```
# %%
import numpy as np
import math
import matplotlib.pyplot as plt
import pandas as pd
from matplotlib.animation import FuncAnimation
from mpl_toolkits.mplot3d import Axes3D
from IPython.display import HTML
import matplotlib.cm as cm

# %%

def get_random_K(arr1, K):
    random_values = []
    for _ in range(K):
        random_value = np.random.uniform(np.min(arr1), np.max(arr1))
        random_values.append(random_value)

    result = np.array(random_values)
    return result

# %%
def get_distance(x,y,a,b):
    return np.sqrt(( x - a )**2 + ( y - b )**2)

# %%
def arrange_spot(arrx, arry, point_x, poiny_y):
    K_point_list = []
    for i in range(len(arrx)):
        temp = []
        for j in range(len(point_x)):
            distance = get_distance(arrx[i], arry[i], point_x[j], poiny_y[j])
            distance = round(distance,2)
            temp.append(distance)
        min_index = int(np.argmin(temp))
        K_point_list.append(min_index)
    return K_point_list


# %%
def new_point(arrx, arry, list,K):
    mid_x = []
    mid_y = []
    for i in range(K):
        temp = []
        for j in range(len(arrx)):
            if(list[j] == i):
```

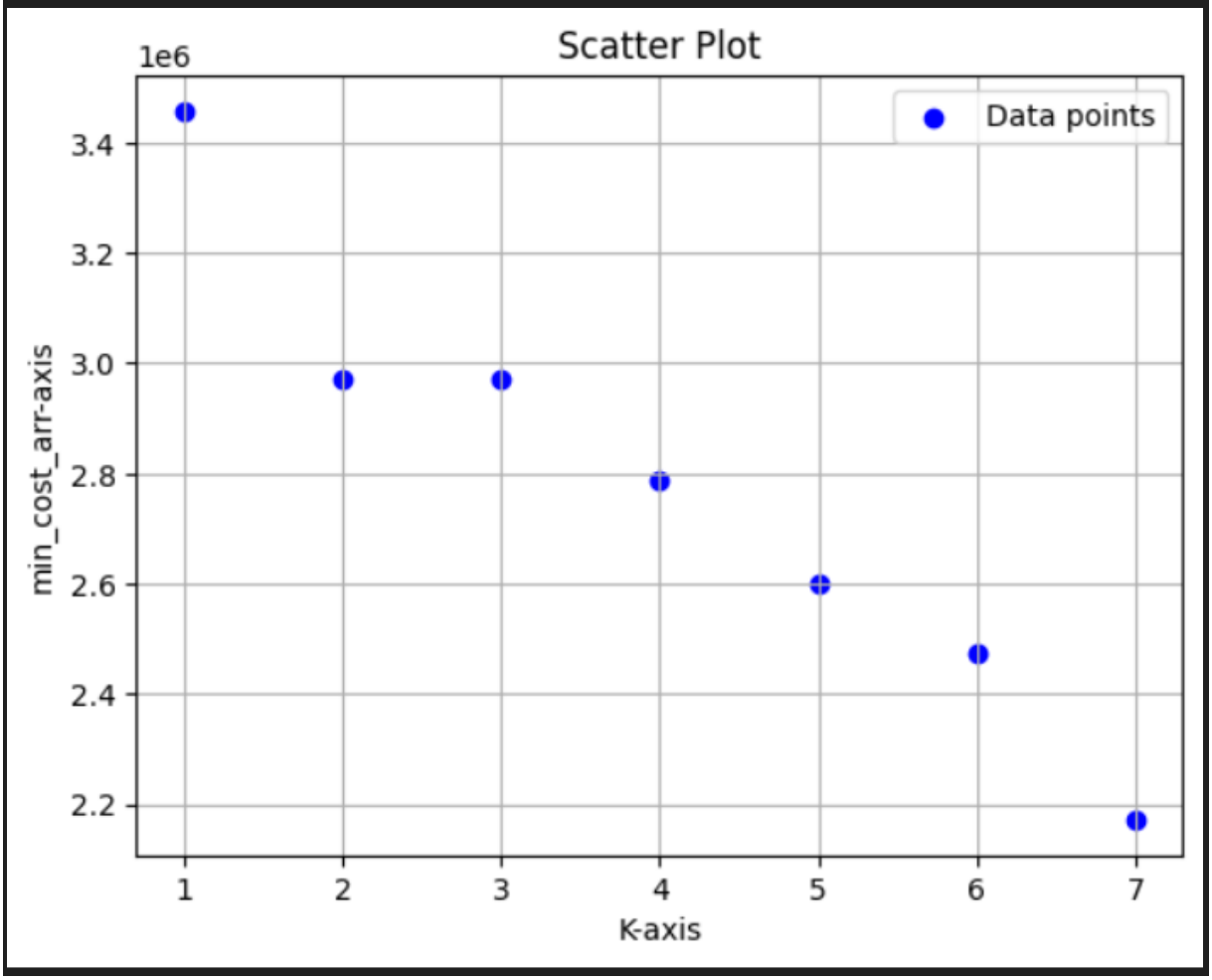
```
                temp.append(arrx[j])
        average_K_x = sum(temp)/(len(temp) + 0.001)
        mid_x.append(average_K_x)
    for i in range(K):
        temp = []
        for j in range(len(arry)):
            if(list[j] == i):
                temp.append(arry[j])
        average_K_y = sum(temp)/(len(temp) + 0.001)
        mid_y.append(average_K_y)

    return mid_x, mid_y


# %%
def cost_function(arrx, arry, x_new_point, y_new_point, list, K):
    cost_arr = []
    for i in range(K):
        average = []
        for j in range(len(arrx)):
            if(list[j] == i):
                temp = get_distance(arrx[j],
arry[j],x_new_point[i],y_new_point[i])
                average.append(temp)
        average = sum(average)
        cost_arr.append(average)
    return cost_arr

# %%
def find_elbow_spot(K, min_cost_arr):
    angles = []
    for i in range(1, len(K) - 1):
        A = np.array([K[i - 1], min_cost_arr[i - 1]])
        B = np.array([K[i], min_cost_arr[i]])
        C = np.array([K[i + 1], min_cost_arr[i + 1]])

        AB = B - A
        BC = C - B
        cos_ab_bc = (AB[0] * BC[0] + AB[1] * BC[1]) / (math.sqrt(AB[0]**2 +
AB[1]**2) * math.sqrt(BC[0]**2 + BC[1]**2)) # góc giữa 2 vecto
        angle = math.degrees(math.acos(cos_ab_bc))
        print(angle)
        angles.append(angle)
    for i in range(len(angles)):
        if(angles[i]> 70):
            return i
    return -1
```

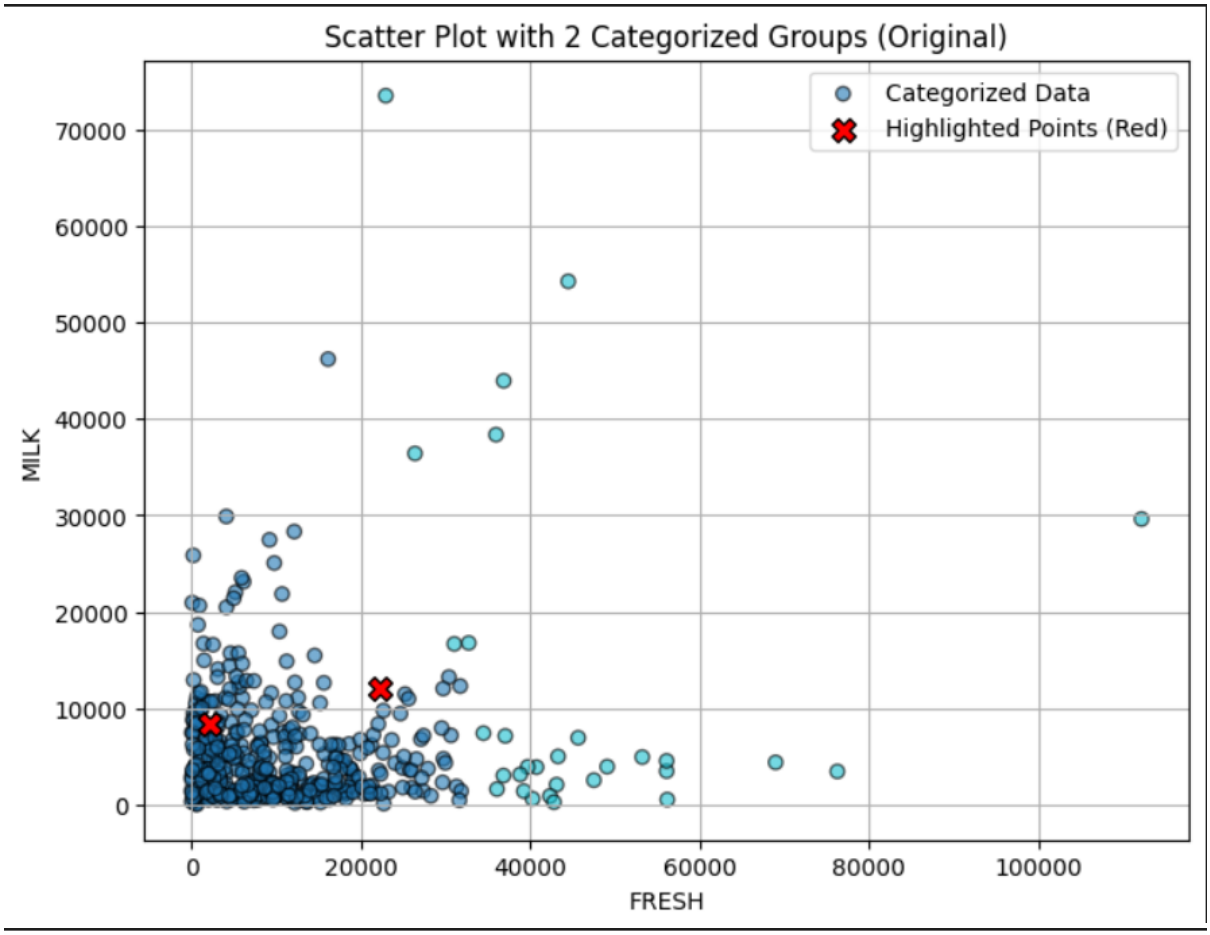
```
# %%
def cluster(Xdata,Ydata,iterations, var_K):
    min_cost = 999999999
    min_cost_arr = []
    K_min = 0
    min_X = 0
    min_Y = 0
    min_x_new_point = 0
    min_y_new_point = 0
    K=1
    for i in  range(7):
        if(var_K == 0 ):
            K = K + 1
        else:
            K = var_K
        for j in range(iterations):
            X = get_random_K(Xdata,K)
            Y = get_random_K(Ydata,K)
            point_in_group = arrange_spot(Xdata, Ydata, X, Y)
            x_new_point, y_new_point = new_point(Xdata, Ydata,
point_in_group,K)
            temp = cost_function(Xdata, Ydata, x_new_point, y_new_point,
point_in_group, K)
            temp = sum(temp)
            if (min_cost >= temp):
                min_cost = temp
                K_min = K
                min_X = X
                min_Y = Y
                min_x_new_point = x_new_point
                min_y_new_point = y_new_point
        min_cost_arr.append(min_cost)
    return K_min, min_X, min_Y, min_x_new_point, min_y_new_point,
point_in_group, min_cost,min_cost_arr




# %%
# Đọc dữ liệu
data = pd.read_csv('kmeans_dataset.csv', delimiter=";")
iterations = 300

# Lấy dữ liệu đầu vào và đầu ra
Xdata = data['Fresh'].values
```

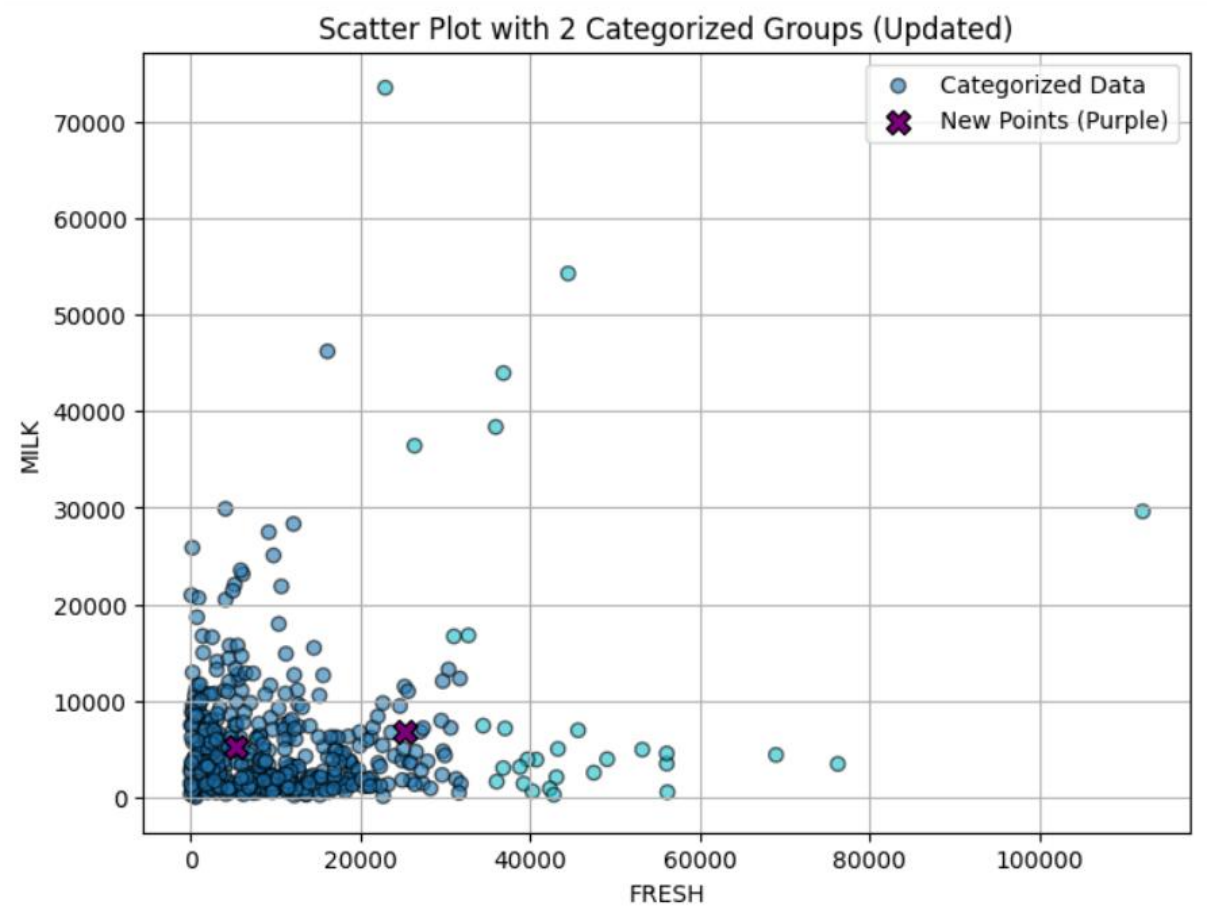
```
Ydata = data['Milk'].values

elbow_spot = -1
while(elbow_spot == -1):
    K, X, Y, x_new_point, y_new_point, point_in_group, min_cost,min_cost_arr =
cluster(Xdata, Ydata, iterations,0)
    elbow_spot = find_elbow_spot(list(range(1, K)),min_cost_arr)
    if(elbow_spot != -1):
        elbow_spot +=2

plt.scatter(list(range(1, K)), min_cost_arr, color='blue', marker='o',
label='Data points')
plt.xlabel("K-axis")
plt.ylabel("min_cost_arr-axis")
plt.title("Scatter Plot")
plt.legend()
plt.grid(True)
plt.show()

K, X, Y, x_new_point, y_new_point, point_in_group, min_cost,min_cost_arr =
cluster(Xdata, Ydata, iterations,elbow_spot)
#print(min_cost)

colormap = cm.get_cmap('tab10' if K <= 10 else 'gist_ncar', K)

# Tạo màu cho từng nhóm
unique_groups = sorted(set(point_in_group))  # Lấy danh sách nhóm theo thứ tự
colors = {group: colormap(i / (K - 1)) for i, group in
enumerate(unique_groups)}

category_colors = [colors[c] for c in point_in_group]



plt.figure(figsize=(8, 6))
plt.scatter(Xdata, Ydata, c=category_colors, alpha=0.6, edgecolors='black',
label="Categorized Data")
plt.scatter(X, Y, c='red', s=100, edgecolors='black', marker='X',
label="Highlighted Points (Red)")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title(f"Scatter Plot with {K} Categorized Groups (Original)")
plt.legend()
plt.grid(True)
plt.show()



plt.figure(figsize=(8, 6))
```

```
plt.scatter(Xdata, Ydata, c=category_colors, alpha=0.6, edgecolors='black',
label="Categorized Data")
plt.scatter(x_new_point, y_new_point, c='purple', s=100, edgecolors='black',
marker='X', label="New Points (Purple)")
plt.xlabel("X-axis")
plt.ylabel("Y-axis")
plt.title(f"Scatter Plot with {K} Categorized Groups (Updated)")
plt.legend()
plt.grid(True)
plt.show()
```

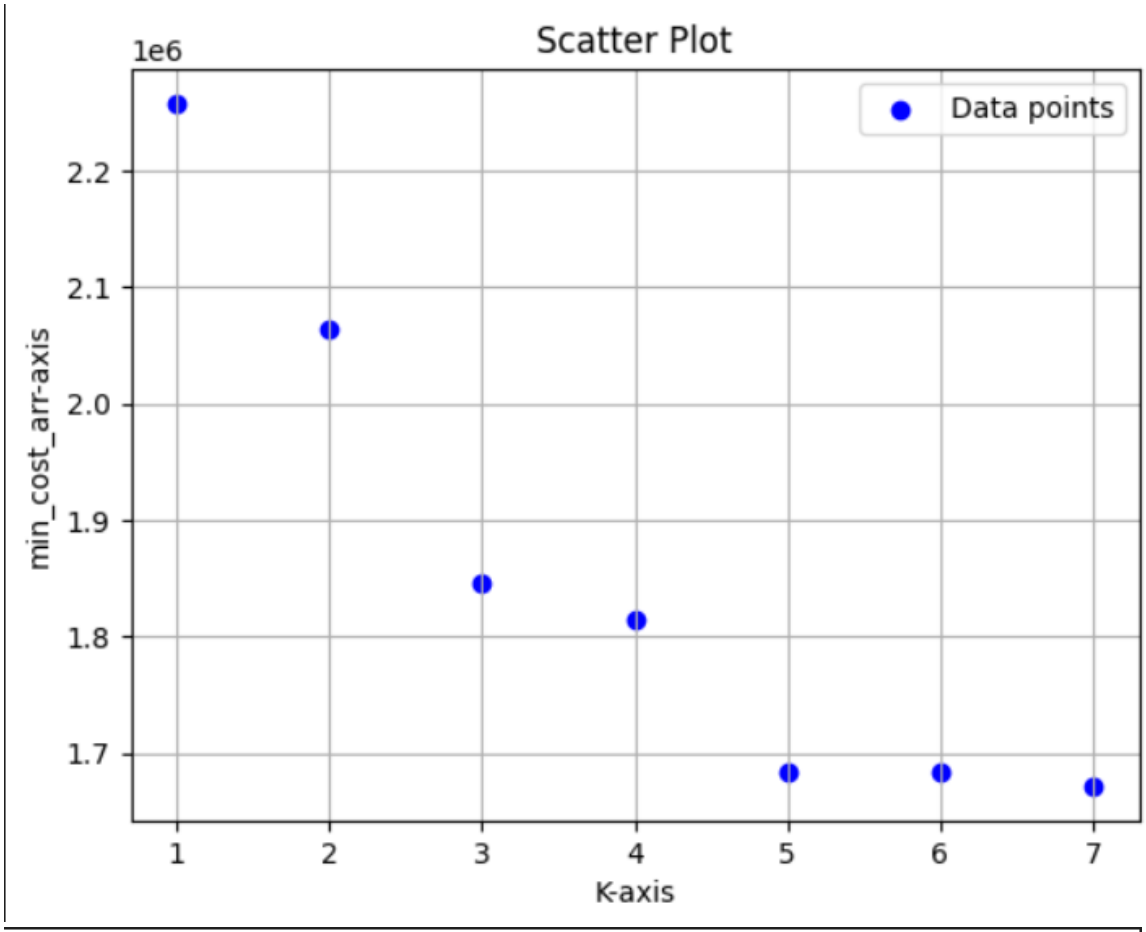
EXAMPLE : FRESH and MILK

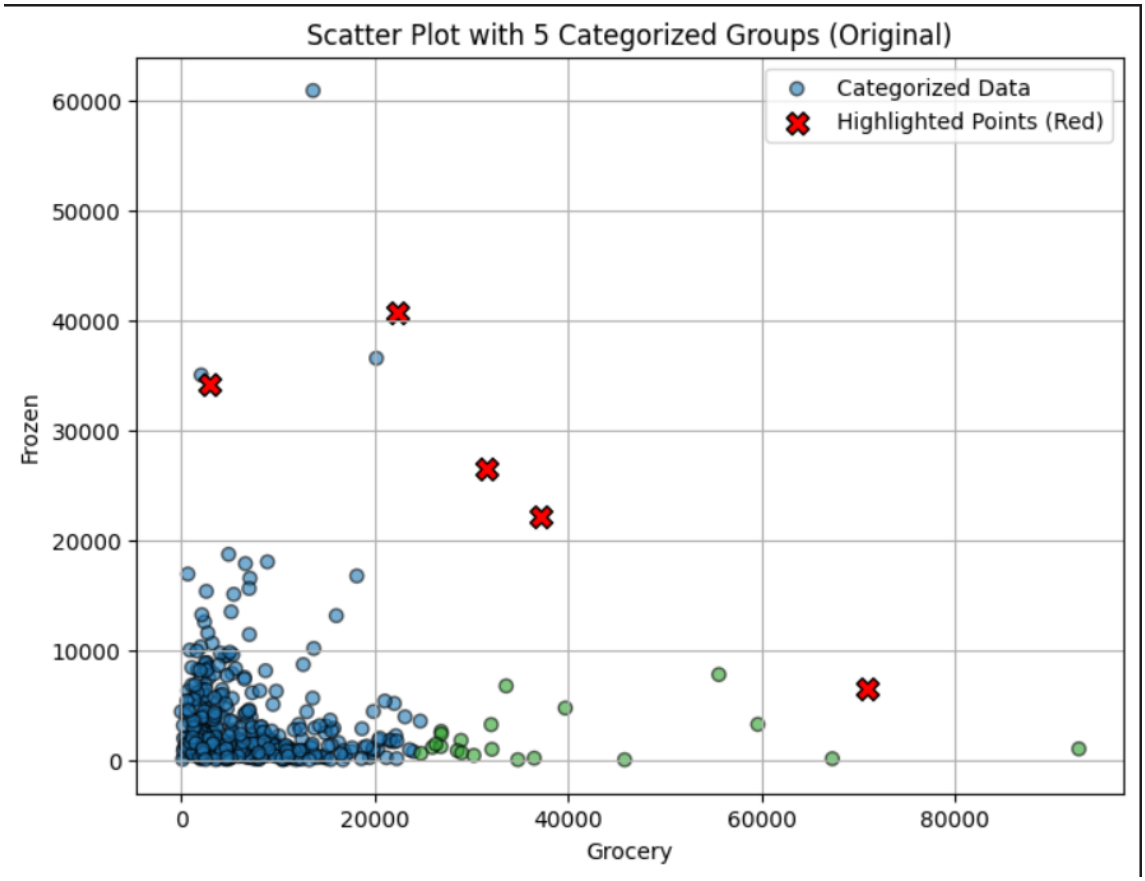


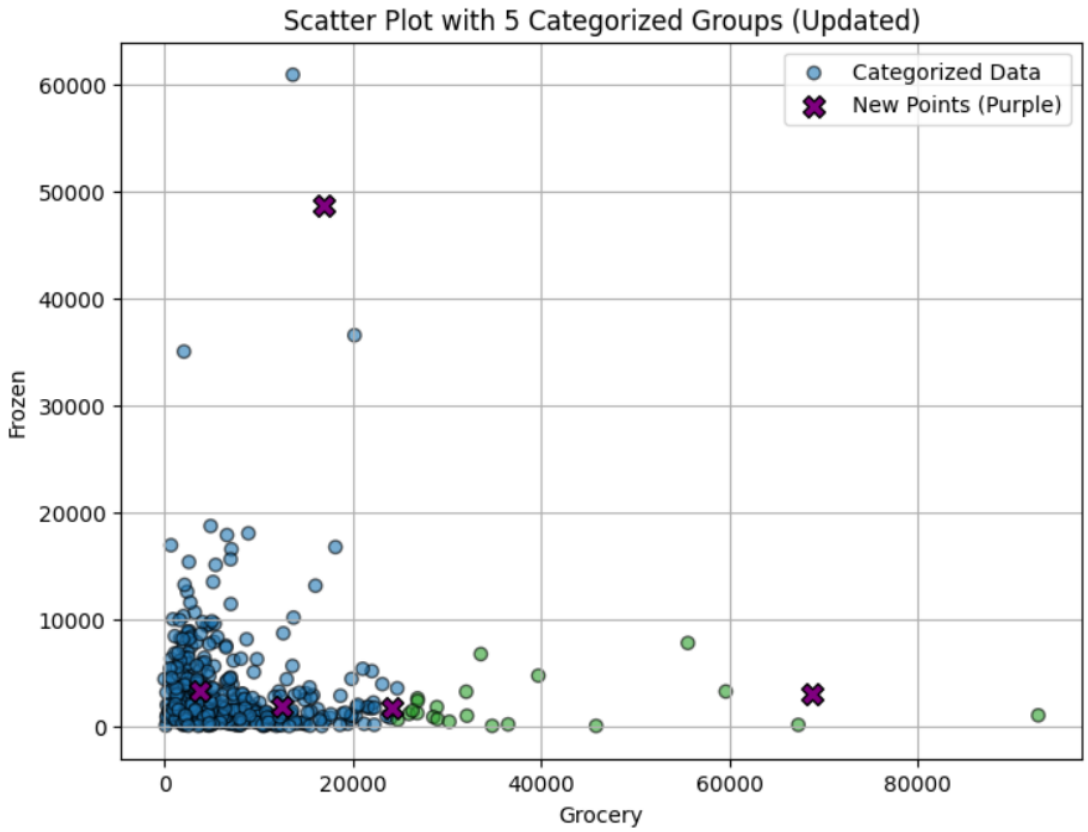

Scatter Plot with 2 Categorized Groups (Original)
Categorized Data
Highlighted Points (Red)
MILK
FRESH
0
10000
20000
30000
40000
50000
60000
70000
0
20000
40000
60000
80000
100000


Scatter Plot with 2 Categorized Groups (Updated)
Categorized Data
New Points (Purple)
MILK
FRESH
0
10000
20000
30000
40000
50000
60000
70000
0
20000
40000
60000
80000
100000

EXAMPLE : GROCERY and FROZEN


Scatter Plot
1e6
Data points
2.2
2.1
2.0
1.9
1.8
1.7
min_cost_arr-axis
1
2
3
4
5
6
7
K-axis


Scatter Plot with 5 Categorized Groups (Original)
Categorized Data
Highlighted Points (Red)
60000
50000
40000
30000
20000
10000
0
Frozen
0
20000
40000
60000
80000
Grocery


Scatter Plot with 5 Categorized Groups (Updated)
Categorized Data
New Points (Purple)
Frozen
Grocery
0
10000
20000
30000
40000
50000
60000
0
20000
40000
60000
80000