Data Mining Final Projesi

Grup Üyeleri:

Sara Nefise - 18120212009

Betül Albayrak - 18120212004

Verilere Erişim:

https://www.google.com/covid19/mobility/

Kodların Github Linki:

https://github.com/sarana200/Data_analysis

İçerikler:

Genel Bakış

COVID-19 döneminde Türkiye datalarını (insanların ziyaret ettikleri yerlerin datası) kullanarak descriptive bir problemin çözümünü **clustering** metodunu kullanarak gösterir.

Problem tanıtımı

Hayatın günlük alanları covid-19 nedeniyle nasıl etkilendiğini 2021 ile 2020 arasındaki değişimleri inceleyip ne gibi etkenler bu değişime sebep olduğunu göstermektir.

Projede kullanılan algoritmalar

- K Means metodu
- DbScan metodu

Aşamalar

1) Datayı Hazırlama

- Gereken kütüphaneleri import etme **[fig-1]**
- Datayı okuma **[fig-1]**
- Tüm null olan sütunları düşürme **[fig-2]**
- Sayısal ifadeleri içermeyen sütunları düşürme **[fig-2]**
- Kalan sütunları (retail sütunundan residential sütununa kadar) "Fillna" metodu kullanarak doldurma **[fig-2]**
- Alınan dataları normalize etme **[fig-3]**
- Datadaki outlier noktaları bulup data dan kaldırma **[fig-4]**

Kod kısmı:

```
import pandas as pd
from scipy.spatial import distance
import numpy as np
import random
import matplotlib.pyplot as plt
```

```
import sklearn
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans
from sklearn.preprocessing import normalize
from scipy.stats import zscore
from scipy.spatial.distance import cdist
from sklearn.datasets import load_digits
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans


#https://www.google.com/covid19/mobility/
url =
'https://drive.google.com/file/d/18gyHbx6rfogq3yQ-GR9COjcGgyYlCnBZ/view?usp=sharing'
url2020 = 'https://drive.google.com/uc?id=' + url.split('/')[-2]

df20 = pd.read_csv(url2020)
df20
```

[fig-1]

```
df20=df20.dropna(axis=1,how='all')
df20=df20.drop(df20.columns[[0,1,2, 3,4,5, 6]], axis = 1)
df20["residential_percent_change_from_baseline"].isna().sum()
df20["retail_and_recreation_percent_change_from_baseline"].fillna( method ='ffill', inplace =
True)
df20["grocery_and_pharmacy_percent_change_from_baseline"].fillna( method ='ffill', inplace
= True)
df20["parks_percent_change_from_baseline"].fillna( method ='ffill', inplace = True)
df20["transit_stations_percent_change_from_baseline"].fillna( method ='ffill', inplace = True)
df20["residential_percent_change_from_baseline"].fillna( method ='ffill', inplace = True)
df20["workplaces_percent_change_from_baseline"].fillna( method ='ffill', inplace = True)
df20
```

[fig-2]

```
normalizeData =df20.apply(zscore)
def outlier_removal(df, variable):
    upper_limit = df[variable].mean() + 1.5 * df[variable].std()
    lower_limit = df[variable].mean() - 1.5 * df[variable].std()
```

```
    return upper_limit, lower_limit
```

[fig-3]

```
upper_limit, lower_limit = outlier_removal(normalizeData,
"retail_and_recreation_percent_change_from_baseline")
print("Upper limit: ", upper_limit)
print("Lower Limit: ",lower_limit)

upper_limit1, lower_limit1 = outlier_removal(normalizeData,
"grocery_and_pharmacy_percent_change_from_baseline")
print("Upper limit: ", upper_limit1)
print("Lower Limit: ",lower_limit1)

upper_limit2, lower_limit2 = outlier_removal(normalizeData,
"parks_percent_change_from_baseline")
print("Upper limit: ", upper_limit2)
print("Lower Limit: ",lower_limit2)

upper_limit3, lower_limit3 = outlier_removal(normalizeData,
"transit_stations_percent_change_from_baseline")
print("Upper limit: ", upper_limit3)
print("Lower Limit: ",lower_limit3)

upper_limit4, lower_limit4 = outlier_removal(normalizeData,
"workplaces_percent_change_from_baseline")
print("Upper limit: ", upper_limit4)
print("Lower Limit: ",lower_limit4)

upper_limit5, lower_limit5 = outlier_removal(normalizeData,
"residential_percent_change_from_baseline")
print("Upper limit: ", upper_limit5)
print("Lower Limit: ",lower_limit5)
```

[fig-4]

```
normalizeData
=normalizeData[(normalizeData['retail_and_recreation_percent_change_from_baseline'] >
lower_limit) & (normalizeData['retail_and_recreation_percent_change_from_baseline'] <
upper_limit)]
```

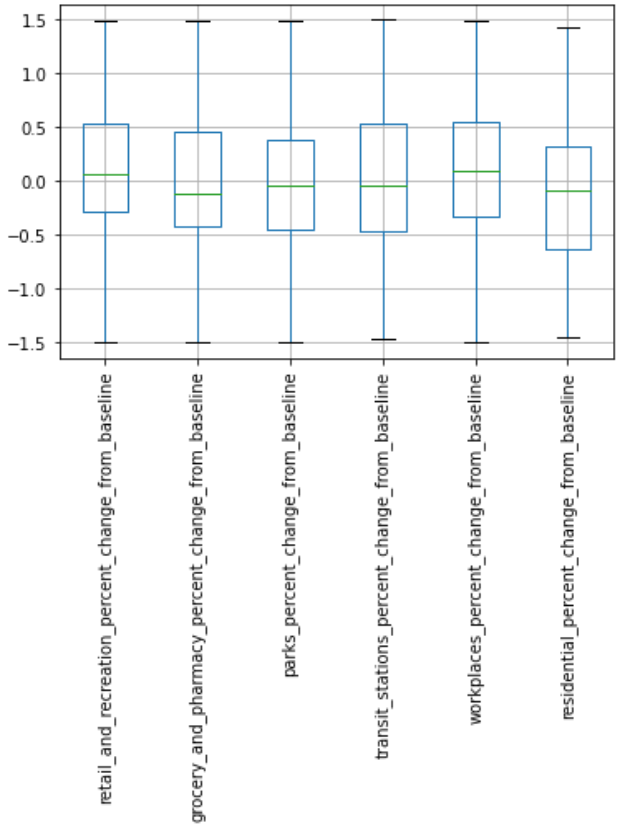
```
normalizeData
=normalizeData[(normalizeData['grocery_and_pharmacy_percent_change_from_baseline']>
lower_limit1) & (normalizeData['grocery_and_pharmacy_percent_change_from_baseline'] <
upper_limit1)]
normalizeData =normalizeData[(normalizeData['parks_percent_change_from_baseline']
>lower_limit2) & (normalizeData['parks_percent_change_from_baseline'] < upper_limit2)]
normalizeData
=normalizeData[(normalizeData['transit_stations_percent_change_from_baseline'] >
lower_limit3) & (normalizeData['transit_stations_percent_change_from_baseline'] <
upper_limit3)]
normalizeData =normalizeData[(normalizeData['workplaces_percent_change_from_baseline']
> lower_limit4) & (normalizeData['workplaces_percent_change_from_baseline'] <
upper_limit4)]
normalizeData =normalizeData[(normalizeData['residential_percent_change_from_baseline']
> lower_limit5) & (normalizeData['residential_percent_change_from_baseline'] <
upper_limit5)]
```

[fig-4]

```
normalizeData.boxplot(['retail_and_recreation_percent_change_from_baseline',
'grocery_and_pharmacy_percent_change_from_baseline','parks_percent_change_from_basel
ine' , 'transit_stations_percent_change_from_baseline',
'workplaces_percent_change_from_baseline', 'residential_percent_change_from_baseline'],
rot=90)
```

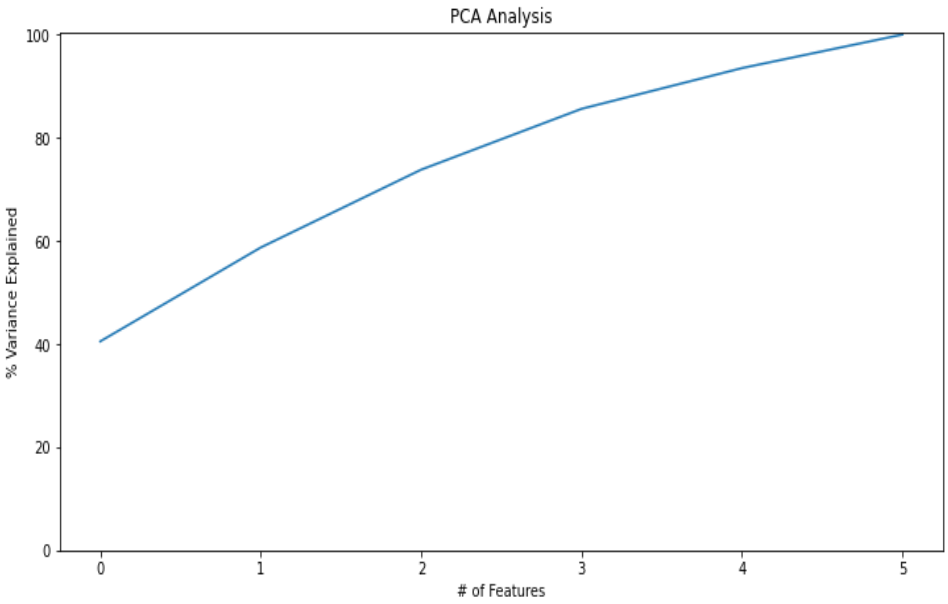
[fig-4]

Outlier'leri kaldırıldıktan sonra:

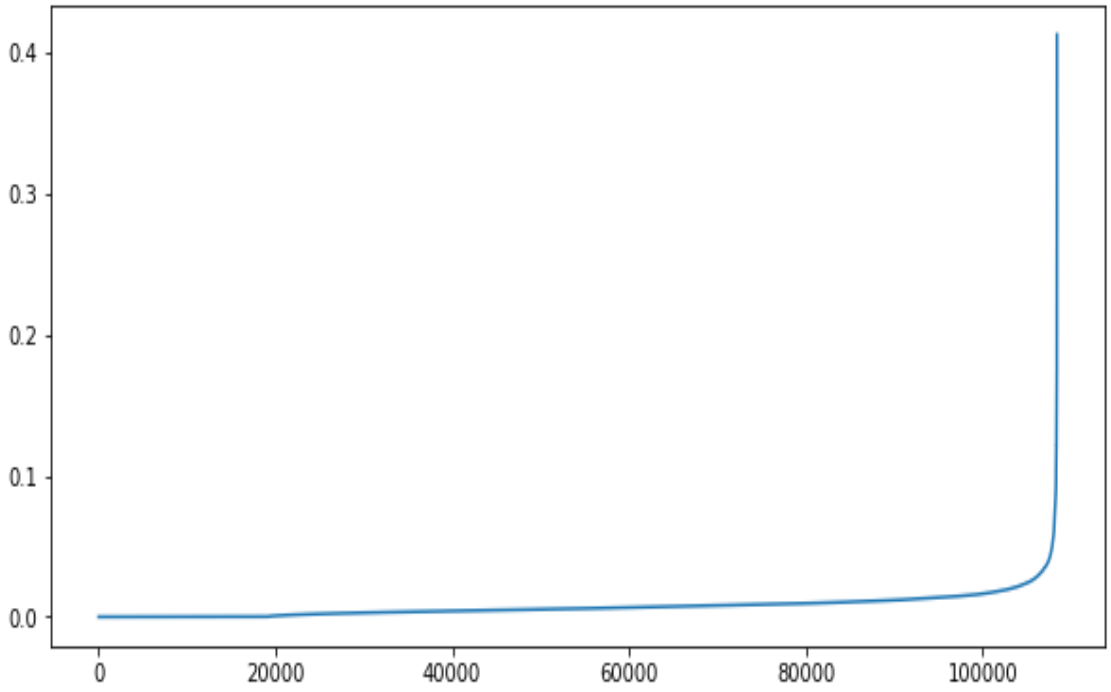


2) Dbscan algoritmasının aşamaları

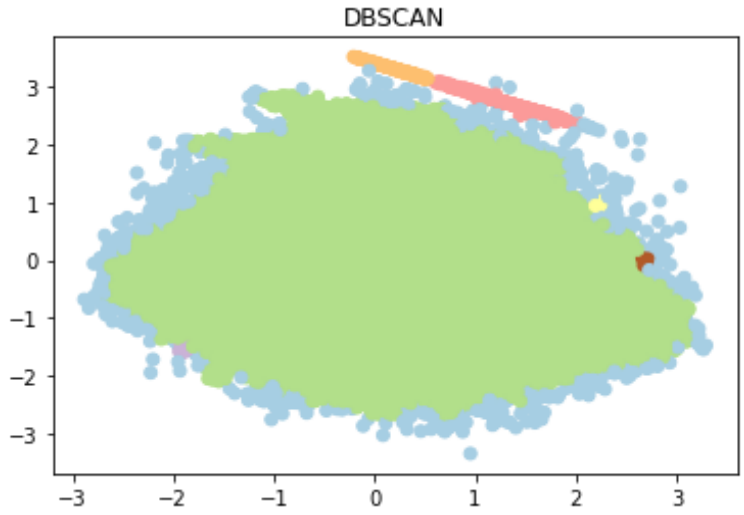
- PCA algoritmasını kullanarak dimension reduction yapıldı, PCA 2 olarak seçildi .



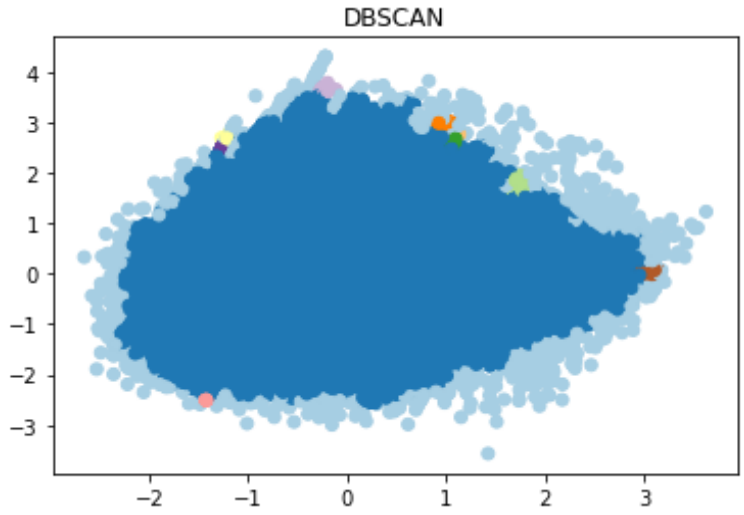
- eps değerini çıkartmak için KNN algoritması kullanıldı.



- Dbscan algoritması 2020 datasını kümelendirdi.



- Dbscan algoritması 2021 datasını kümelendirdi.



- DbScan Kodlar:

```
def dbscan(X, eps, min_samples):
    ss = StandardScaler()
    X = ss.fit_transform(X)
    db = DBSCAN(eps=eps, min_samples=min_samples)
    db.fit(X)
    y_pred = db.fit_predict(X)
```

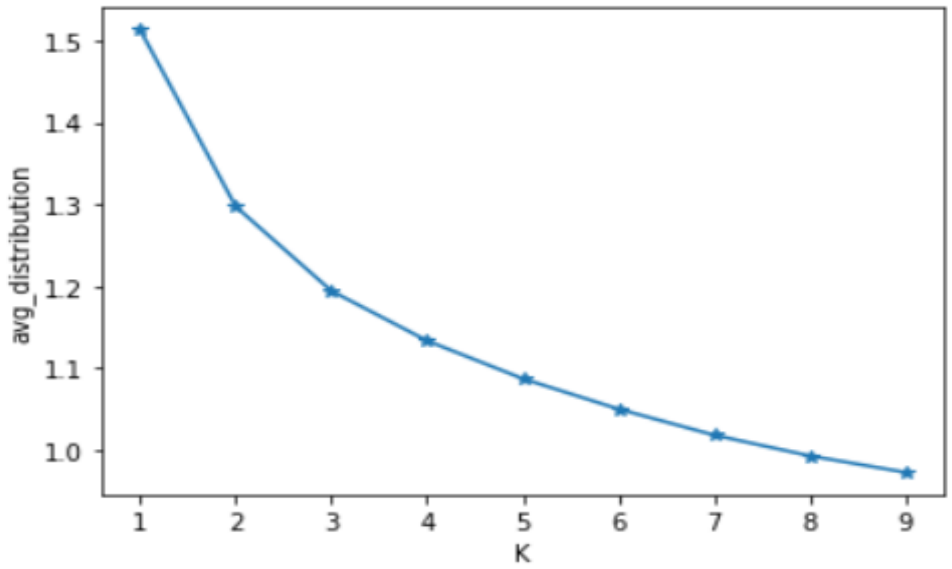
```
    plt.scatter(X[:,0], X[:,1],c=y_pred, cmap='Paired')
    plt.title("DBSCAN")
    labels = db.labels_# Number of clusters in labels, ignoring noise if
present.
    n_clusters_ = len(set(labels)) - (1 if -1 in labels else 0)
    n_noise_ = list(labels).count(-1)
    print('Estimated number of clusters: %d' % n_clusters_)
    print('Estimated number of noise points: %d' % n_noise_)
```

3) k means algoritmasının aşamaları

- Verilere en uygun cluster sayısı bulmak için Elbow yöntemi kullanma

```
from scipy.spatial.distance import cdist
clusters = range(1, 10)
meanDistortions = []

for k in clusters:
  model = KMeans(n_clusters = k)
  model.fit(sample_z)
  prediction = model.predict(sample_z)
  meanDistortions.append(sum(np.min(cdist(sample_z,
model.cluster_centers_, 'euclidean'), axis = 1)) /sample_z
    .shape[0])

plt.plot(clusters, meanDistortions, "*-")
plt.xlabel('K')
plt.ylabel('avg_distribution')
```

```
Text(0, 0.5, 'avg_distribution')
```

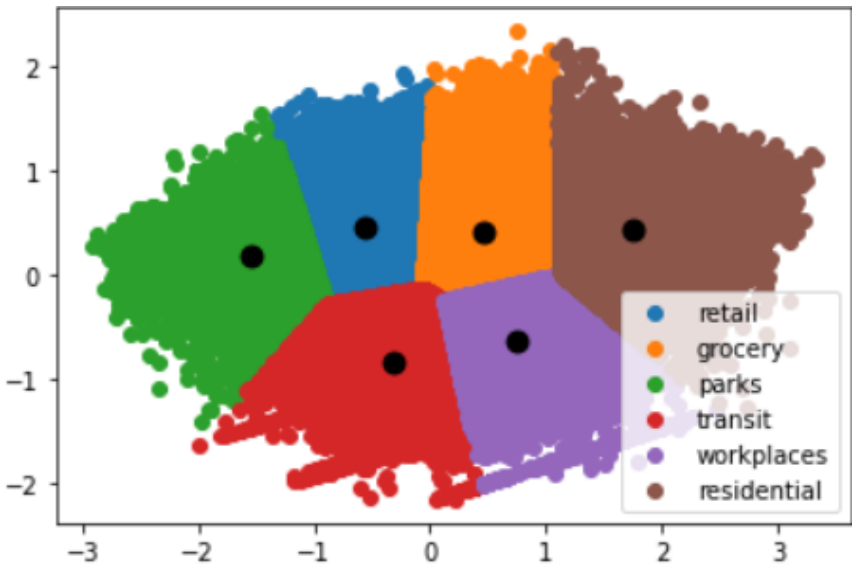


Not: burada görüldüğü gibi 2, 3, 4, 5, 6 ve 7 alınabilir tek bizim datamız 6 kategoriden oluştuğu için 6 kümeye bölmeyi tercih ettik.

- K means algoritması kullanılıp verileri gösterme

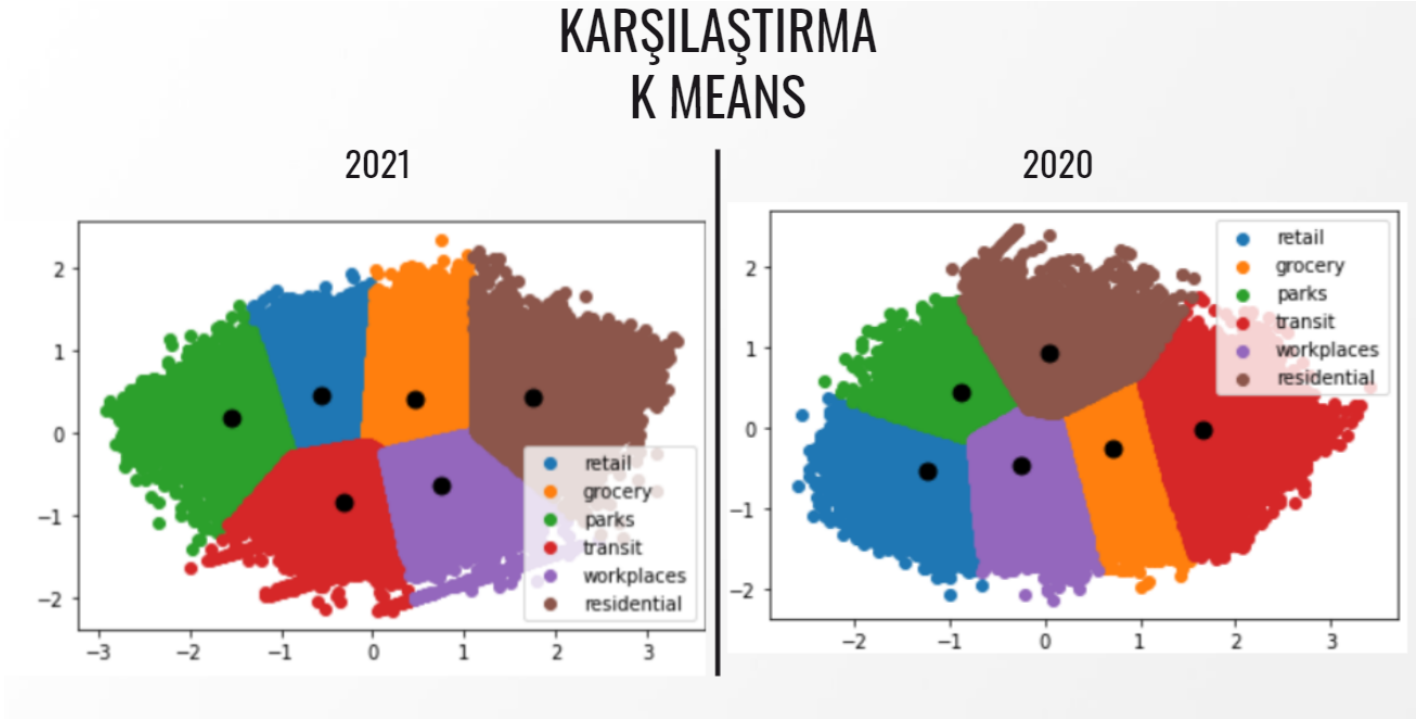
```
from sklearn.datasets import load_digits
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
import numpy as np

#Load Data
data = sample_z
pca = PCA(2)

#Transform the data
df = pca.fit_transform(data)

#Import KMeans module
from sklearn.cluster import KMeans

#Initialize the class object
kmeans = KMeans(n_clusters= 6)

#predict the labels of clusters.
label = kmeans.fit_predict(df)
```

```
##
sample_z['kmean'] = kmeans.labels_
cluster_count = sample_z['kmean'].value_counts()
cluster_count_sum = cluster_count.sum()
custer_count_sorted = cluster_count.sort_index()

#Getting unique labels
u_labels = np.unique(label)
centroids = kmeans.cluster_centers_
#plotting the results:
columns = ["retail", "grocery", "parks", "transit", "workplaces",
"residential"]
for i in u_labels:
    plt.scatter(df[label == i , 0] , df[label == i , 1] , label =columns[i])
    print ('cluster name {}, percentage of cluster in whole data {:.2f}%
'.format(columns[i],(custer_count_sorted[i]*100)/cluster_count_sum))
plt.scatter(centroids[:,0] , centroids[:,1] , s = 80, color = 'k')
plt.legend()
plt.show()
```



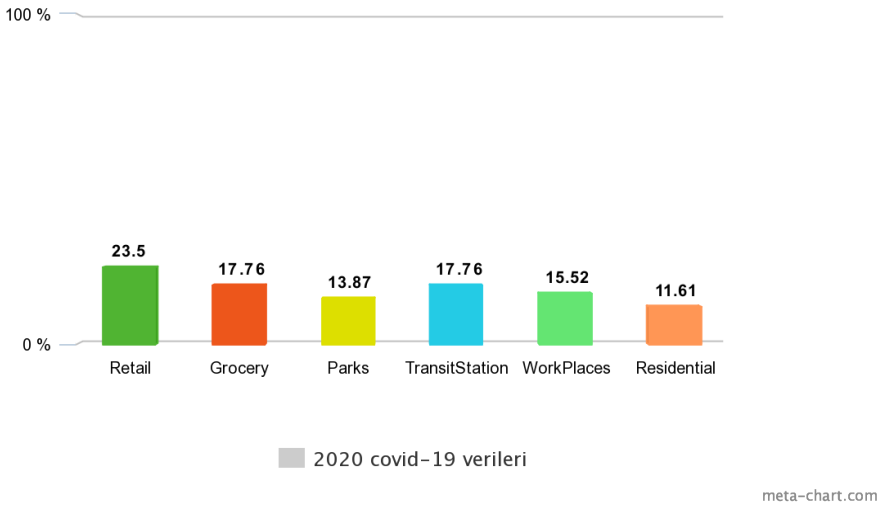
Analiz Kısmı:

K means algoritmasının çıktıları doğrultusuna 2020 - 2021 yılların analizi gerçekleştirdik. Bu analiz Türkiye'de Covid-19 eğlence yerlerine gitme , eczanelere ve marketlere gitme, parklara gitme, ulaşım araçları kullanma, işe gitme, evde kalma 2020-2021'de oranları kıyaslayarak nasıl değişiklik gösterdiğini yorumlamaktadır.

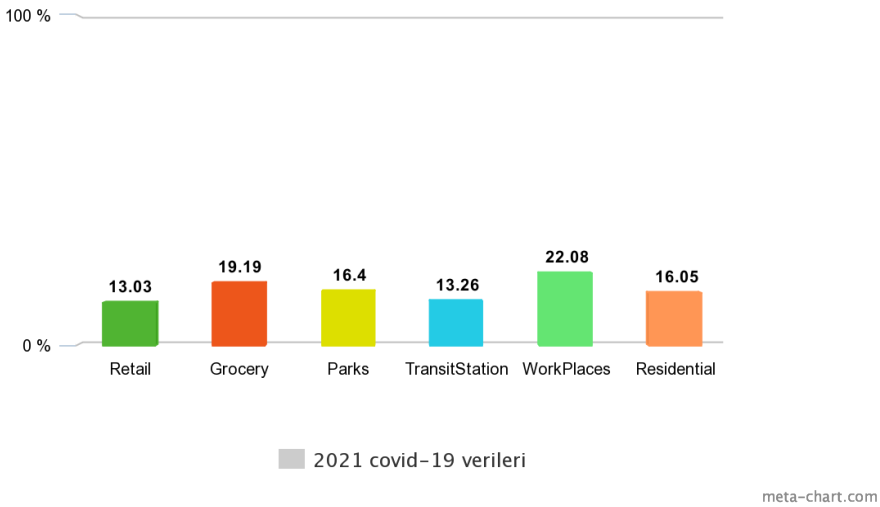


Not: değişiklikleri genel olarak bu şekilde çıkmaktadır. Bazı alanların oranı arttı ve bazılarınki de azaldı. bunları sayısal değerleriyle karşılaştırarak tane tane ele alalım.

2020 covid-19 verilerine clustering metodu (k means) kullanarak 6 kategorilere kümeleyerek sayısal değerleri

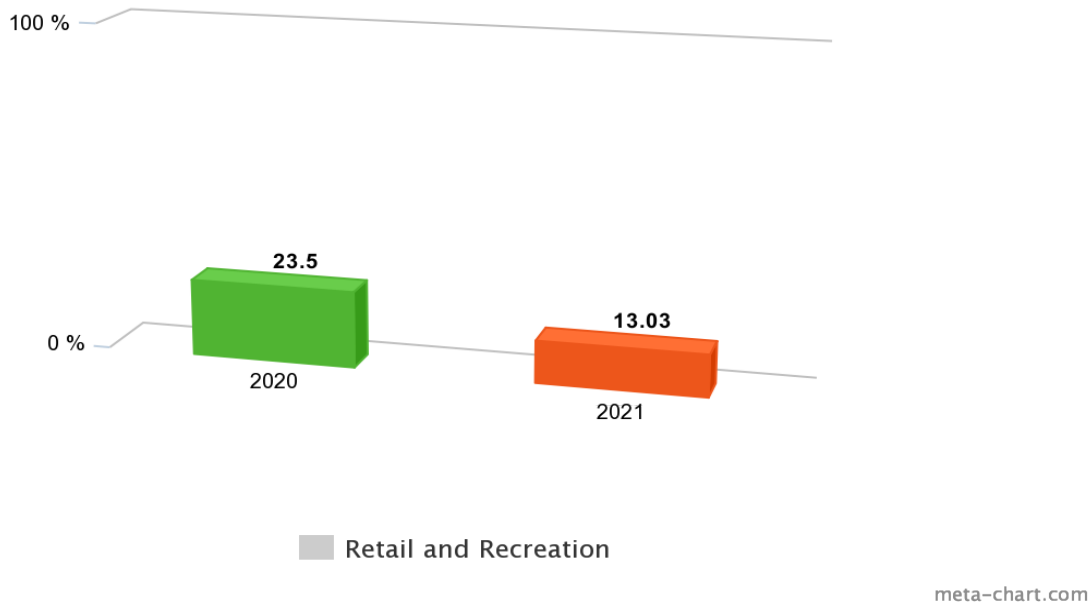


2021 covid-19 verilerine clustering metodu (k means) kullanarak 6 kategorilere kümeleyerek sayısal değerleri



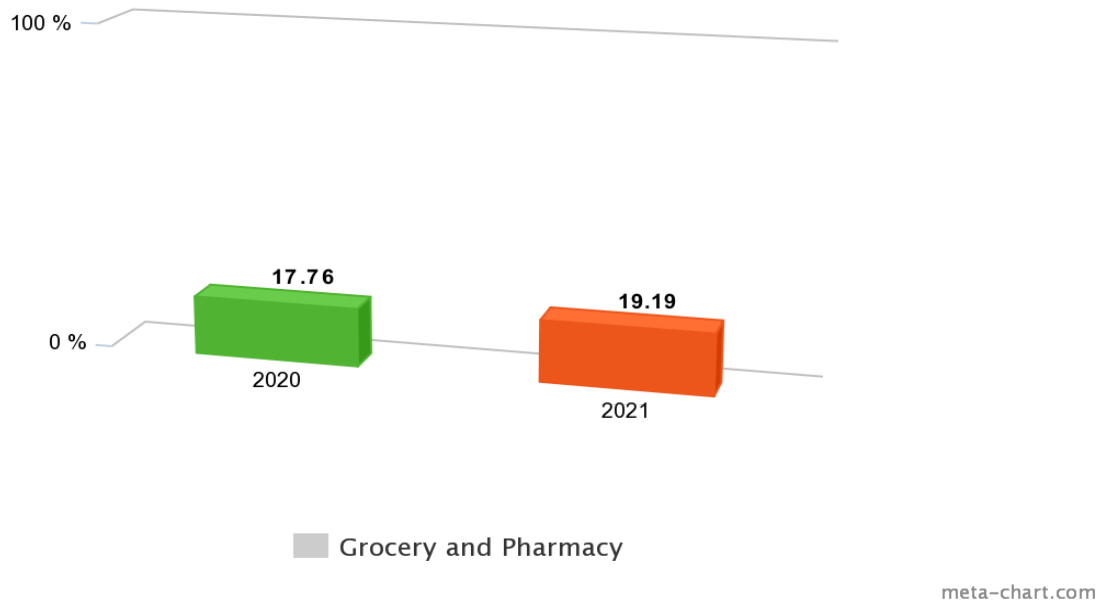
Retail and Recreation: Grafikte gözüktüğü gibi 2020 yılında insanların avm, sinema, kafeye gitme oranı 23.5 iken, 2021 yılından daha az olduğu gözlemlenmektedir. ve bundan bunları anlayabiliyoruz:

- Türkiye'nin ekonomik durumu 2021 yılında kötüleştiğini
- insanların ticari (alış-veriş) işlemleri azaltmış olduğunu
- hastalık yüzünden eğlence yerlerine fazla gitmediklerini gözlemleyebiliyoruz.



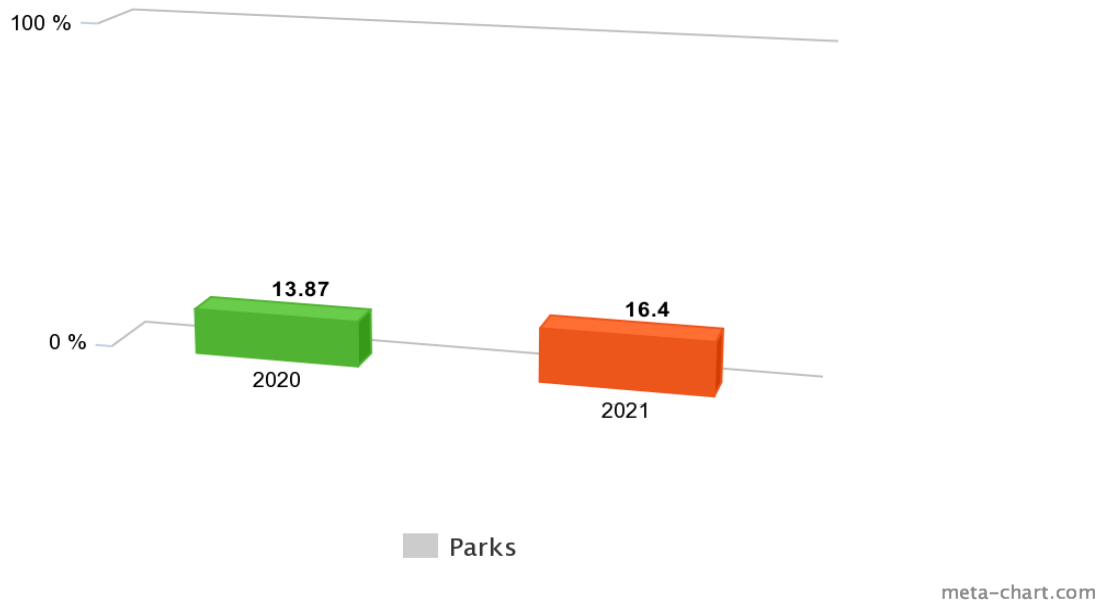
Grocery and Pharmacy: Grafikte gözüktüğü gibi 2021 yılında insanların eczaneye ve marketlere gitme oranları 2020 yılından birbirine yakın olduğu gözlemlenmektedir. bu oranlarda bunu görebiliyoruz:

- insanların eczanelerden ilaç alma oranı büyük bir değişim görmediğini
- hastalık durumunda fazla fark olmadığını
- yaşam kalitesinde de 2021 ile 2020 birbiriyle yakındır.



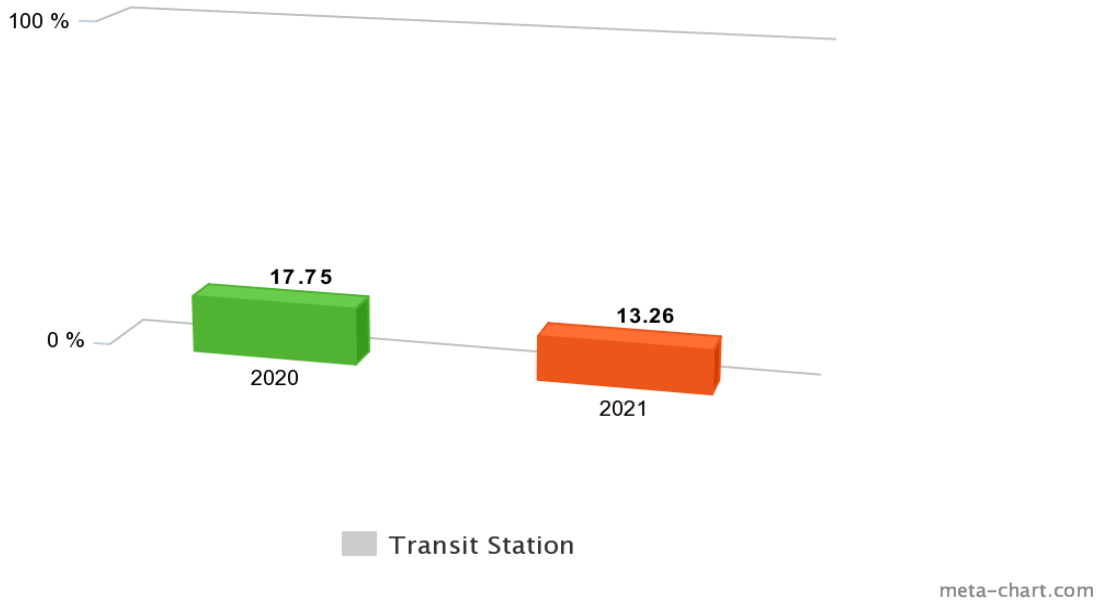
Parks: Grafikte gözüktüğü gibi 2021 yılında insanların parklara ve bahçelere gitme oranları 2020 yılından daha yüksek olduğu gözlemlenmektedir. bundan bu anlamları çıkarabiliyoruz:

- insanların sosyal hayatını 2021 daha iyi olduğu
- public alanları kullanmaya döndüğünü
- insanların psikolojilerini bir tık iyileştiğini gözlemleyebiliyoruz.



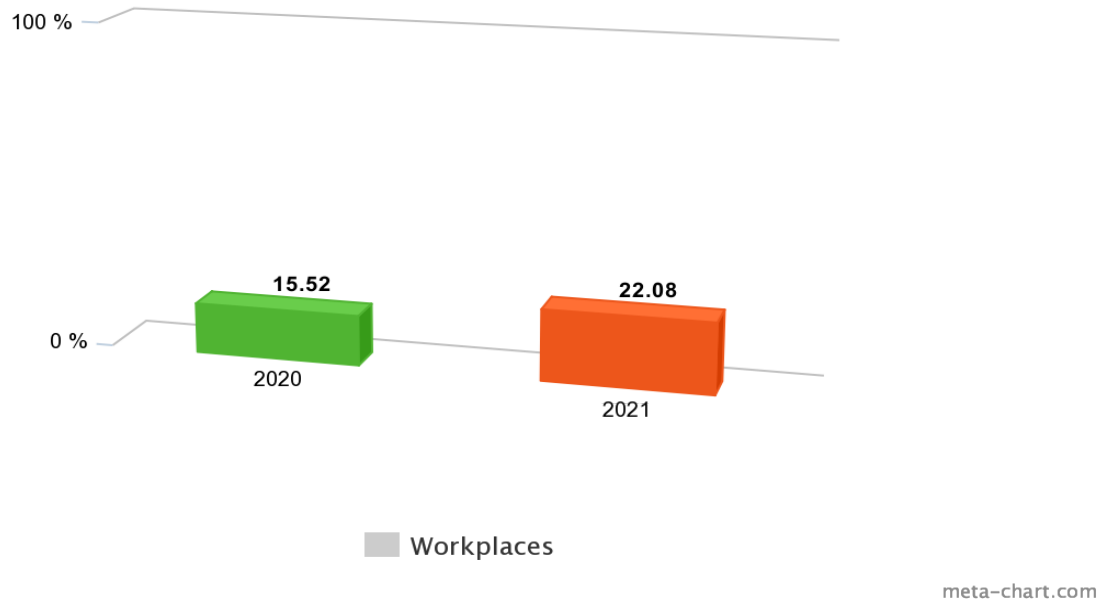
Transit Station: Grafikte gözüktüğü gibi 2020 yılında insanların otobüs, metro kullanma oranları 2021 yılına göre daha yüksek olduğu gözlemlenmektedir. ve bunun analizi yaparsak şunlara varabiliyoruz:

- ülkenin 2020 yılında insanların ulaşım araçları kullanmaya daha ihtiyaç duyduğunu
- aynı zamanda covid-19 tedbirlerini alarak şehirler arası (iç hatları) kullanılabilir hale geldiğini gözlemleyebiliriz.



WorkPlace: Grafikte gözüktüğü gibi 2021 yılında insanların işe gitme oranları 2020 yılına göre daha yüksek olduğu gözlemlenmektedir. bundan da bu noktaları anlayabiliriz.

- uzaktan çalışma alanları, bazıları yüz yüze çalışma şekline döndüğünü
- insanların normal hayatına dönmeye başladıklarını
- ülke, covid-19'unun aşısı insanlara vererek insanların normal hayatına dönmeleri sağlamaya çalıştığını görebiliyoruz.



Residential: Grafikte gözüktüğü gibi 2021 yılında insanların evde kalma oranları 2020 yılına göre daha yüksek olduğu gözlemlenmektedir. bundan bunu gözlemleyebiliriz.

- 2021 yılında insanların evde kalmaları alışmaya başladığını
- hayatın çoğu alanları online olmaya başladığını görebiliyoruz