in enumerate(data):

        # Get the axis from the array of axes returned when the plot is created
        if len(data) > 1:
            ax = axarr[idx]
```

```
        # Get the axis from the array of axes returned when the plot is created
        if len(data) > 1:
            ax = axarr[idx]

        # Formatting to get rid of extra marking clutter
        ax.set_aspect('equal')
        ax.set_yticklabels([item[0]])
        ax.set_yticks([1])
        ax.spines['bottom'].set_visible(False)
        ax.spines['top'].set_visible(False)
        ax.spines['right'].set_visible(False)
        ax.spines['left'].set_visible(False)

        prev_limit = 0
        for idx2, lim in enumerate(limits):
            # Draw the bar
            ax.barh([1], lim - prev_limit, left=prev_limit, height=h,
                    color=palette[idx2])
            prev_limit = lim
        rects = ax.patches
        # The last item in the list is the value we're measuring
        # Draw the value we're measuring
        ax.barh([1], item[1], height=(h / 3), color=bar_color)

        # Need the ymin and max in order to make sure the target marker
        # fits
        ymin, ymax = ax.get_ylim()
        ax.vlines(
            item[2], ymin * .9, ymax * .9, linewidth=1.5, color=target_color)

    # Now make some labels
    if labels is not None:
        for rect, label in zip(rects, labels):
            height = rect.get_height()
            ax.text(
                rect.get_x() + rect.get_width() / 2,
                -height * .4,
                label,
                ha='center',
                va='bottom',
                color=label_color)
    if formatter:
        ax.xaxis.set_major_formatter(formatter)
    if axis_label:
        ax.set_xlabel(axis_label)
    if title:
        fig.suptitle(title, fontsize=14)
    fig.subplots_adjust(hspace=0)
```

```
data=[]
for index, row in education_df.iterrows():
    if index == 0:
        us_score = row['reading']
    else:
        state_score = ['reading']
        data.append((row["state"],state_score, us_score))
```

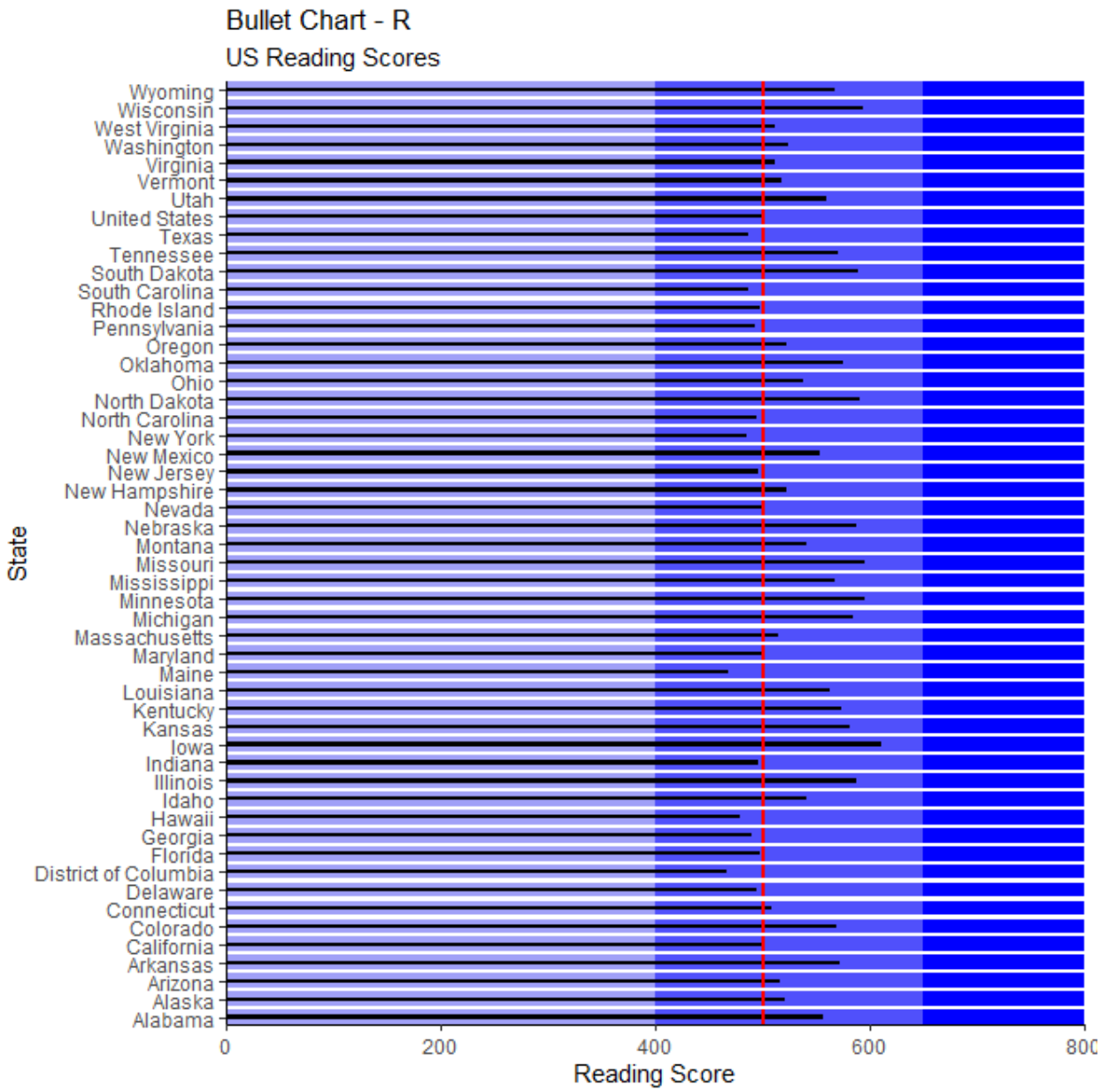
```
bulletgraph(data,limits = [0, 400, 650, 800],
            axis_label ='SAT ReadingScores', title = "Bullet Chart - Python\nU.S. SAT READING Score with US Avg")
#disable warnings
import warnings
warnings.filterwarnings("ignore")
```


Alabama
Alaska
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
District of Columbia
Florida
Georgia
Hawaii
Idaho
Illinois
Indiana
Iowa
Kansas
Kentucky
Louisiana
Maine
Maryland
Massachusetts
Michigan
Minnesota
Mississippi
Missouri
Montana
Nebraska
Nevada
New Hampshire
New Jersey
New Mexico
New York
North Carolina
North Dakota
Ohio
Oklahoma
Oregon
Pennsylvania
Rhode Island
South Carolina

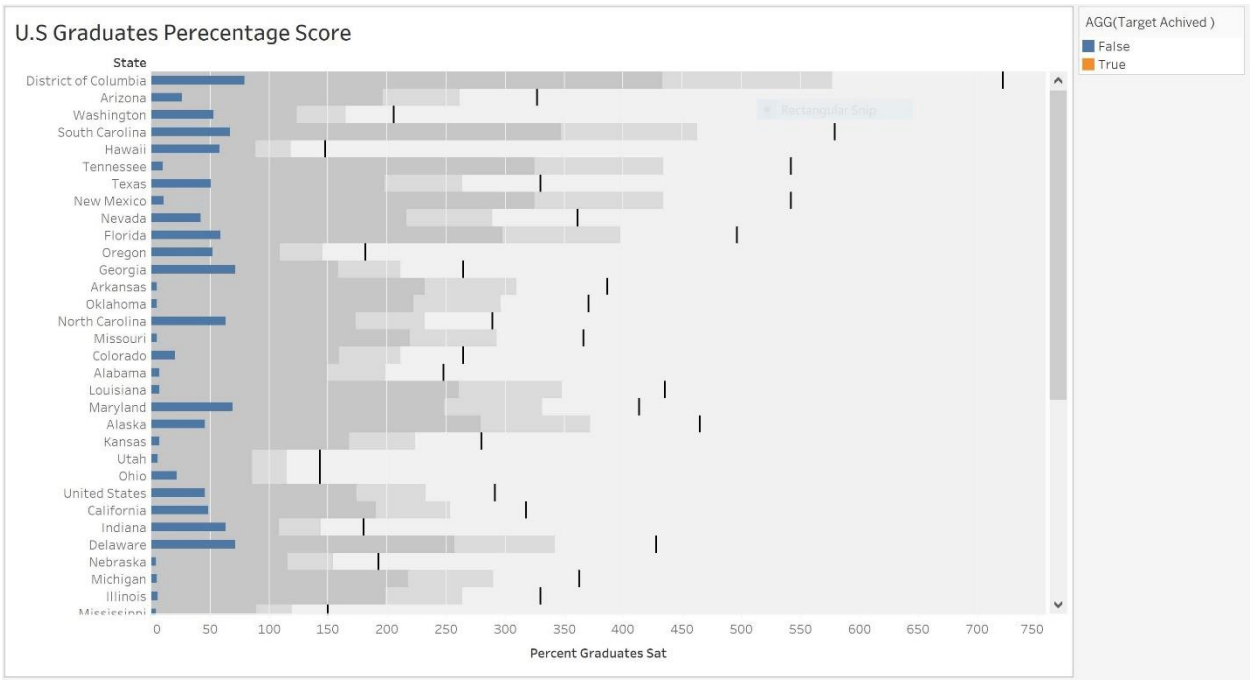


R

```r
# Bullet Chart
```{r}
#| label: formatdata
us_states <- education_df$state
reading_scores <- education_df$reading
data = data.frame(us_states, reading_scores, stringsAsFactors = TRUE)
data <- data[-1,]
data
```
```{r}
#| label: usreading
us_average <- education_df[education_df$state == 'United States', 'reading']
us_average
```
```{r}
#| label: bulletchart
fig <- ggplot() +
  geom_bar(data = education_df,
           aes(x = state, y = 800), stat = "identity", width = .8, fill = '#0000ff',
           position = "stack") +
  geom_bar(data = education_df,
           aes(x = state, y = 650), stat = "identity", width = .8, fill = '#5050fb',
           position = "stack") +
  geom_bar(data = education_df,
           aes(x = state, y = 400), stat = "identity", width = .8, fill = '#a0a0f7',
           position = "stack") +
  geom_bar(data = education_df,
           aes(x = state, y = reading), fill = "black", width = 0.2,
           stat = "identity") +
  coord_flip(expand = FALSE) +
  labs(title='Bullet Chart - R', subtitle = 'US Reading Scores',
       x = 'State', y = 'Reading Score') +
  geom_errorbar(data = education_df,
                aes(x = state, ymin = us_average, ymax = us_average),
                color = 'red', width = 0.8, size = 1)
fig
```
```


Bullet Chart - R
US Reading Scores
Wyoming
Wisconsin
West Virginia
Washington
Virginia
Vermont
Utah
United States
Texas
Tennessee
South Dakota
South Carolina
Rhode Island
Pennsylvania
Oregon
Oklahoma
Ohio
North Dakota
North Carolina
New York
New Mexico
New Jersey
New Hampshire
Nevada
Nebraska
Montana
Missouri
Mississippi
Minnesota
Michigan
Massachusetts
Maryland
Maine
Louisiana
Kentucky
Kansas
Iowa
Indiana
Illinois
Idaho
Hawaii
Georgia
Florida
District of Columbia
Delaware
Connecticut
Colorado
California
Arkansas
Arizona
Alaska
Alabama
State
0
200
400
600
800
Reading Score

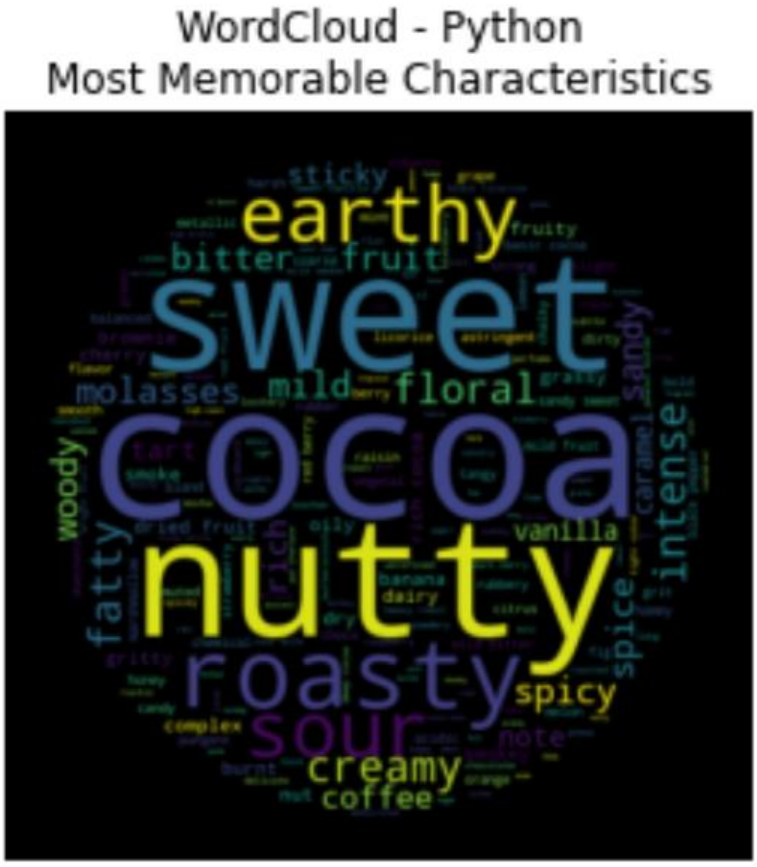
Tableau


U.S Graduates Perecentage Score
State
District of Columbia
Arizona
Washington
South Carolina
Hawaii
Tennessee
Texas
New Mexico
Nevada
Florida
Oregon
Georgia
Arkansas
Oklahoma
North Carolina
Missouri
Colorado
Alabama
Louisiana
Maryland
Alaska
Kansas
Utah
Ohio
United States
California
Indiana
Delaware
Nebraska
Michigan
Illinois
0
50
100
150
200
250
300
350
400
450
500
550
600
650
700
750
Percent Graduates Sat
AGG(Target Achived)
False
True

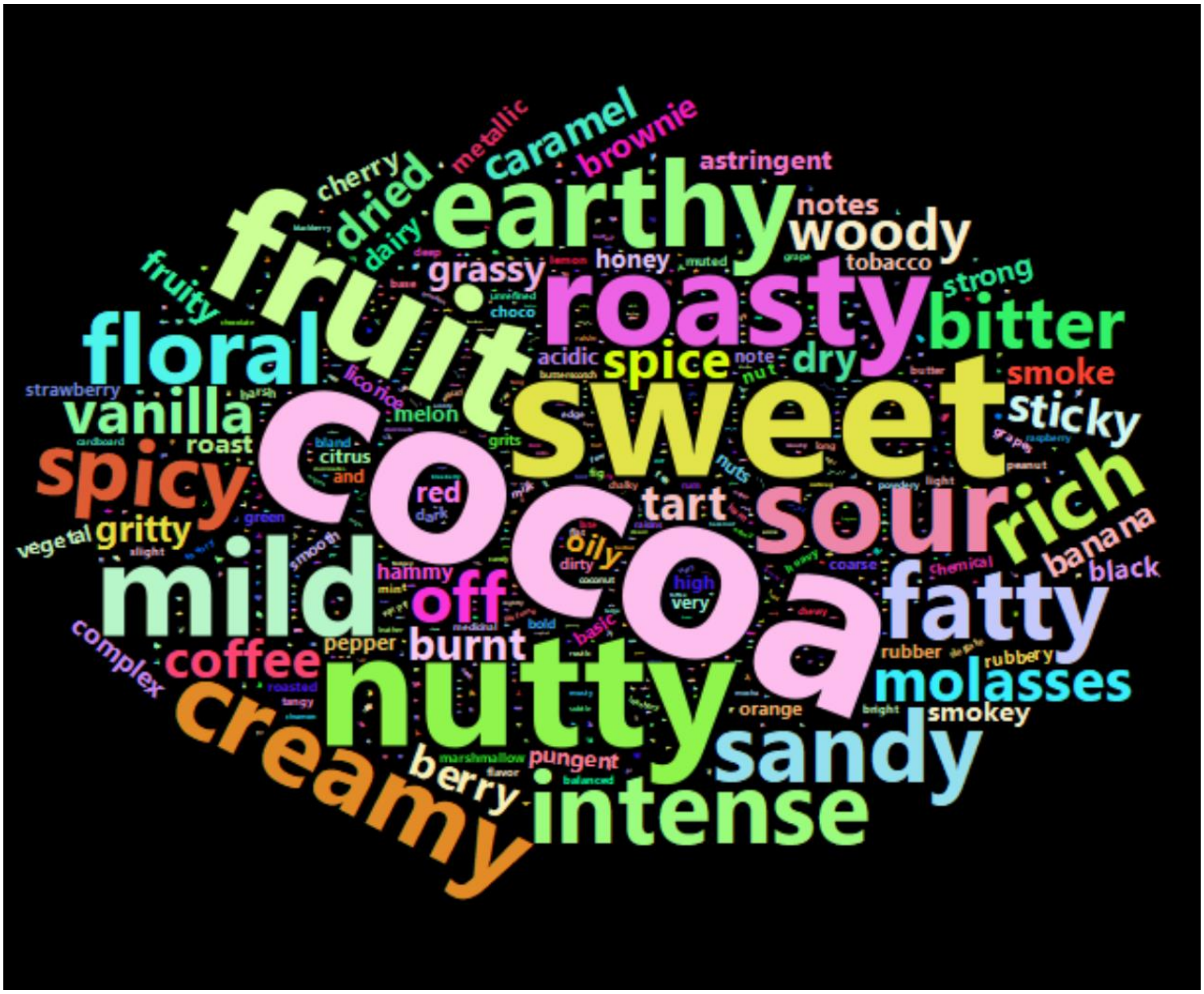
My Choice – Word Cloud

Python

```
# Most frequent words in the Characteristics
word_string = " ".join(Chocolate_df['Most Memorable Characteristics'].str.lower())
# Create a circle mask
x, y = np.ogrid[:400, :400]
mask = (x - 200) ** 2 + (y - 200) ** 2 > 180 ** 2
mask = 255 * mask.astype(int)
wc = WordCloud(background_color = "black", repeat = True, mask = mask)
wc.generate(word_string)
plt.axis("off")
plt.imshow(wc, interpolation = "bilinear")
plt.title('WordCloud - Python\nMost Memorable Characteristics')
```



R


cocoa
sweet
fruit
roasty
earthy
nutty
sour
mild
fatty
sandy
creamy
intense
floral
spicy
rich
bitter
vanilla
woody
molasses
coffee
spice
sticky
burnt
off
tart
dried
caramel
brownie
grassy
gritty
banana
smoke
dry
strong
fruity
cherry
astringent
metallic
notes
honey
tobacco
dairy
berry
smokey
roast
red
oily
melon
acidic
citrus
hammy
pepper
rubber
rubbery
orange
pungent
complex
vegetal
strawberry
licorice
black
high
very
note
nut
nuts
choco
marshmallow
coarse
chemical
grits
dirty
basic
bold
green
roasted
smooth
and
mint
bland
harsh
dark
balanced
bright

Tableau


cocoa and coconut
creamy, rounded, orange
creamy, sticky, fruit
creamy, acidic, balanced
creamy, sticky, dried fruit
caramel, nuts, dried fruit
creamy, nutty, earthy
creamy, nutty
balanced, nuts, strawberry
balanced, cherry, choco
distinct lemon
complex,spice,caramel,cocoa
creamy, nutty, banana, rich
dark berry, tropical, nutty
chocolate covered banana
complex, hazelnut, dairy, fruit
creamy,sticky, peanut butter
creamy, nuts, woody, cocoa
dried fruit, fig, astringent, sticky
creamy, blueberry, raspberry
creamy, nutty, delicate fruit
cardamon
cocoa, mild fruit
dry, red berry, off note
creamy, nutty, cocoa
dried fruit, intense
creamy, nutty, fruity
creamy, honey,peanut butter
creamy,choco strawberry,vanilla
delicate, nutty, cocoa, dairy
creamy, woody, cocoa
creamy, raisin, lemon
creamy, honey, marshmallow
creamy, mint, tobacco, olive
creamy, homey, nutty
cinamon, nutmeg, hot cocoa
creamy, honey, blackberry
creamy, sweet,cocoa,banana
complex, strawberry, floral
creamy, strawberry, nutty
dark berry, honey, cream
creamy, grassy, smoke, nut
delicate, hazelnut, brownie
cocoa with hint of melon
distinct choco and graham
complex, raspberry, cocoa
creamy, pistachio, floral
creamy, woody, nutty
creamy, rich, complex
creamy, mild spice, cocoa
chocolate and grapes
creamy, spicy, cocoa
cinamon and nutmeg
creamy, bright fruit