Ryan Long
DSC640
3.2 Exercises: Charts

Datasets used:
expenditures.txt
unemployement-rate-1948-2010.csv

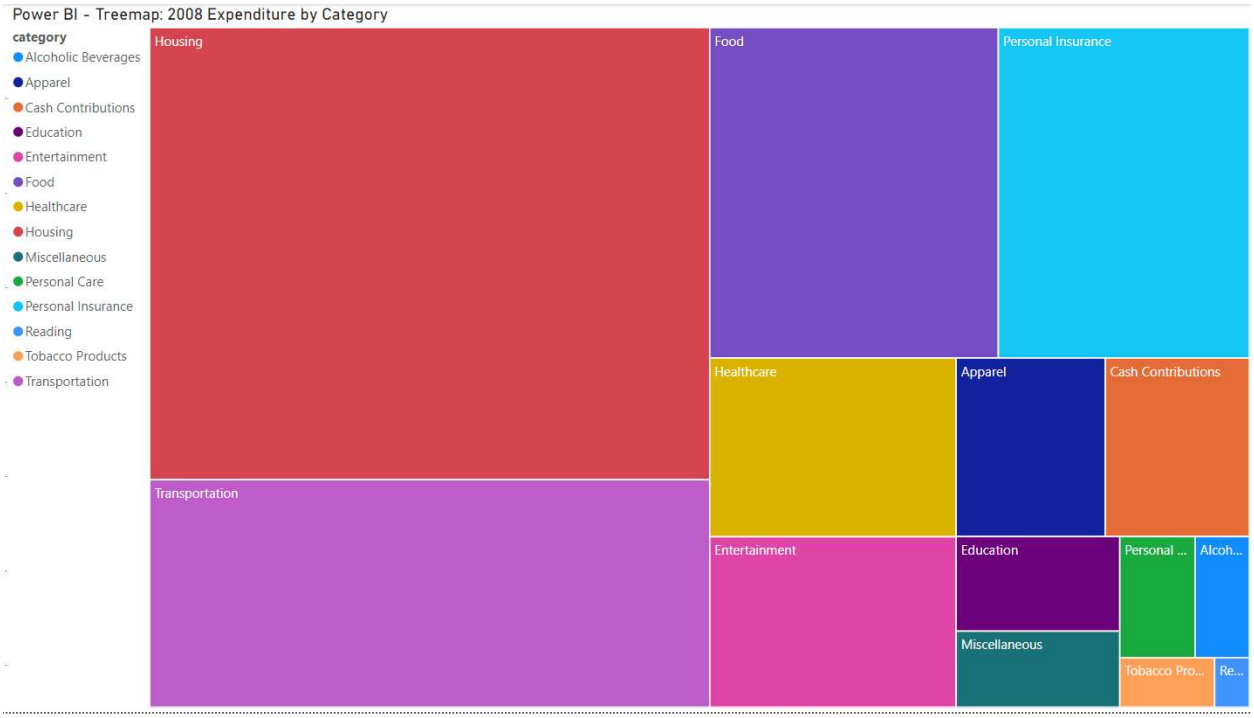
Summary
I began this exercise as I have the prior, working first from Power BI, Python, then R. My two main challenges were managing the data so that it could be graphed and understanding how to apply the required graphs to the data. It was very circular in nature and even as I progressed through the application/languages I revisited my previously created graphs and adjusted what I was attempting to depict.

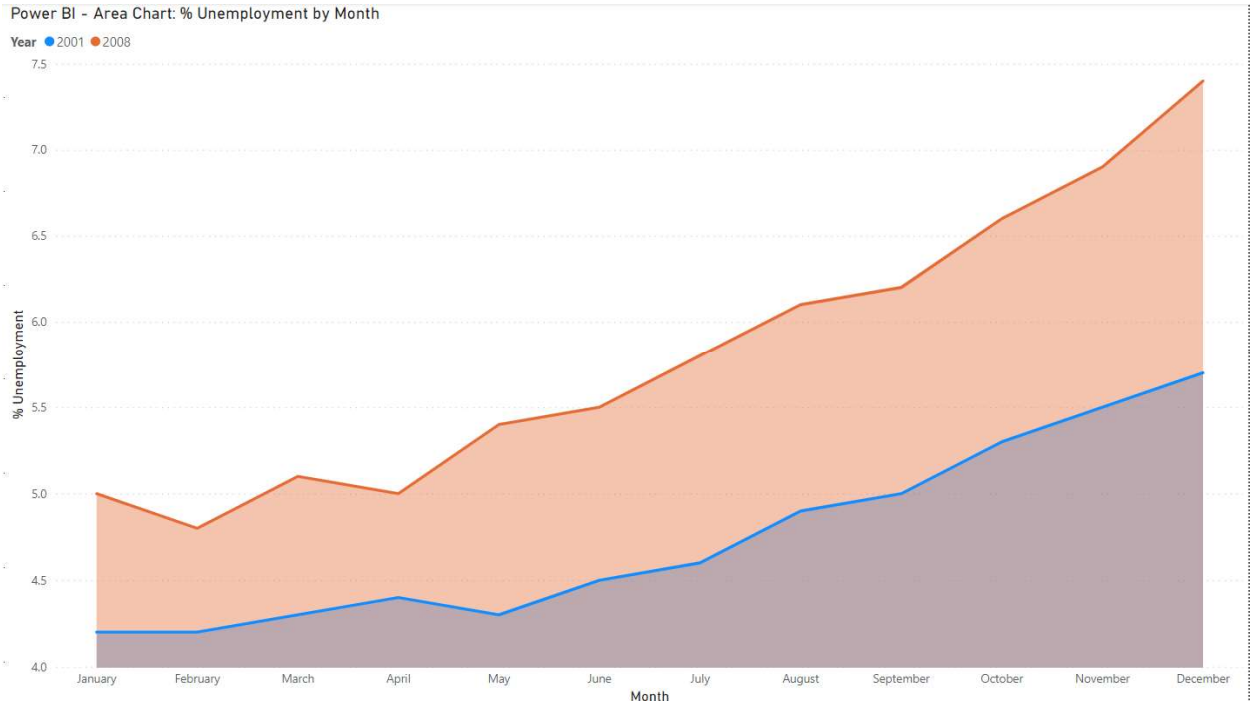
The greatest frustration came from the area charts. I was not pleased with the output from either of the application/languages. The Power BI was the best, as the colored line depicting 2001 helps the reader associate the information from the legend within the chart. For the Python chart, 2001 is defined as pink, however it shows up as a deeper blue than 2008. This occurred regardless of the colors I chose. Finally, R turned out a little better than Python in that the colors were more distinct for both years used. As an afterthought, this data may not have been the best use for the area charts as it is meant to depict a volume/quantity and unemployment is simply a rate. A simple line chart would be the best representation of this data, but I wanted to experiment with both datasets.

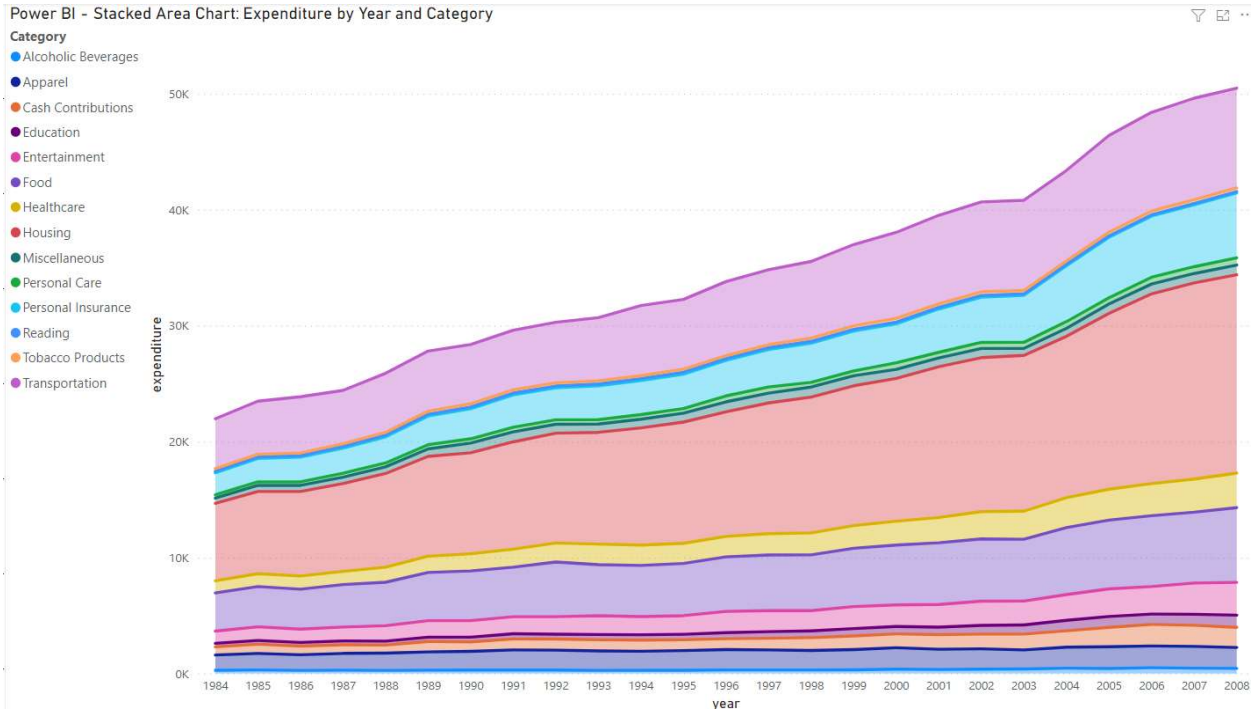
The stacked area charts turned out much better than the area charts and I enjoyed working the data into an easy-to-use format for all application/languages.

The code support contains a few more iterations and attempts of the graphs. Please consider the snippets prior to the 'Appendix' my submission.

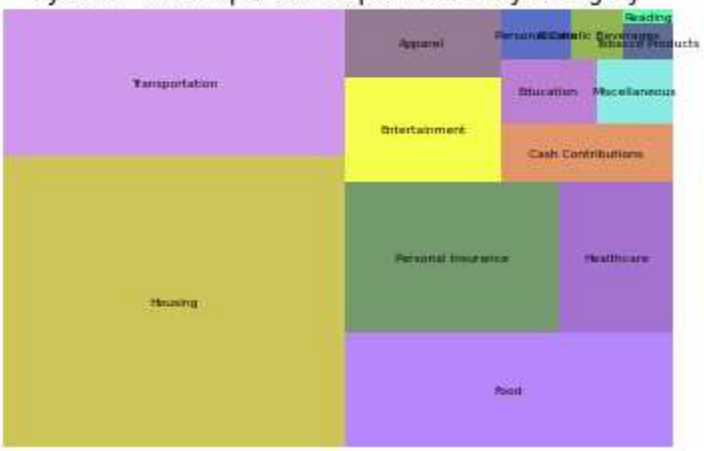
The following pages contain:


Power BI - Treemap: 2008 Expenditure by Category
category
Alcoholic Beverages
Apparel
Cash Contributions
Education
Entertainment
Food
Healthcare
Housing
Miscellaneous
Personal Care
Personal Insurance
Reading
Tobacco Products
Transportation
Housing
Food
Personal Insurance
Healthcare
Apparel
Cash Contributions
Transportation
Entertainment
Education
Personal ...
Alcoh...
Miscellaneous
Tobacco Pro...
Re...

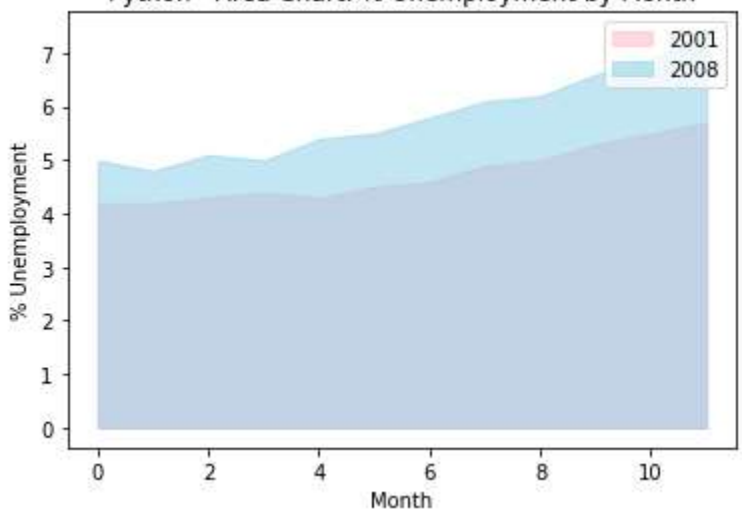

Power BI - Area Chart: % Unemployment by Month
Year 2001 2008
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
% Unemployment
January
February
March
April
May
June
July
August
September
October
November
December
Month


Power BI - Stacked Area Chart: Expenditure by Year and Category
Category
Alcoholic Beverages
Apparel
Cash Contributions
Education
Entertainment
Food
Healthcare
Housing
Miscellaneous
Personal Care
Personal Insurance
Reading
Tobacco Products
Transportation
expenditure
50K
40K
30K
20K
10K
0K
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
year

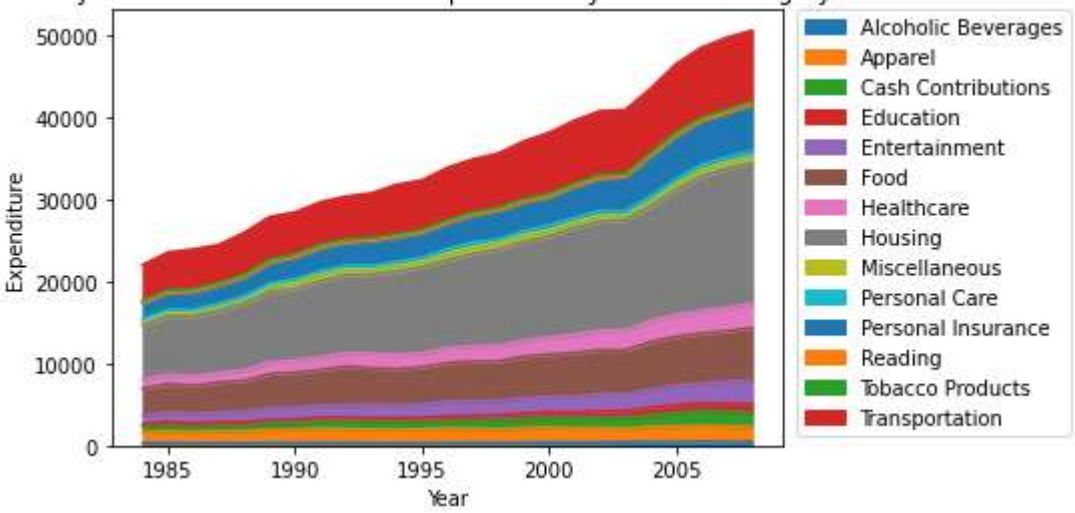
Python - Treemap: 2008 Expenditures by Category



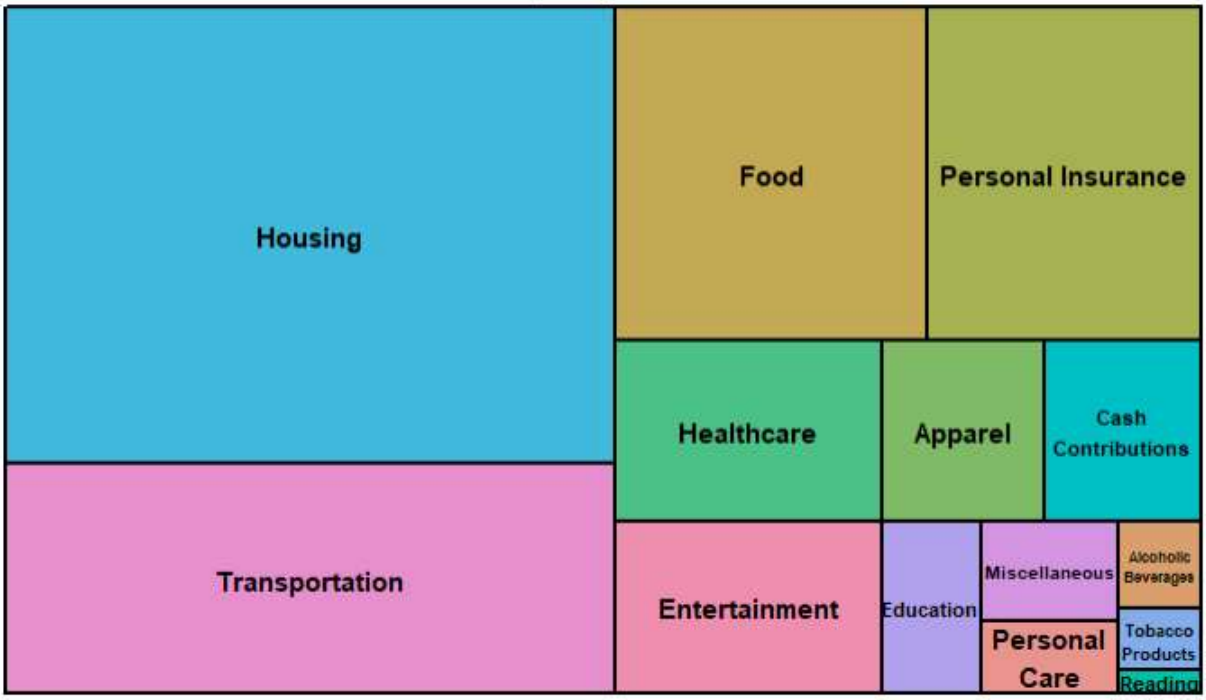
Python - Area Chart: % Unemployment by Month



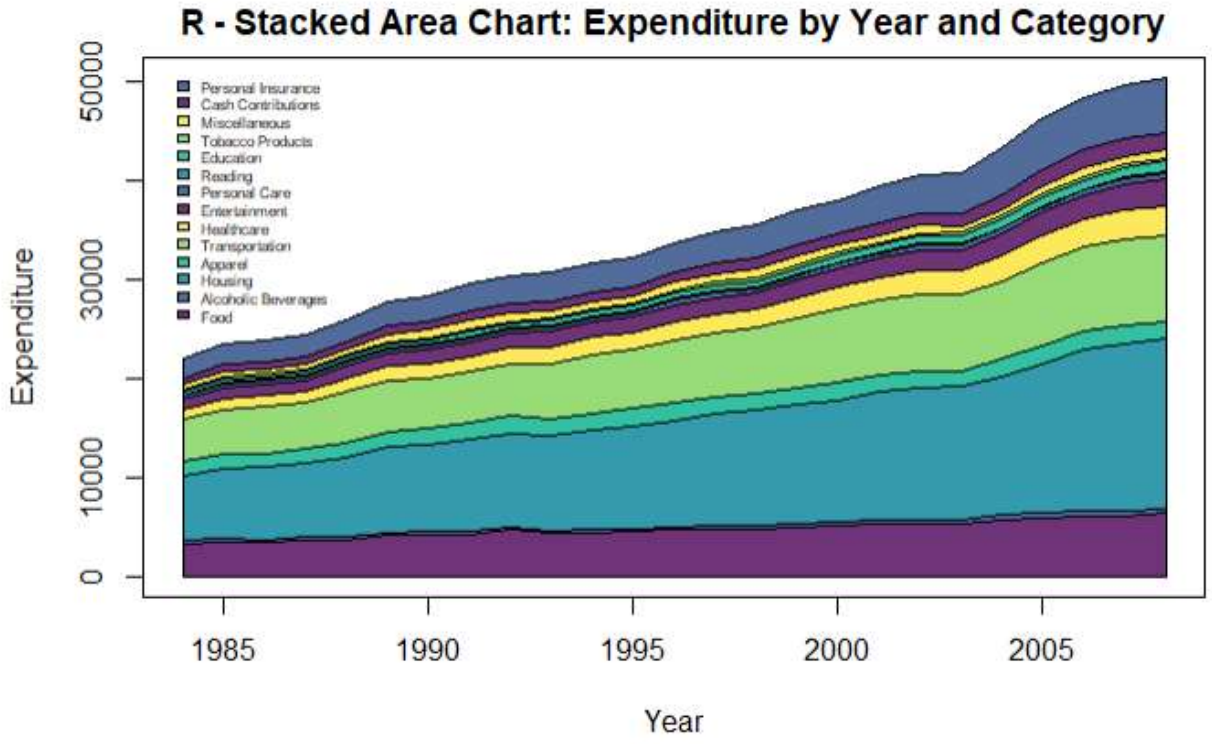
Python - Stacked Area Chart: Expenditure by Year and Category



R - Treemap: Expenditures 2008




R - Area Chart: Monthly % Unemployment
% Unemployment
0
2
4
6
2.5
5.0
7.5
10.0
12.5
Month
Year
2001
2008


R - Stacked Area Chart: Expenditure by Year and Category
Expenditure
0
10000
30000
50000
1985
1990
1995
2000
2005
Year
Personal Insurance
Cash Contributions
Miscellaneous
Tobacco Products
Education
Reading
Personal Care
Entertainment
Healthcare
Transportation
Apparel
Housing
Alcoholic Beverages
Food

APPENDIX

In [1]:
```
#load libraries
import pandas as pd
import matplotlib.pyplot as plt
import squarify
import numpy as np
import matplotlib.ticker as plticker # for plot ticks
```

In [2]:
```
#import data as dataframe
data = pd.read_csv('expenditures.txt',names=['Year','Category','Expenditure','Sex
data2 = pd.read_csv('unemployement-rate-1948-2010.csv')
```

In [3]:
```
#data.info()
```

In [4]:
```
#data.head()
```

In [5]:
```
#data2.head()
```

In [6]:
```
#create date column for unemployment df
data2['Month'] = pd.to_numeric(data2['Period'].str[-2:])
data2['Day'] = 1
data2['Date'] = pd.to_datetime(data2[['Year','Month','Day']])
```

In [7]:
```
data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 746 entries, 0 to 745
Data columns (total 7 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Series id  746 non-null    object
 1   Year       746 non-null    int64
 2   Period     746 non-null    object
 3   Value      746 non-null    float64
 4   Month      746 non-null    int64
 5   Day        746 non-null    int64
 6   Date       746 non-null    datetime64[ns]
dtypes: datetime64[ns](1), float64(1), int64(3), object(2)
memory usage: 40.9+ KB
```

Treemap

Resources: https://www.analyticsvidhya.com/blog/2021/06/build-treemaps-in-python-using-squarify/ (https://www.analyticsvidhya.com/blog/2021/06/build-treemaps-in-python-using-squarify/)

https://jingwen-z.github.io/data-viz-with-matplotlib-series5-treemap/ (https://jingwen-z.github.io/data-viz-with-matplotlib-series5-treemap/)

In [8]:
```
# get filtered data
#selecting rows based year column
treemapdata = data[((data['Year'] >= 1999) & (data['Category'] == "Alcoholic Beve
```

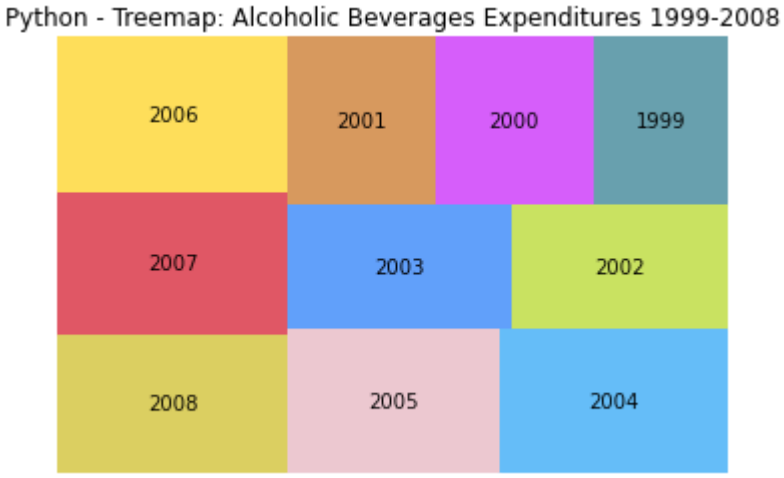
In [9]:
```
treemapdata.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10 entries, 1 to 127
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Year         10 non-null     int64
 1   Category     10 non-null     object
 2   Expenditure  10 non-null     int64
 3   Sex          10 non-null     int64
dtypes: int64(3), object(1)
memory usage: 400.0+ bytes
```

In [10]:
```
def get_cmap(n, name='hsv'):
    '''Returns a function that maps each index in 0, 1, ..., n-1 to a distinct
    RGB color; the keyword argument name must be a standard mpl colormap name.'''
    return plt.cm.get_cmap(name, n)
```

In [11]:
```
squarify.plot(sizes=treemapdata['Expenditure'],
              label=treemapdata['Year'],
              color=np.random.rand(len(treemapdata['Year']),3), #creates a random
              alpha=0.7 )
#plt.xlabel("not used")  # X-axis label
#plt.ylabel("not used")  # Y-axis label
plt.axis('off') # turns axis off
plt.title("Python - Treemap: Alcoholic Beverages Expenditures 1999-2008")  # titl
```

Out[11]:
```
Text(0.5, 1.0, 'Python - Treemap: Alcoholic Beverages Expenditures 1999-2008')
```



In [42]:
```
tmdata2 = data.loc[data["Year"] == 2008].sort_values(by=['Expenditure'], ascendir
```

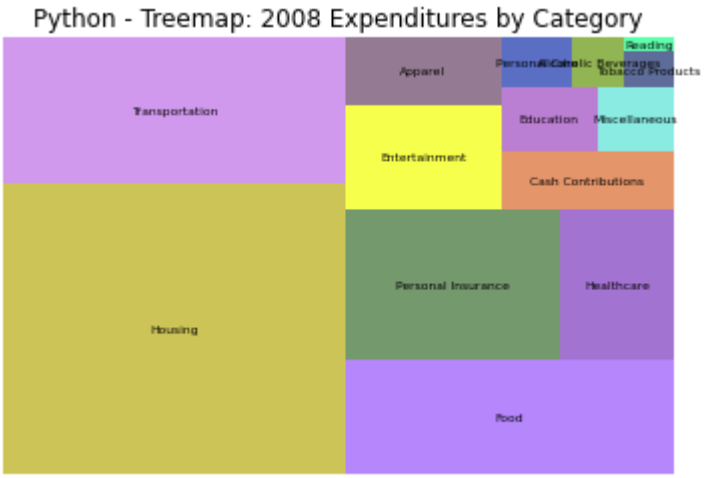
In [43]:
```
tmdata2
```

Out[43]:

	Year	Category	Expenditure	Sex
2	2008	Housing	17109	1
4	2008	Transportation	8604	1
0	2008	Food	6443	1
13	2008	Personal Insurance	5605	1
5	2008	Healthcare	2976	1
6	2008	Entertainment	2835	1
3	2008	Apparel	1801	1
12	2008	Cash Contributions	1737	1
9	2008	Education	1046	1
11	2008	Miscellaneous	840	1
7	2008	Personal Care	616	1
1	2008	Alcoholic Beverages	444	1
10	2008	Tobacco Products	317	1
8	2008	Reading	116	1

In [54]:
```
squarify.plot(sizes=tmdata2['Expenditure'],
              label=tmdata2['Category'],
              color=np.random.rand(len(tmdata2['Category']),3), #creates a random
              alpha=0.7,
              text_kwargs={'fontsize':6})
#plt.xlabel("not used")  # X-axis label
#plt.ylabel("not used")  # Y-axis label
plt.axis('off') # turns axis off
plt.title("Python - Treemap: 2008 Expenditures by Category")  # title
```

Out[54]:
```
Text(0.5, 1.0, 'Python - Treemap: 2008 Expenditures by Category')
```



Area Chart

References

https://jingwen-z.github.io/data-viz-with-matplotlib-series7-area-chart/ (https://jingwen-z.github.io/data-viz-with-matplotlib-series7-area-chart/)

In [12]:
```
# area chart data
areadata = data2[((data2['Year'] == 2001) | (data2['Year'] == 2008))]

#areadata=areadata.set_index('Date')
#Year2001 = data2(data2['Year'] == 2001)
#Year2001 = data2[(data2['Year'] == 2001)]
#Year2008 = data2[(data2['Year'] == 2008)]
```

In [13]:
```
pivareadata=areadata.pivot(index='Month',columns='Year',values='Value')
pivareadata.columns = ['2001','2008']
```

In [14]:
```
pivareadata.info()
```

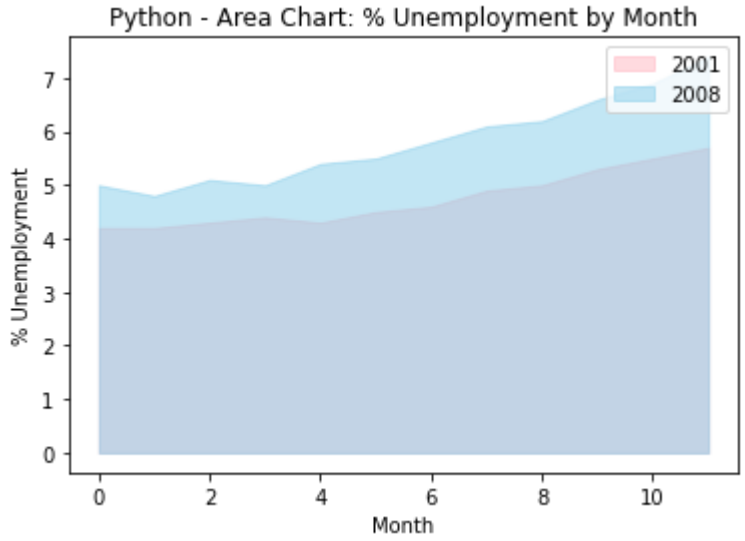
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 12 entries, 1 to 12
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   2001    12 non-null     float64
 1   2008    12 non-null     float64
dtypes: float64(2)
memory usage: 288.0 bytes
```

In [15]:
```
pivareadata
```

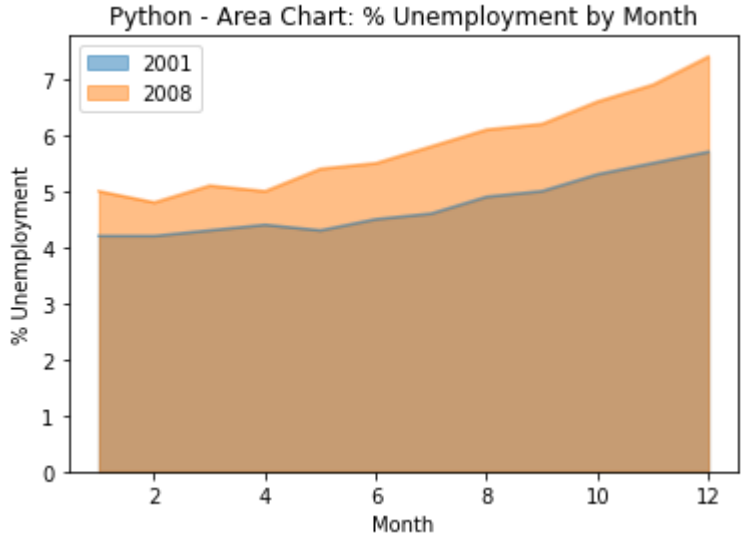
Out[15]:

Month	2001	2008
1	4.2	5.0
2	4.2	4.8
3	4.3	5.1
4	4.4	5.0
5	4.3	5.4
6	4.5	5.5
7	4.6	5.8
8	4.9	6.1
9	5.0	6.2
10	5.3	6.6
11	5.5	6.9
12	5.7	7.4

In [16]:
```
plt.fill_between(np.arange(12), pivareadata['2001'], color="lightpink", alpha=0.5
plt.fill_between(np.arange(12), pivareadata['2008'], color="skyblue", alpha=0.5,

plt.xlabel("Month")  # X-axis label
plt.ylabel("% Unemployment")  # Y-axis label
plt.title("Python - Area Chart: % Unemployment by Month")  # title

plt.legend()
plt.show()
```



In [17]:
```
pivareadata.plot(kind='area',stacked=False)

plt.xlabel("Month")  # X-axis label
plt.ylabel("% Unemployment")  # Y-axis label
plt.title("Python - Area Chart: % Unemployment by Month")  # title


plt.show(block=True)
```



Stacked Area Chart

plt.stackplot(pivareadata.index, [pivareadata['2001'], pivareadata['2008']], labels=['2001', '2008'], alpha=0.8)

plt.xlabel("Month") # X-axis label plt.ylabel("% Unemployment") # Y-axis label plt.title("Python - Stacked Area Chart: % Unemployment by Month") # title

plt.legend(loc=2, fontsize='large') plt.show()

In [18]:
```
data
```

Out[18]:

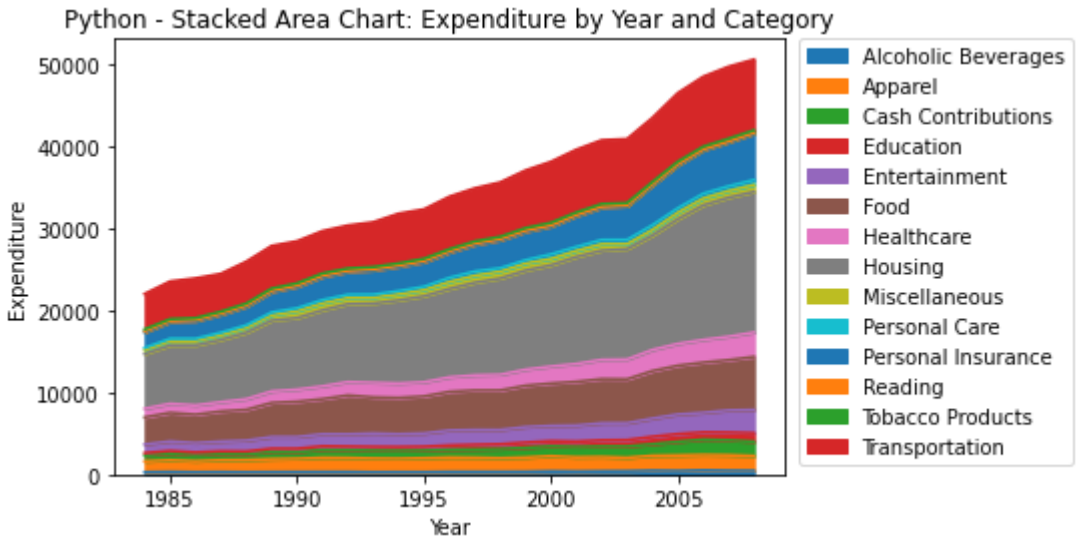
	Year	Category	Expenditure	Sex
0	2008	Food	6443	1
1	2008	Alcoholic Beverages	444	1
2	2008	Housing	17109	1
3	2008	Apparel	1801	1
4	2008	Transportation	8604	1
...	...	...	...	...
345	1984	Education	303	1
346	1984	Tobacco Products	228	1
347	1984	Miscellaneous	451	1
348	1984	Cash Contributions	706	1
349	1984	Personal Insurance	1897	1

350 rows × 4 columns

In [19]:
```
newdf=data.pivot(index='Year',columns='Category',values='Expenditure')
```

In [20]:
```
#newdf
```

In [21]:
```
newdf.plot.area()

plt.legend(bbox_to_anchor=(1.02, 1), loc='upper left', borderaxespad=0)
plt.xlabel("Year")  # X-axis label
plt.ylabel("Expenditure")  # Y-axis label
plt.title("Python - Stacked Area Chart: Expenditure by Year and Category")  # tit

plt.show()
```



In [ ]:

Week 5 & 6

Code

Hide

```
#load libraries
library(ggplot2)
library(dplyr)
```

```
Attaching package: ‘dplyr’

The following objects are masked from ‘package:stats’:

    filter, lag

The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union
```

Hide

```
library(tidyr)
library(treemap) #for treemap
```

```
Registered S3 method overwritten by 'data.table':
  method           from
  print.data.table
Registered S3 methods overwritten by 'htmltools':
  method               from
  print.html           tools:rstudio
  print.shiny.tag      tools:rstudio
  print.shiny.tag.list tools:rstudio
```

Hide

```
library(hrbrthemes)
```

```
Registering Windows fonts with R
NOTE: Either Arial Narrow or Roboto Condensed fonts are required to use these themes.
      Please use hrbrthemes::import_roboto_condensed() to install Roboto Condensed and
      if Arial Narrow is not on your system, please see https://bit.ly/arialnarrow
```

Hide

```
library(pivottabler)
```

```
Registered S3 method overwritten by 'htmlwidgets':
  method           from
  print.htmlwidget tools:rstudio
```

Hide

```
library(areaplot)
```

Hide

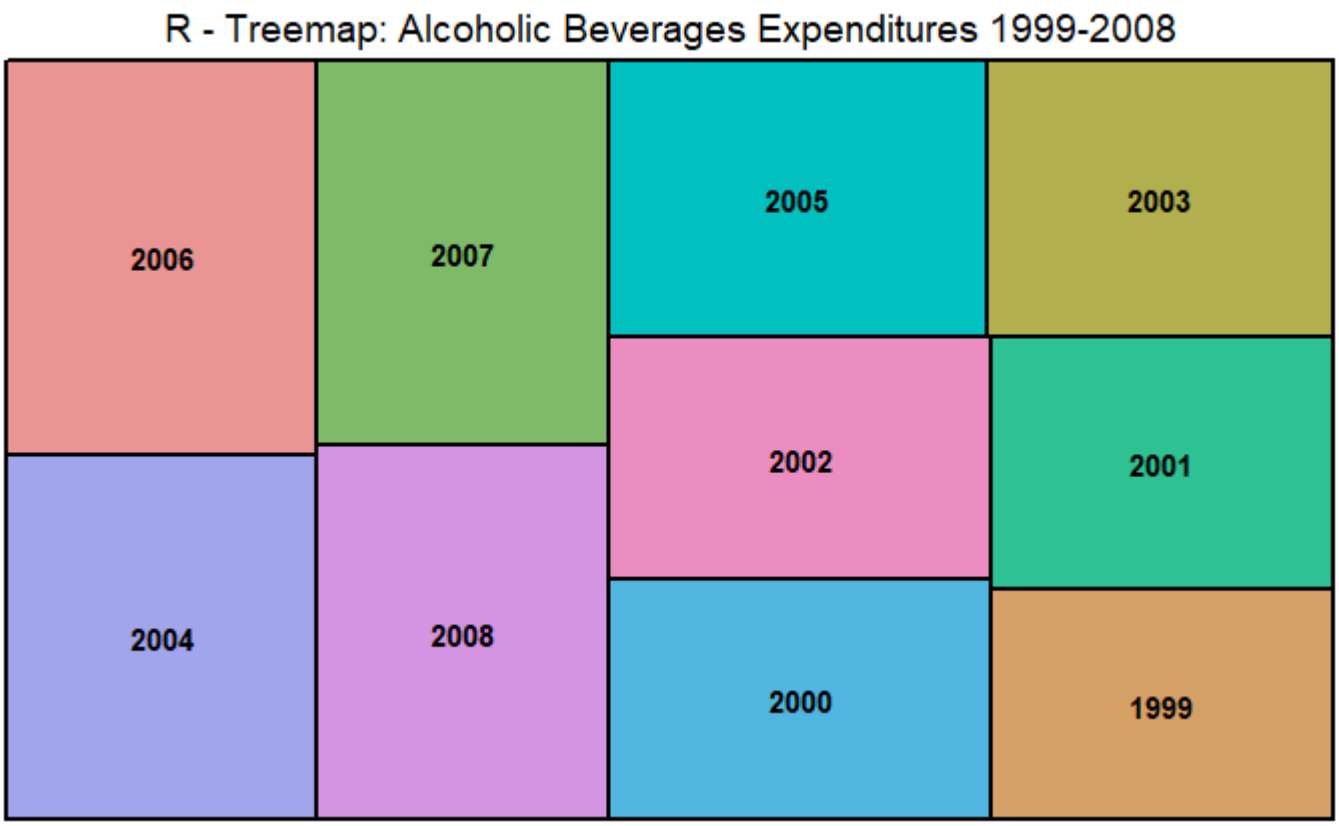
```
#import data
data1 = read.delim("C:\\Users\\longr\\Documents\\DSC 640\\Week 5 & 6\\3.2 Exercises\\expenditure
s.txt", sep = '\t')
data2 = read.csv("C:\\Users\\longr\\Documents\\DSC 640\\Week 5 & 6\\3.2 Exercises\\unemployement
-rate-1948-2010.csv")
```

Hide

```
#filter data for treemap
tmd <- filter(data1, year >= 1999 & category == "Alcoholic Beverages")
```

Hide

```
#create treemap
treemap(tmd,
        index="year",
        vSize="expenditure",
        type="index",
        title ='R - Treemap: Alcoholic Beverages Expenditures 1999-2008'
            )
```

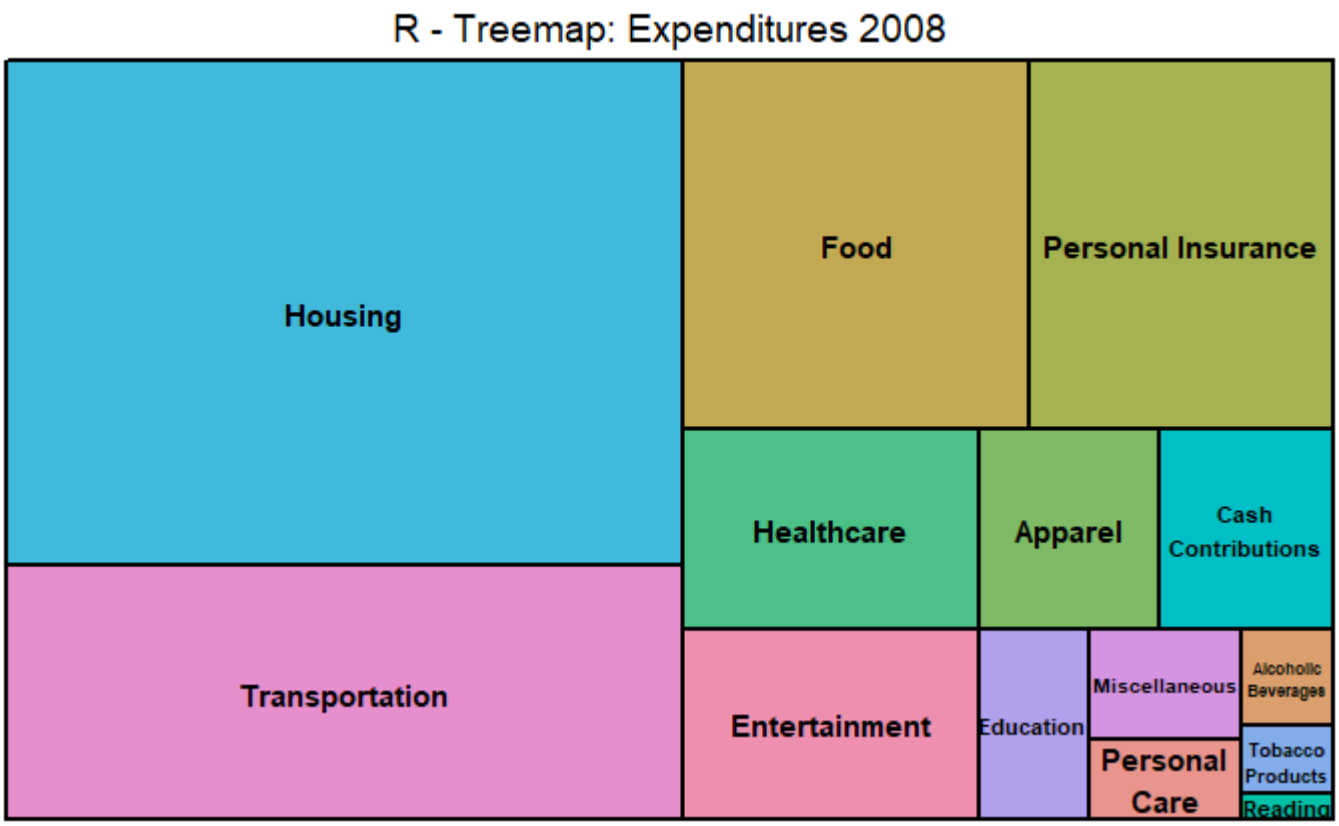


Hide

```
#filter data for treemap
tmd2 <- filter(data1, year == 2008)
```

Hide

```
#create another treemap
treemap(tmd2,
        index="category",
        vSize="expenditure",
        type="index",
        title ='R - Treemap: Expenditures 2008'
            )
```

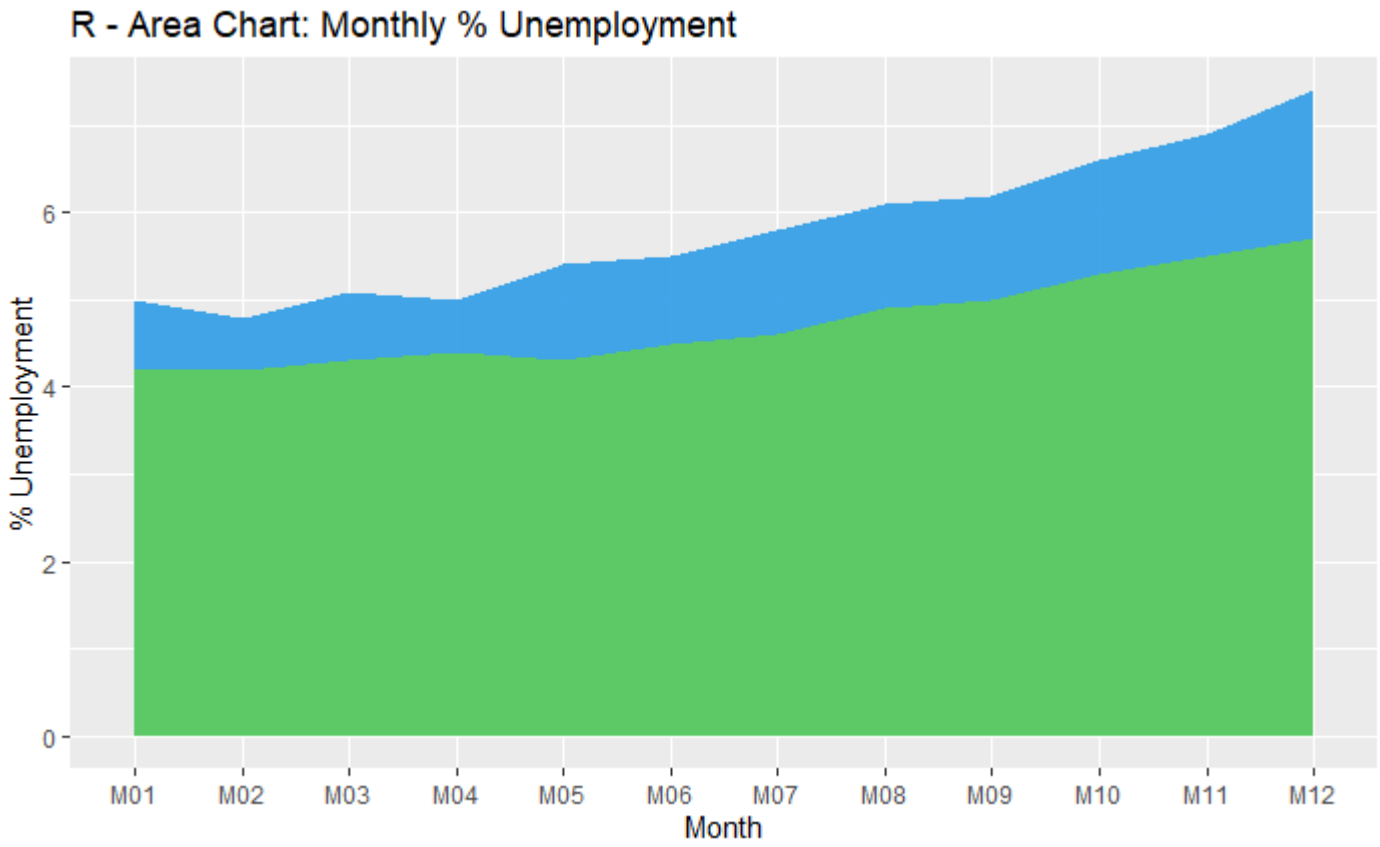


Hide

```
#create new dataframe
areadata1 <- filter(data2, Year == 2001)
areadata8 <- filter(data2, Year == 2008)
newdf <- data.frame(areadata1$Period, areadata1$Value, areadata8$Value)
names(newdf) <- c("Period","Year01","Year08")
```

Hide

```
#Don't use
# Area chart
ggplot(newdf) +
  geom_area(aes(x = Period, y = Year08, group=1),fill = 4, alpha = 0.85)+
  geom_area(aes(x = Period, y = Year01, group=1),fill = 3, alpha = 0.85)+
  xlab("Month")+ylab("% Unemployment")+
  ggtitle("R - Area Chart: Monthly % Unemployment")
```

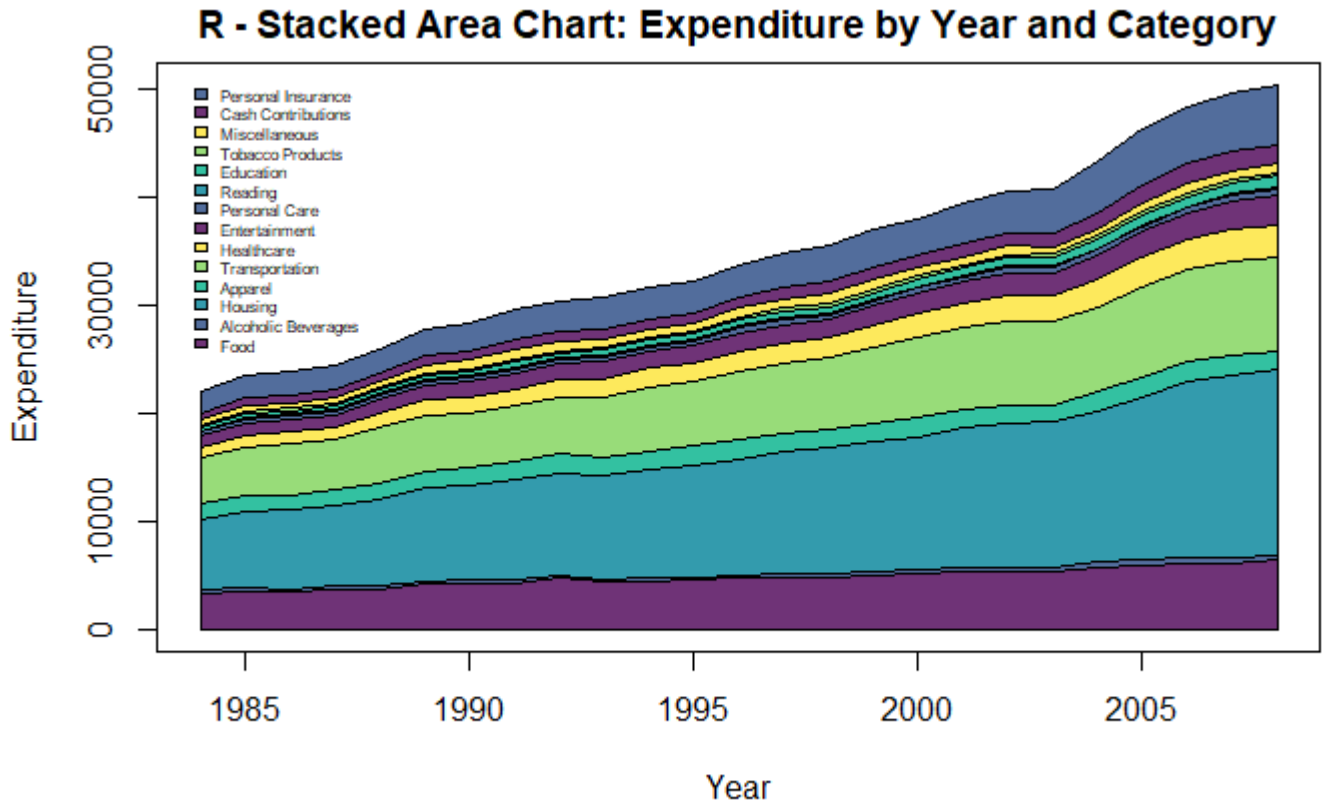


Hide

```
#pivot the data for the stacked area chart
pivot_data1 = pivot_wider(data1, names_from = category, values_from = expenditure)
drops = c("sex")
pivot_data1 = pivot_data1[ , !(names(pivot_data1) %in% drops)]
pivot_data1 <- pivot_data1[order(pivot_data1$year),]
```

Hide

```
#https://r-charts.com/evolution/stacked-area/

cols <- hcl.colors(6, palette = "viridis", alpha = 0.8)

areaplot(.~year, data = pivot_data1,
         main = "R - Stacked Area Chart: Expenditure by Year and Category",
         xlab = "Year",
         ylab = "Expenditure",
         col=cols,
         legend = TRUE,
         args.legend = list(x="topleft",cex=0.5))
```



Hide

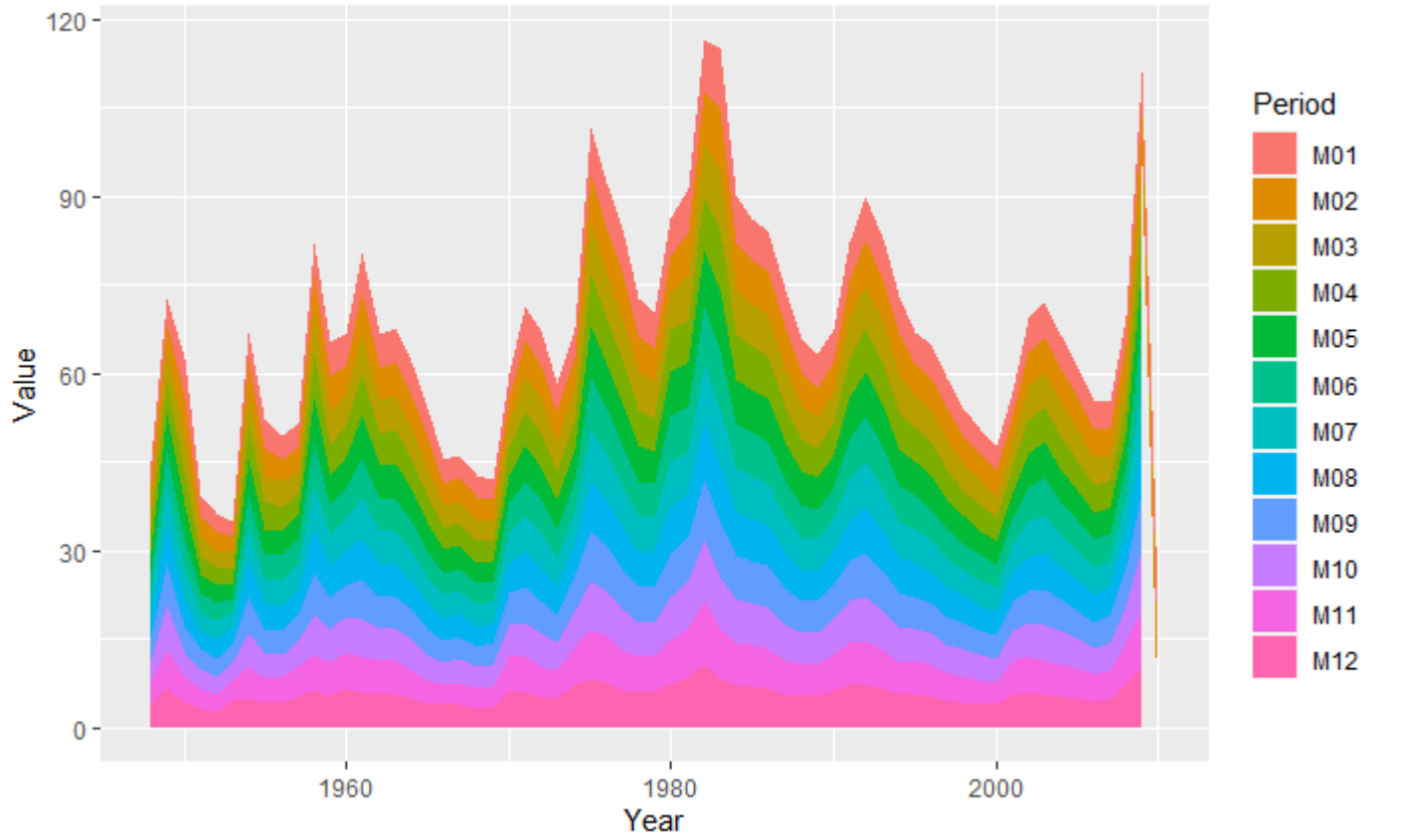
```
NA
NA
```

Hide

```
#adjust data
data2$Month = data2$Period #make new column
data2$Month <- gsub("[^0-9.-]", "", data2$Month)
data2$Month = as.numeric(as.character(data2$Month))
```

Hide

```
plot = ggplot(data2, aes(x=Year, y=Value, fill=Period))
plot + geom_area()
```

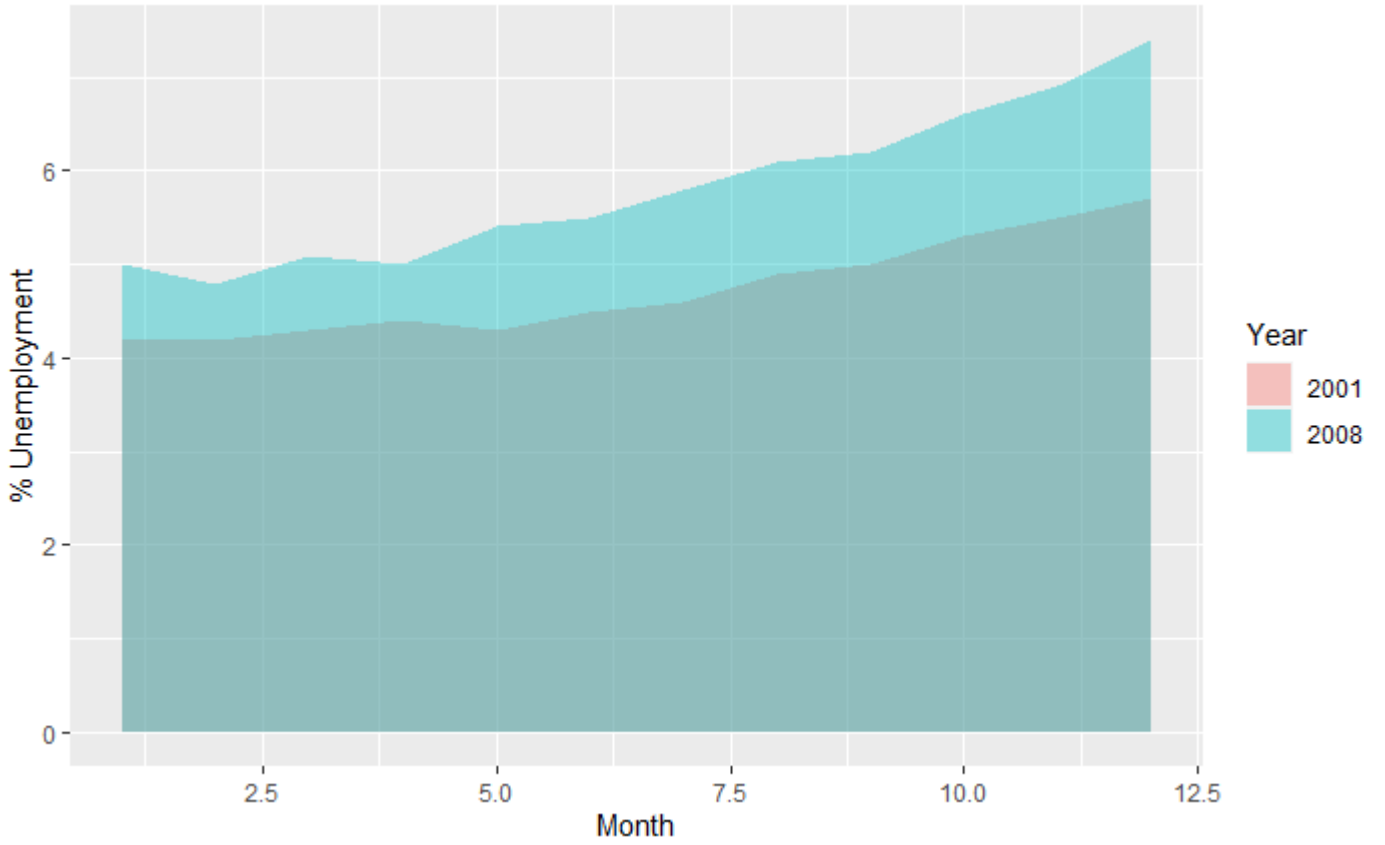


Hide

```
#area data again....
data2$Year = as.character(data2$Year)
areadf <- subset(data2,(Year == 2001 | Year == 2008))
```

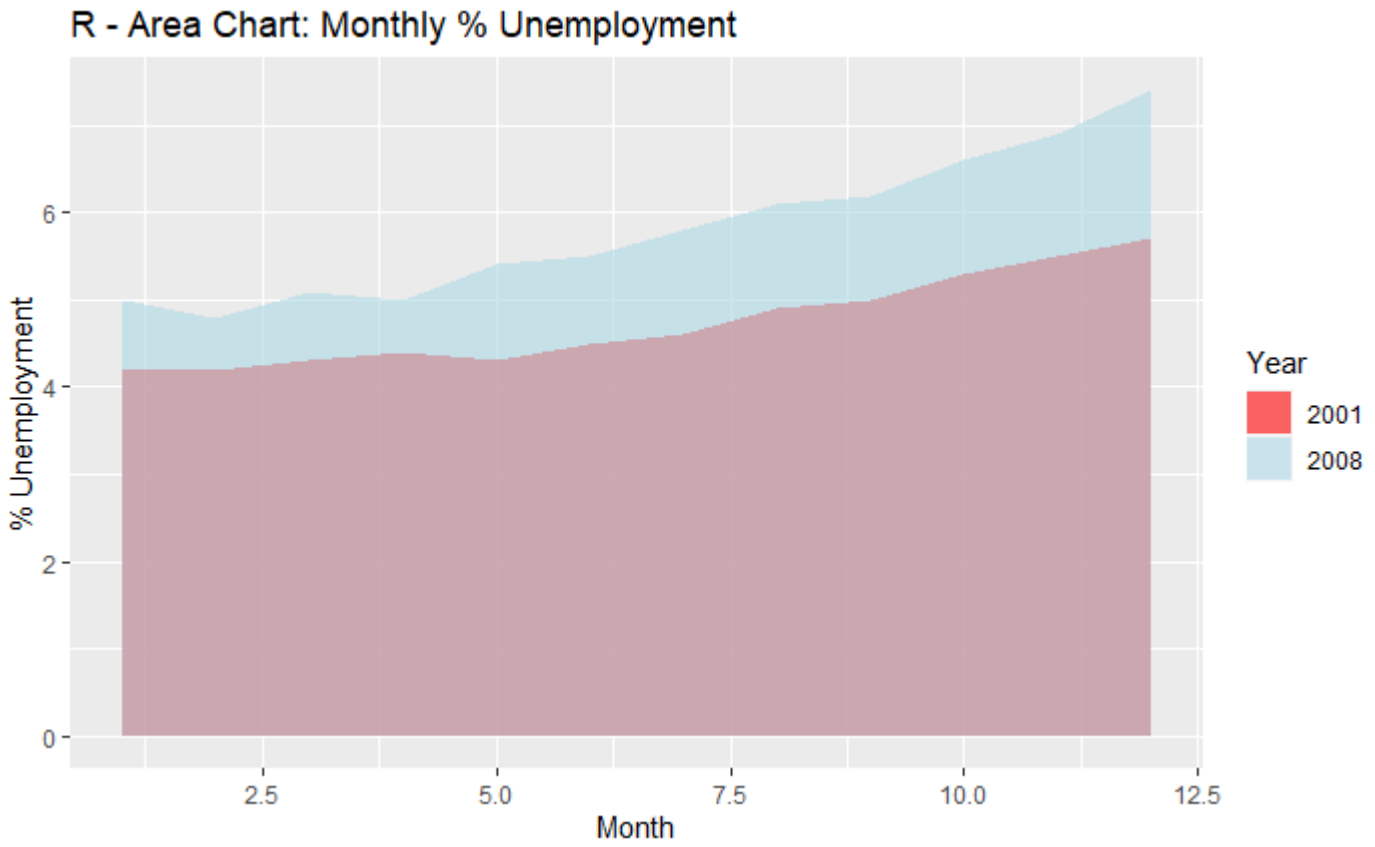
Hide

```
#Area plot retry
plot = ggplot(areadf, aes(x=Month, y=Value, fill=Year))
plot + geom_area(position = "identity",alpha=.4)+  xlab("Month")+ylab("% Unemployment")
```



Hide

```
plot = ggplot(areadf, aes(x=Month, y=Value, fill=Year))
plot + geom_area(position = "identity",alpha=.6)+  xlab("Month")+ylab("% Unemployment")+
ggtitle("R - Area Chart: Monthly % Unemployment")+scale_fill_manual(values = c('red','lightblue'
)) #this helps prevent overlapping and discoloration
```


R - Area Chart: Monthly % Unemployment
% Unemployment
0
2
4
6
2.5
5.0
7.5
10.0
12.5
Month
Year
2001
2008