数据探索与数据预处理(4)EDA_customers

覃雄派

数据探索与数据预处理
(4)EDA_customers

提纲

- 装载数据，查看前几行
- 缺失值
- 重复记录
- 描述性统计信息、与可视化
- encode age & gender
- Box plot for variables
- Scatter for variables
- 尝试k-means聚类算法
- Annual Income (k$) - Spending Score (1-100) scatter plot With respect to male and female
- Annual Income (k$) - Spending Score (1-100) scatter plot With respect to young and old

数据探索与数据预处理(4)EDA_customers

- 装载数据，查看前几行

load data

```
customers = pd.read_csv("customers.csv")
```

simple exploration

```
customers.head()
```

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)
0	1	Male	19	15	39
1	2	Male	21	15	81
2	3	Female	20	16	6
3	4	Female	23	16	77
4	5	Female	31	17	40

数据探索与数据预处理(4)EDA_customers

- 查看缺失值

```
print("Missing values in each variable: \n{}".format(customers.isnull().sum()))
```

```
Missing values in each variable:
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

数据探索与数据预处理(4)EDA_customers

- 重复记录

```
print("Duplicated rows: {}".format(customers.duplicated().sum()))
```

```
Duplicated rows: 0
```

数据探索与数据预处理(4)EDA_customers

- 数据类型

```
print("Variable:                    Type: \n{}".format(customers.dtypes))
```

```
Variable:                    Type:
CustomerID                    int64
Gender                       object
Age                           int64
Annual Income (k$)            int64
Spending Score (1-100)        int64
dtype: object
```

数据探索与数据预处理(4)EDA_customers

数据探索与数据预处理(4)EDA_customers

- 描述性统计信息
  - 辅助函数

```
def statistics(variable):
    if variable.dtype == "int64" or variable.dtype == "float64":
        return pd.DataFrame([[variable.name, np.mean(variable), np.std(variable), np.median(variable), np.var(variable)]],
                            columns = ["Variable", "Mean", "Standard Deviation", "Median", "Variance"]).set_index("Variable")
    else:
        return pd.DataFrame(variable.value_counts())
```

数据探索与数据预处理(4)EDA_customers

- 描述性统计信息
  - Numeric 变量的可视化辅助函数

```
def graph_histo(x):
    if x.dtype == "int64" or x.dtype == "float64":
        # Select size of bins by getting maximum and minimum and divide the substraction by 10
        size_bins = 10
        # Get the title by getting the name of the column
        title = x.name
        #Assign random colors to each graph
        color_kde = list(map(float, np.random.rand(3,)))
        color_bar = list(map(float, np.random.rand(3,)))

        # Plot the displot
        sns.distplot(x, bins=size_bins, kde_kws={"lw": 1.5, "alpha":0.8, "color":color_kde},
                     hist_kws={"linewidth": 1.5, "edgecolor": "grey",
                               "alpha": 0.4, "color":color_bar})
        # Customize ticks and labels
        plt.xticks(size=14)
        plt.yticks(size=14);
        plt.ylabel("Frequency", size=16, labelpad=15);
        # Customize title
        plt.title(title, size=18)
        # Customize grid and axes visibility
        plt.grid(False);
        plt.gca().spines["top"].set_visible(False);
        plt.gca().spines["right"].set_visible(False);
        plt.gca().spines["bottom"].set_visible(False);
        plt.gca().spines["left"].set_visible(False);
```

数据探索与数据预处理(4)EDA_customers

- 描述性统计信息
  - Category 变量的可视化辅助函数

```
else:
    x = pd.DataFrame(x)
    # Plot
    sns.catplot(x=x.columns[0], kind="count", palette="spring", data=x)
    # Customize title
    title = x.columns[0]
    plt.title(title, size=18)
    # Customize ticks and labels
    plt.xticks(size=14)
    plt.yticks(size=14);
    plt.xlabel("")
    plt.ylabel("Counts", size=16, labelpad=15);
    # Customize grid and axes visibility
    plt.gca().spines["top"].set_visible(False);
    plt.gca().spines["right"].set_visible(False);
    plt.gca().spines["bottom"].set_visible(False);
    plt.gca().spines["left"].set_visible(False);
```

数据探索与数据预处理(4)EDA_customers

- Spending score的统计信息与可视化

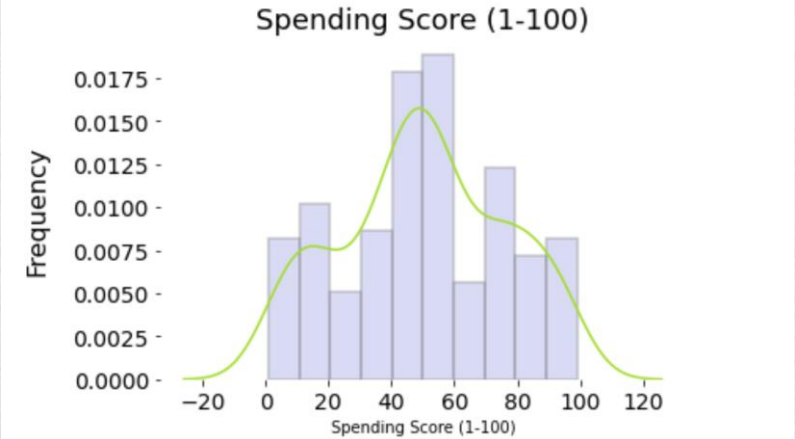
```
spending = customers["Spending Score (1-100)"]
statistics(spending)
```

Variable	Mean	Standard Deviation	Median	Variance
Spending Score (1-100)	50.2	25.758882	50.0	663.52

数据探索与数据预处理(4)EDA_customers

- Spending score的统计信息与可视化

```
graph_histo(spending)
```



数据探索与数据预处理(4)EDA_customers

- Age的统计信息与可视化

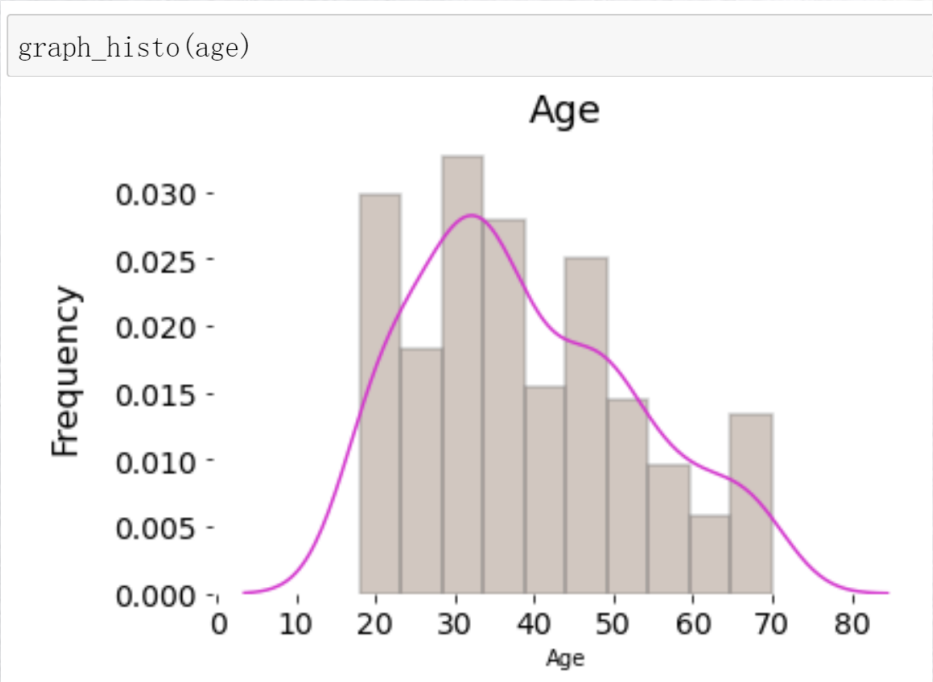
```
age = customers["Age"]
statistics(age)
```

Variable	Mean	Standard Deviation	Median	Variance
Age	38.85	13.934041	36.0	194.1575

数据探索与数据预处理(4)EDA_customers

- Age的统计信息与可视化

```
graph_histo(age)
```



数据探索与数据预处理(4)EDA_customers

- income的统计信息与可视化

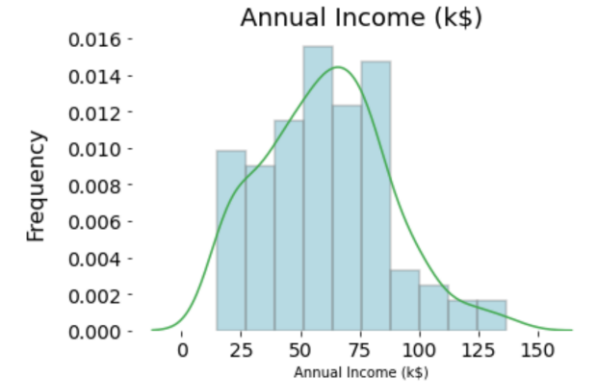
```
income = customers["Annual Income (k$)"]
statistics(income)
```

Variable	Mean	Standard Deviation	Median	Variance
Annual Income (k$)	60.56	26.198977	61.5	686.3864

数据探索与数据预处理(4)EDA_customers

- income的统计信息与可视化

```
graph_histo(income)
```



数据探索与数据预处理(4)EDA_customers

- Gender的统计信息与可视化

```
gender = customers["Gender"]
#print(gender)


count_Male = len(customers[customers['Gender'].isin(['Male'])] )
print ("count_Male",(count_Male))
count_Female = len (customers[customers['Gender'].isin(['Female'])] )
print ("count_Female",(count_Female))
```

```
count_Male 88
count_Female 112
```

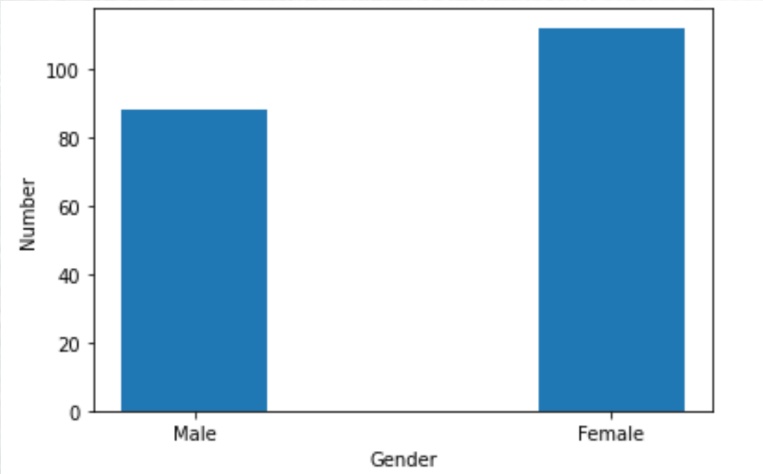
```
statistics(gender)
```

	Gender
Female	112
Male	88

数据探索与数据预处理(4)EDA_customers

- gender的统计信息与可视化

```
import matplotlib.pyplot as plt

plt.xlabel("Gender")
plt.ylabel("Number")

plt.xticks((0,1),("Male","Female"))
xlist = [0,1]
ylist=[count_Male,count_Female]

plt.bar(x=xlist,height=ylist,width = 0.35,align="center")

plt.show()
```



数据探索与数据预处理(4)EDA_customers

数据探索与数据预处理(4)EDA_customers

- **encode age & gender**

```
def encode_Age(old):
    if old <=35:
        return 0
    else:
        return 1

one_column = customers["Age"]
one_column = one_column.apply(encode_Age )
customers["Age2"] =one_column

customers.head()
```

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	Age2
0	1	Male	19	15	39	0
1	2	Male	21	15	81	0
2	3	Female	20	16	6	0
3	4	Female	23	16	77	0
4	5	Female	31	17	40	0

数据探索与数据预处理(4)EDA_customers

- **encode age & gender**

```
def encode_Gender(old):
    if old == 'Male':
        return 0
    elif old == 'Female':
        return 1
    else:
        return 0

one_column = customers["Gender"]
one_column = one_column.apply(encode_Gender )
customers["Gender2"] =one_column

customers.head()
```

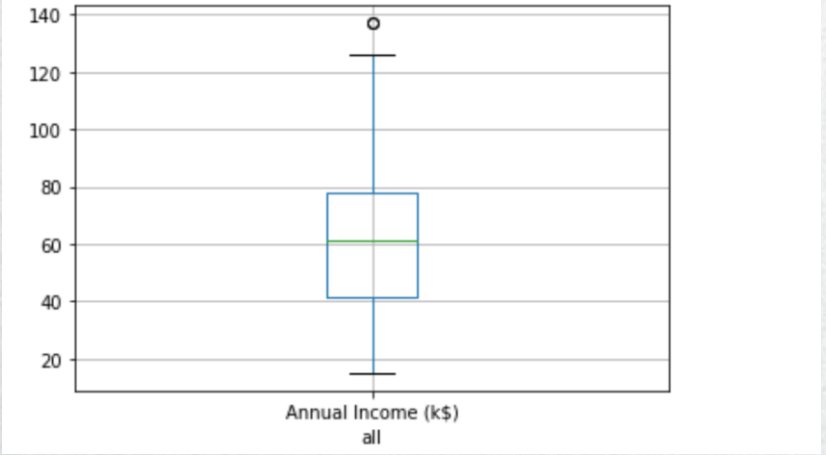
	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	Age2	Gender2
0	1	Male	19	15	39	0	0
1	2	Male	21	15	81	0	0
2	3	Female	20	16	6	0	1
3	4	Female	23	16	77	0	1
4	5	Female	31	17	40	0	1

数据探索与数据预处理(4)EDA_customers

数据探索与数据预处理(4)EDA_customers

- Box plot for variables
  - Income

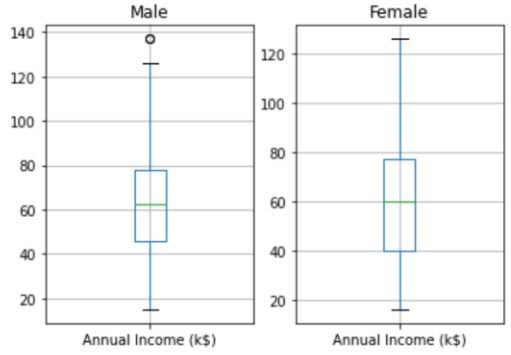
```
#box
customers.boxplot(column=["Annual Income (k$)"],showfliers=True)
plt.xlabel("all")
plt.show()
```



数据探索与数据预处理(4)EDA_customers

- Box plot for variables
  - Income group by gender

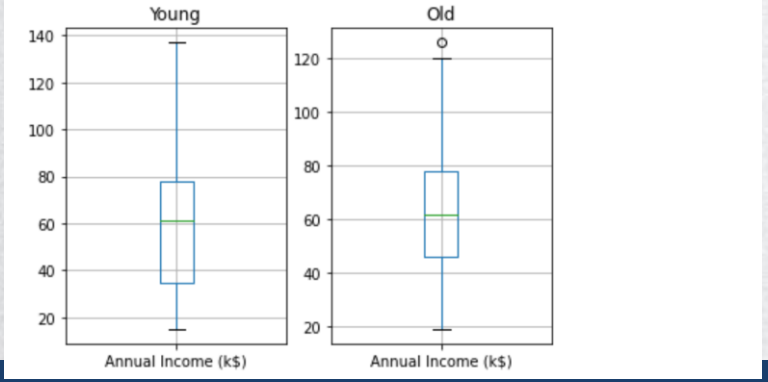
```
df_male = customers[customers.Gender =='Male']
df_female = customers[customers.Gender == 'Female']


fig, axes = plt.subplots(1,2) # create figure and axes
df_male.boxplot(column=["Annual Income (k$)"],    ax=axes.flatten()[0])
axes.flatten()[0].set_title("Male")

df_female.boxplot(column=["Annual Income (k$)"],    ax=axes.flatten()[1])
axes.flatten()[1].set_title("Female")

plt.show()
```



数据探索与数据预处理(4)EDA_customers

- Box plot for variables
  - Income group by age

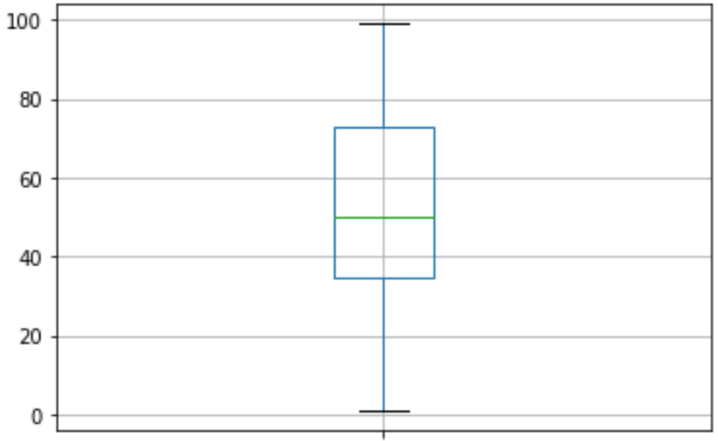
```
df_yound = customers[customers.Age <=35]
df_old = customers[customers.Age >35]

fig, axes = plt.subplots(1,2) # create figure and axes
df_yound.boxplot(column=["Annual Income (k$)"],    ax=axes.flatten()[0])
axes.flatten()[0].set_title("Young")

df_old.boxplot(column=["Annual Income (k$)"],    ax=axes.flatten()[1])
axes.flatten()[1].set_title("Old")

plt.show()
```



数据探索与数据预处理(4)EDA_customers

- Box plot for variables
  - Spending score

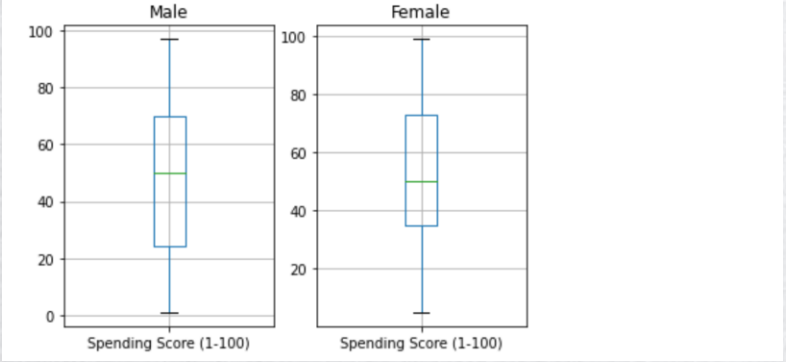
```
customers.boxplot(column=["Spending Score (1-100)"],showfliers=True)
plt.xlabel("all")
plt.show()
```



数据探索与数据预处理(4)EDA_customers

- Box plot for variables
  - Spending score group by gender

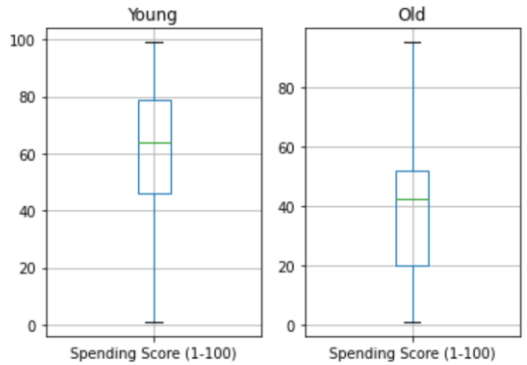
```
df_male = customers[customers.Gender =='Male']
df_female = customers[customers.Gender == 'Female']


fig, axes = plt.subplots(1,2) # create figure and axes
df_male.boxplot(column=["Spending Score (1-100)"],    ax=axes.flatten()[0])
axes.flatten()[0].set_title("Male")

df_female.boxplot(column=["Spending Score (1-100)"],    ax=axes.flatten()[1])
axes.flatten()[1].set_title("Female")

plt.show()
```



数据探索与数据预处理(4)EDA_customers

- Box plot for variables
  - Spending score group by age

```
df_yound = customers[customers.Age <=35]
df_old = customers[customers.Age >35]

fig, axes = plt.subplots(1,2) # create figure and axes
df_yound.boxplot(column=["Spending Score (1-100)"],    ax=axes.flatten()[0])
axes.flatten()[0].set_title("Young")

df_old.boxplot(column=["Spending Score (1-100)"],    ax=axes.flatten()[1])
axes.flatten()[1].set_title("Old")

plt.show()
```

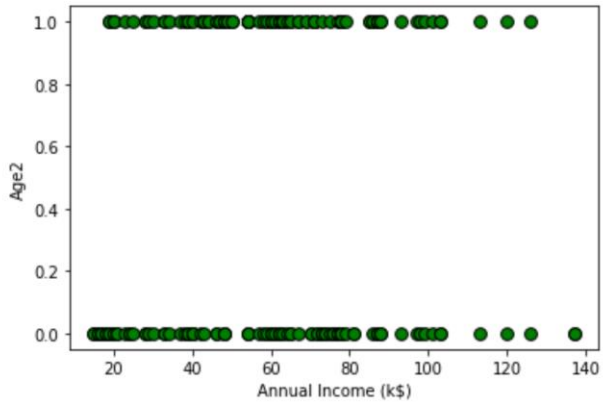


数据探索与数据预处理(4)EDA_customers

数据探索与数据预处理(4)EDA_customers

- Scatter for variables
  - Annual income vs. age groups

```
xx = customers["Annual Income (k$)"]
yy = customers["Age2"]
plt.scatter(xx, yy, 60, edgecolors='black', c='green')

plt.xlabel("Annual Income (k$)")
plt.ylabel("Age2")
#plt.legend()
plt.show()
```



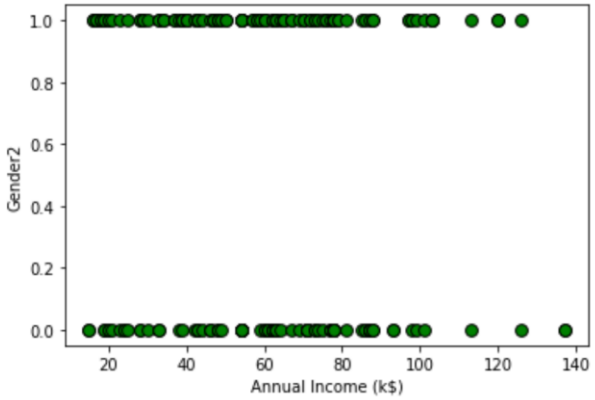
老年人

青年人

数据探索与数据预处理(4)EDA_customers

- Scatter for variables
  - Annual income vs. gender groups

```
xx = customers["Annual Income (k$)"]
yy = customers["Gender2"]
plt.scatter(xx, yy, 60, edgecolors='black', c='green')

plt.xlabel("Annual Income (k$)")
plt.ylabel("Gender2")
#plt.legend()
plt.show()
```



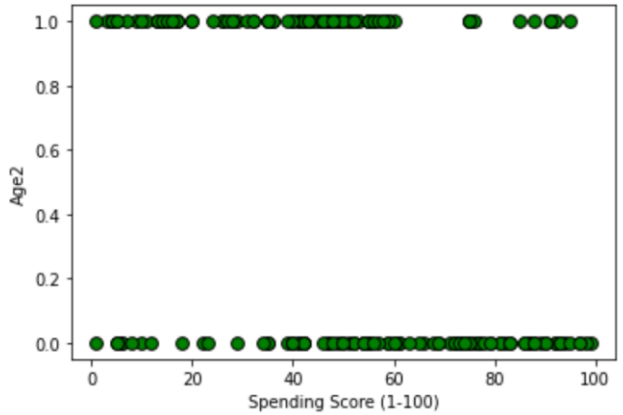
女

男

数据探索与数据预处理(4)EDA_customers

- Scatter for variables
  - Spending score vs. age groups

```
xx = customers["Spending Score (1-100)"]
yy = customers["Age2"]
plt.scatter(xx, yy, 60, edgecolors='black', c='green')

plt.xlabel("Spending Score (1-100)")
plt.ylabel("Age2")
#plt.legend()
plt.show()
```



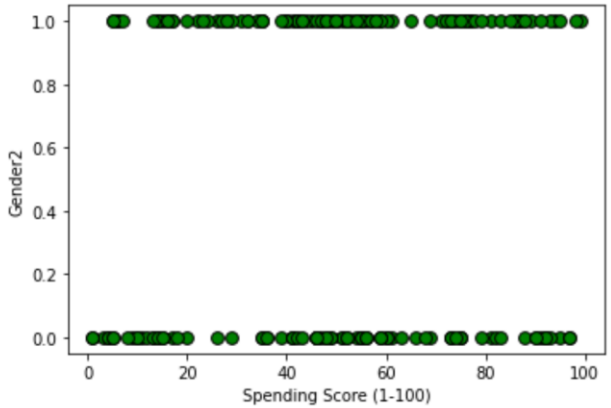
老年人

青年人

数据探索与数据预处理(4)EDA_customers

- Scatter for variables
  - Spending score vs. gender groups

```
xx = customers["Spending Score (1-100)"]
yy = customers["Gender2"]
plt.scatter(xx, yy, 60, edgecolors='black', c='green')

plt.xlabel("Spending Score (1-100)")
plt.ylabel("Gender2")
#plt.legend()
plt.show()
```



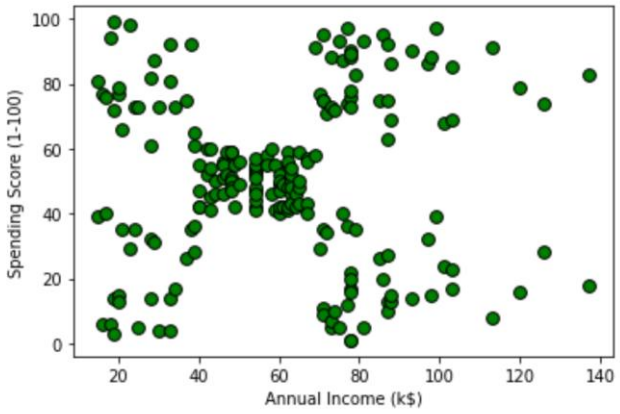
女

男

数据探索与数据预处理(4)EDA_customers

- Annual Income (k$) - Spending Score (1-100) scatter

```
xx = customers["Annual Income (k$)"]
yy = customers["Spending Score (1-100)"]
plt.scatter(xx, yy, 60, edgecolors='black', c='green')

plt.xlabel("Annual Income (k$)")
plt.ylabel("Spending Score (1-100)")
#plt.legend()
plt.show()
```



似乎有些聚拢的群组

数据探索与数据预处理(4)EDA_customers

数据探索与数据预处理(4)EDA_customers

- 尝试k-means
  - 切割数据，只取两列

```
customers.head()
```

	CustomerID	Gender	Age	Annual Income (k$)	Spending Score (1-100)	Age2	Gender2
0	1	Male	19	15	39	0	0
1	2	Male	21	15	81	0	0
2	3	Female	20	16	6	0	1
3	4	Female	23	16	77	0	1
4	5	Female	31	17	40	0	1

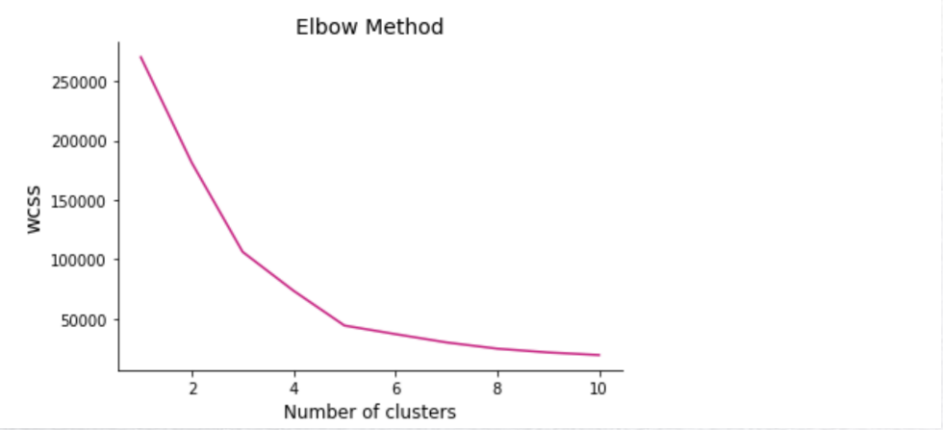
```
X = customers[["Annual Income (k$)","Spending Score (1-100)"]]
X.head()
```

	Annual Income (k$)	Spending Score (1-100)
0	15	39
1	15	81
2	16	6
3	16	77
4	17	40

数据探索与数据预处理(4)EDA_customers

- 尝试k-means
  - 多大的K
  - 合适呢?

```
wcss = []
for i in range(1,11):
    km = KMeans(n_clusters=i,init='k-means++', max_iter=300, n_init=10, random_state=0)
    km.fit(X)
    wcss.append(km.inertia_)
plt.plot(range(1,11),wcss, c="#c51b7d")
plt.gca().spines["top"].set_visible(False)
plt.gca().spines["right"].set_visible(False)
plt.title('Elbow Method', size=14)
plt.xlabel('Number of clusters', size=12)
plt.ylabel('wcss', size=14)
plt.show()
```



数据探索与数据预处理(4)EDA_customers

- 尝试k-means
  - K=5
  - 训练并且预测

```
kmeans = KMeans(n_clusters=5, init='k-means++', max_iter=10, n_init=10, random_state=0)

# Fit and predict
cluster_label = kmeans.fit_predict(X)
```

```
print(cluster_label)
```

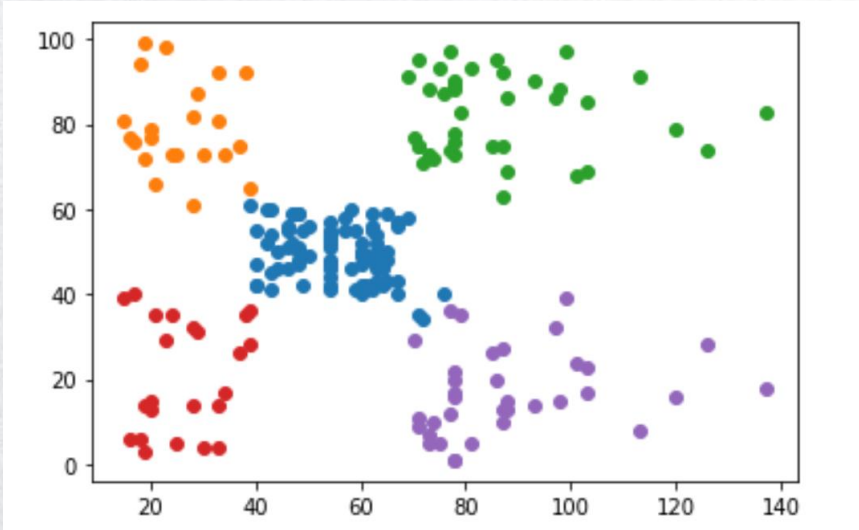
```
[3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1
 1 3 1 3 1 3 0 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 0 0 0 0 0 0 0 0 0 2 4 2 0 2 4 2 4 2 0 2 4 2 4 2 4 2 0 2 4 2 4 2
 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2 4
 2 4 2 4 2 4 2 4 2 4 2 4 2 4 2]
```

数据探索与数据预处理(4)EDA_customers

- 尝试k-means
  - 类簇可视化

```
xx = X.values
```

```
import matplotlib.pyplot as plt

one_cluster = xx[cluster_label == 0]
plt.scatter(one_cluster[:,0] , one_cluster[:,1])

one_cluster = xx[cluster_label == 1]
plt.scatter(one_cluster[:,0] , one_cluster[:,1])

one_cluster = xx[cluster_label == 2]
plt.scatter(one_cluster[:,0] , one_cluster[:,1])

one_cluster = xx[cluster_label == 3]
plt.scatter(one_cluster[:,0] , one_cluster[:,1])

one_cluster = xx[cluster_label == 4]
plt.scatter(one_cluster[:,0] , one_cluster[:,1])

plt.show()
```



数据探索与数据预处理(4)EDA_customers

- 尝试k-means
  - 显示类簇中心

```
centroids = pd.DataFrame(kmeans.cluster_centers_, columns = [  "Annual Income", "Spending"])
centroids.index_name = "ClusterID"
centroids["ClusterID"] = centroids.index
centroids = centroids.reset_index(drop=True)
centroids
```

	Annual Income	Spending	ClusterID
0	55.296296	49.518519	0
1	25.727273	79.363636	1
2	86.538462	82.128205	2
3	26.304348	20.913043	3
4	88.200000	17.114286	4

数据探索与数据预处理(4)EDA_customers

- 尝试k-means
  - 新数据点应该归入什么类簇呢?

```
X_new = np.array([[50.0, 50.0]])

new_customer = kmeans.predict(X_new)
print("The new customer belongs to segment {}".format(new_customer[0]))
```

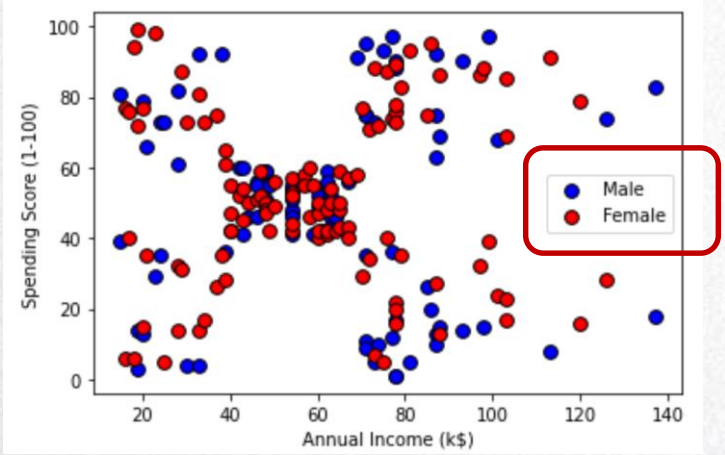
```
The new customer belongs to segment 0
```

数据探索与数据预处理(4)EDA_customers

数据探索与数据预处理(4)EDA_customers

- 进一步探索
  - Annual Income (k$) - Spending Score (1-100) scatter plot
  - With respect to male and female

```
df_male = customers[customers.Gender =='Male']
df_female = customers[customers.Gender == 'Female']

x_male = df_male["Annual Income (k$)"]
y_male = df_male["Spending Score (1-100)"]
x_female = df_female["Annual Income (k$)"]
y_female = df_female["Spending Score (1-100)"]


plt.scatter(x_male, y_male, 60, edgecolors='black', c='blue', label = 'Male')
plt.scatter(x_female, y_female, 60, edgecolors='black', c='red', label = 'Female')

plt.xlabel("Annual Income (k$)")
plt.ylabel("Spending Score (1-100)")
plt.legend()
plt.show()
```

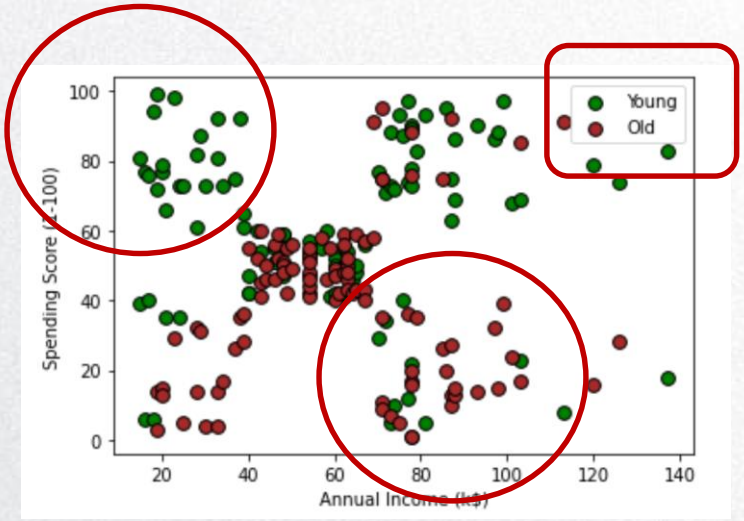


数据探索与数据预处理(4)EDA_customers

- 进一步探索
  - Annual Income (k$) - Spending Score (1-100) scatter plot
  - With respect to young and old

这两组好像有点区别呀？

```
df_young = customers[customers.Age <=35]
df_old = customers[customers.Age >35]

x_young = df_young["Annual Income (k$)"]
y_young = df_young["Spending Score (1-100)"]
x_old = df_old["Annual Income (k$)"]
y_old = df_old["Spending Score (1-100)"]


plt.scatter(x_young, y_young, 60, edgecolors='black', c='green', label = 'Young')
plt.scatter(x_old, y_old, 60, edgecolors='black', c='brown', label = 'Old')

plt.xlabel("Annual Income (k$)")
plt.ylabel("Spending Score (1-100)")
plt.legend()
plt.show()
```



数据探索与数据预处理(4)EDA_customers

数据探索与数据预处理(4)EDA_customers

- Note book中还提供了
  - 对数据进行降维、K-means聚类、可视化的代码
  - 请打开note book研究

数据探索与数据预处理(4)EDA_customers