Veri Bilimi Ödev Raporu

Bu proje, mobil cihazların fiyat aralığını tahmin etmeye yönelik bir çalışma olup, veri analizi, özellik mühendisliği ve makine öğrenmesi tekniklerini bir arada kullanarak kapsamlı bir süreç izlenmiştir. Çalışmada, train_odev.csv adlı veri seti temel alınarak çeşitli analizler ve modelleme adımları gerçekleştirilmiştir.

1. Veri Analizleri

Proje başlangıcında, veri setinin genel yapısı detaylı bir şekilde incelenmiştir. Bu süreçte sütunların benzersiz değer sayıları, eksik veri durumları ve temel istatistiksel özetler analiz edilmiştir.

Veri Setinin Tanımlanması: Veri setindeki kayıt sayısı, ortalama, standart sapma ve temel istatistiksel özetler describe() fonksiyonu kullanılarak elde edilmiştir *(Şekil 1).* İlk beş kayıt ise head() fonksiyonu yardımıyla gözlemlenmiştir *(Şekil 2).*

```
print("Veri Seti Özeti:")
print(df.describe())
```

```
Veri Seti Özeti:
           battery         blue        speed     dual_sim      fcamera  \
count  1200.000000  1200.000000  1200.000000  1200.000000  1200.000000
mean   1239.906667     0.507500     1.492500     0.508333     4.365833
std     442.722035     0.500152     0.821212     0.500139     4.355204
min     501.000000     0.000000     0.500000     0.000000     0.000000
25%     845.500000     0.000000     0.600000     0.000000     1.000000
50%    1231.500000     1.000000     1.400000     1.000000     3.000000
75%    1619.250000     1.000000     2.200000     1.000000     7.000000
max    1998.000000     1.000000     3.000000     1.000000    19.000000

                g4       memory        pdepth      pweight        cores  \
count  1200.000000  1200.000000  1200.000000  1200.000000  1200.000000
mean      0.527500    31.765000     0.501167   140.223333     4.489167
std       0.499451    17.920307     0.285861    35.596878     2.277617
min       0.000000     2.000000     0.100000    80.000000     1.000000
25%       0.000000    16.000000     0.200000   109.000000     3.000000
50%       1.000000    31.000000     0.500000   141.000000     4.000000
75%       1.000000    47.000000     0.800000   170.000000     6.000000
max       1.000000    64.000000     1.000000   200.000000     8.000000

            pcamera    px_height     px_width      sheight      swidth  \
count  1200.000000  1200.000000  1200.000000  1200.000000  1200.00000
mean     10.031667   666.865000  1265.271667    12.380000     5.79000
std       6.010194   451.662648   425.400393     4.223798     4.40932
min       0.000000     0.000000   500.000000     5.000000     0.00000
25%       5.000000   295.000000   896.000000     9.000000     2.00000
50%      10.000000   594.000000  1260.000000    13.000000     5.00000
75%      15.000000   977.250000  1632.250000    16.000000     9.00000
max      20.000000  1949.000000  1997.000000    19.000000    18.00000

         talk_time           g3  touch_screen         wifi  price_range
count  1200.000000  1200.000000   1200.000000  1200.000000    1200.0000
mean     11.325833     0.758333      0.505833     0.507500       1.5000
std       5.516342     0.428272      0.500174     0.500152       1.1185
min       2.000000     0.000000      0.000000     0.000000       0.0000
25%       7.000000     1.000000      0.000000     0.000000       0.7500
50%      12.000000     1.000000      1.000000     1.000000       1.5000
75%      16.000000     1.000000      1.000000     1.000000       2.2500
max      20.000000     1.000000      1.000000     1.000000       3.0000
```

Şekil-1 veri setini "describe()" ile incelenmesi

```
print("\nİlk 5 Satır:")
print(df.head())
```

```
İlk 5 Satır:
   battery  blue  speed  dual_sim  fcamera  g4  memory  pdepth  pweight  \
0      842     0    2.2         0        1   0       7     0.6      188
1     1021     1    0.5         1        0   1      53     0.7      136
2      563     1    0.5         1        2   1      41     0.9      145
3      615     1    2.5         0        0   0      10     0.8      131
4     1821     1    1.2         0       13   1      44     0.6      141

   cores  pcamera  px_height  px_width  sheight  swidth  talk_time  g3  \
0      2        2         20       756        9       7         19   0
1      3        6        905      1988       17       3          7   1
2      5        6       1263      1716       11       2          9   1
3      6        9       1216      1786       16       8         11   1
4      2       14       1208      1212        8       2         15   1

   touch_screen  wifi  price_range
0             0     1            1
1             1     0            2
2             1     0            2
3             0     0            2
4             1     0            1
```

Şekil-2 veri setini "head()" ile ilk 5 değerinin incelenmesi

Veri Özelliklerinin İncelenmesi: Veri setinin toplam özellik sayısı ve örnek sayısı info() fonksiyonu ile analiz edilmiştir. Ayrıca, niteliklerin veri tipleri ve eksik değer durumları incelenmiştir *(Şekil 3 ve Şekil 4).*

```
print("\nVeri Tipleri:")
print(df.info())
```

```
Veri Tipleri:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1200 entries, 0 to 1199
Data columns (total 20 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   battery       1200 non-null   int64
 1   blue          1200 non-null   int64
 2   speed         1200 non-null   float64
 3   dual_sim      1200 non-null   int64
 4   fcamera       1200 non-null   int64
 5   g4            1200 non-null   int64
 6   memory        1200 non-null   int64
 7   pdepth        1200 non-null   float64
 8   pweight       1200 non-null   int64
 9   cores         1200 non-null   int64
 10  pcamera       1200 non-null   int64
 11  px_height     1200 non-null   int64
 12  px_width      1200 non-null   int64
 13  sheight       1200 non-null   int64
 14  swidth        1200 non-null   int64
 15  talk_time     1200 non-null   int64
 16  g3            1200 non-null   int64
 17  touch_screen  1200 non-null   int64
 18  wifi          1200 non-null   int64
 19  price_range   1200 non-null   int64
dtypes: float64(2), int64(18)
memory usage: 187.6 KB
None
```

Şekil-3 veri setini "info()" ile nitelik tiplerinin incelenemesi

```
print("\nEksik Değerlerin Sayısı:")
print(df.isnull().sum())
```

```
Eksik Değerlerin Sayısı:
battery         0
blue            0
speed           0
dual_sim        0
fcamera         0
g4              0
memory          0
pdepth          0
pweight         0
cores           0
pcamera         0
px_height       0
px_width        0
sheight         0
swidth          0
talk_time       0
g3              0
touch_screen    0
wifi            0
price_range     0
dtype: int64
```

Şekil-4 veri setini "isnull" ile eksik değerlerin incelenemesi

Eşsiz Değerlerin Analizi: Her bir özellik için eşsiz değerlerin sayısı "nunique" fonksiyonu ile değerlendirilmiştir *(Şekil 5).*

```
unique = df.nunique()
unique.to_frame().T
```

	battery	blue	speed	dual_sim	fcamera	g4	memory	pdepth	pweight	cores	pcamera	px_height	px_width	sheight	swidth	talk_time	g3	touch_screen	price_range
0	833	2	26	2	20	2	63	10	121	8	21	824	836	15	19	19	2	2	4

Şekil-5 "nunique" ile eşsiz değerlerin incelenmesi

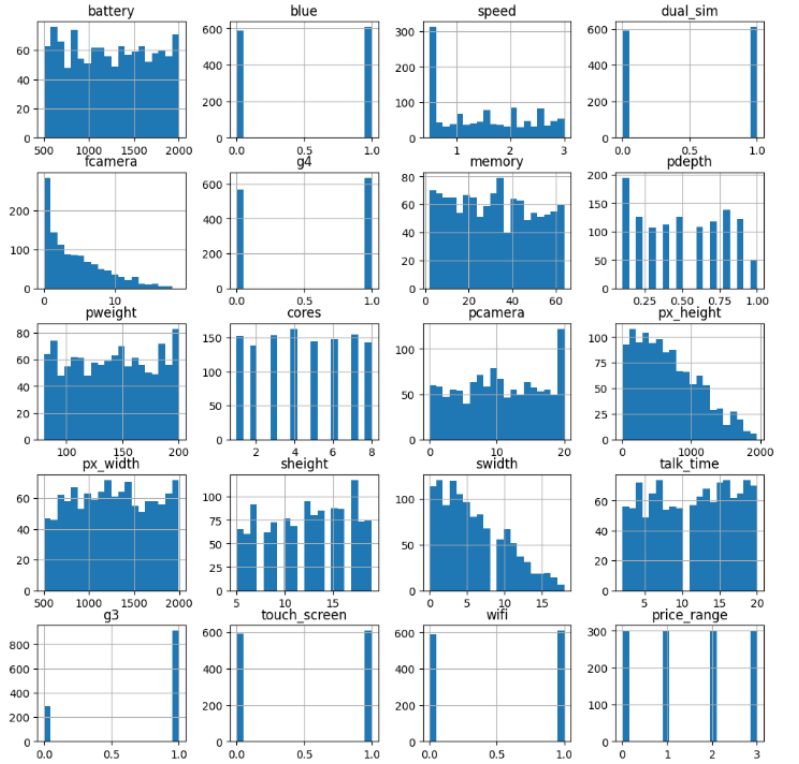
2.Verinin Görselleştirme

Veri analizi sürecinde, verilerin daha anlaşılır hale getirilmesi amacıyla çeşitli görselleştirme teknikleri uygulanmıştır.

Histogram Analizi: Her bir özellik için histogram grafikleri oluşturularak veri dağılımları gözlemlenmiştir *(Şekil 6).*

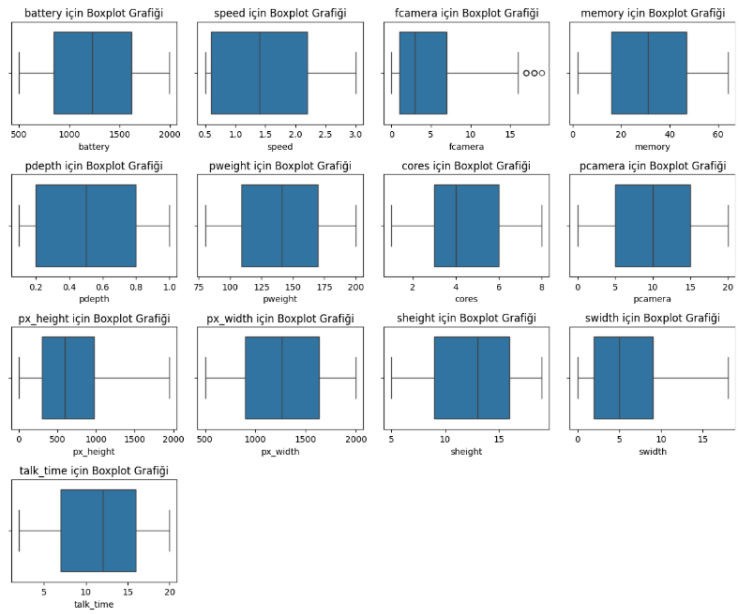
Box Plot Çizimleri: Sürekli değişkenler için box plot grafikleri çizdirilmiş ve bu sayede aykırı değerler ve dağılımlar daha net anlaşılmıştır *(Şekil 7).*

```
df.hist(bins=20,figsize=(12,12))
plt.suptitle("Histogramlar",size=18)
plt.show()
```



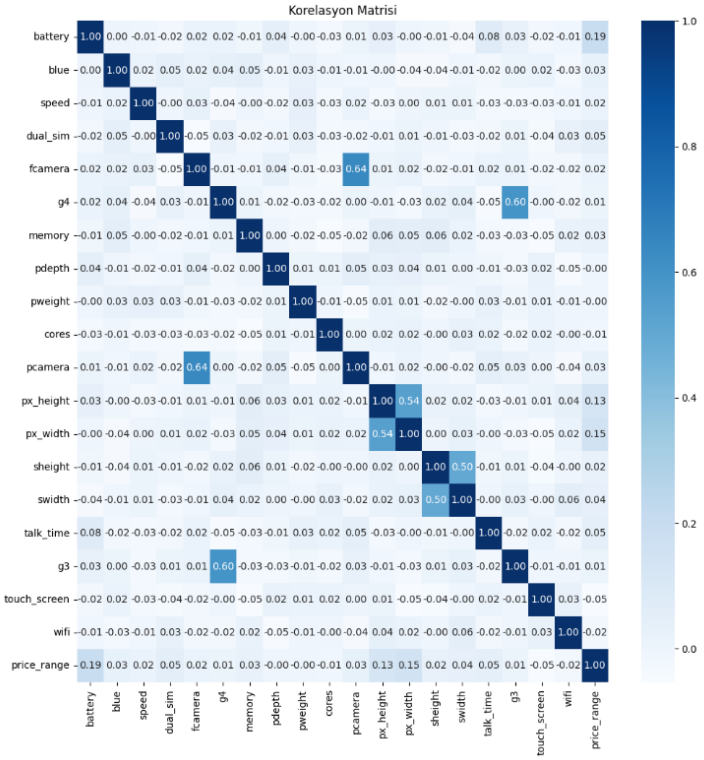
Şekil-6 veri setini histogram grafikleri

```
plt.figure(figsize=(12, 12))
continuous_features = ['battery', 'speed', 'fcamera', 'memory', 'pdepth', 'pweight',
                       'cores', 'pcamera', 'px_height', 'px_width', 'sheight', 'swidth', 'talk_time']


for i, column in enumerate(continuous_features, 1):
    plt.subplot(5, 4, i)
    sns.boxplot(data=df, x=column)
    plt.title(f"{column} için Boxplot Grafiği")

plt.tight_layout()
plt.show()
```



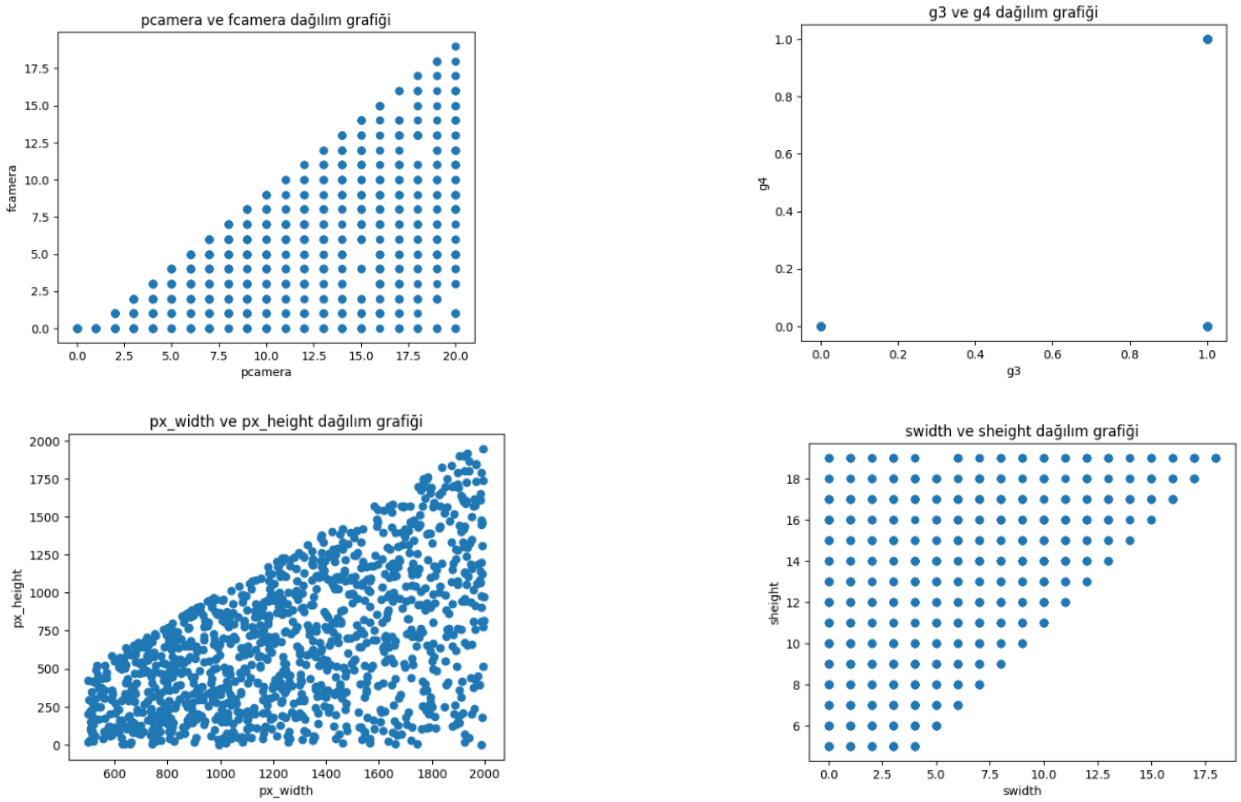
Şekil-7 sürekli değerler için box plots grafikleri

Korelasyon Matrisi: Özellikler arasındaki ilişkileri analiz etmek için bir korelasyon matrisi oluşturulmuştur *(Şekil 8).*



Şekil-8 korelasyon matrisi

Scatter Plot Görselleştirmeleri: Korelasyon matrisi sonuçlarına göre, en yüksek ilişkiye sahip değişken çiftleri için scatter plot grafikleri hazırlanmıştır *(Şekil 9).*



Şekil-9 scatter plots grafikleri

3. Veri Ön İşleme

Veri ön işleme aşamasında, veri seti eksik değerler incelenmiş ve herhangi bir eksikliğin olmadığı tespit edilmiştir. Bu nedenle, eksik veri doldurma veya silme işlemlerine gerek duyulmamıştır.

Korelasyon analizi: Veri setindeki özellikler arasında genel olarak zayıf ilişkiler olduğunu göstermiştir. Bu durum, bazı özelliklerin hedef değişkenle yeterince güçlü bir bağ kuramadığını ve model performansını olumsuz etkileyebileceğini düşündürmüştür. Bu nedenle, özellik mühendisliği ile "battery_feature_2" adında yeni bir değişken eklenmiş ve bu değişkenin korelasyon ilişkilerini belirgin şekilde artırdığı gözlemlenmiştir *(Şekil 10).*

```
if "battery" in df.columns and "price_range" in df.columns:
    price_range = df["price_range"]
    df["battery_feature_2"]=((df["px_height"]*df["px_width"])/df["battery"]) *price_range
```

Şekil-10 "battery_feature_2" özelliğinin eklenmesi

Aykırı Değer Analizi: Veri setindeki her bir özellik için aykırı değerler detaylı bir şekilde incelenmiştir. Özellikle "battery_feature_2" ve "fcamera" özelliklerinde tespit edilen aykırı değerler uygun yöntemlerle baskılanmıştır *(Şekil 11).*

```
for column in df.columns:
    if df[column].dtype in ['int64', 'float64']:
        Q1 = df[column].quantile(0.25)
        Q3 = df[column].quantile(0.75)
        IQR = Q3 - Q1
    #
        outliers = ((df[column] < (Q1 - 1.5 * IQR)) | (df[column] > (Q3 + 1.5 * IQR)))

        num_outliers = outliers.sum()

        print(f"Sütun: {column}, Aykırı Değer Sayısı: {num_outliers}")
```

```
Sütun: battery, Aykırı Değer Sayısı: 0
Sütun: blue, Aykırı Değer Sayısı: 0
Sütun: speed, Aykırı Değer Sayısı: 0
Sütun: dual_sim, Aykırı Değer Sayısı: 0
Sütun: fcamera, Aykırı Değer Sayısı: 13
Sütun: g4, Aykırı Değer Sayısı: 0
Sütun: memory, Aykırı Değer Sayısı: 0
Sütun: pdepth, Aykırı Değer Sayısı: 0
Sütun: pweight, Aykırı Değer Sayısı: 0
Sütun: cores, Aykırı Değer Sayısı: 0
Sütun: pcamera, Aykırı Değer Sayısı: 0
Sütun: px_height, Aykırı Değer Sayısı: 0
Sütun: px_width, Aykırı Değer Sayısı: 0
Sütun: sheight, Aykırı Değer Sayısı: 0
Sütun: swidth, Aykırı Değer Sayısı: 0
Sütun: talk_time, Aykırı Değer Sayısı: 0
Sütun: g3, Aykırı Değer Sayısı: 290
Sütun: touch_screen, Aykırı Değer Sayısı: 0
Sütun: wifi, Aykırı Değer Sayısı: 0
Sütun: price_range, Aykırı Değer Sayısı: 0
Sütun: battery_feature_2, Aykırı Değer Sayısı: 86
```

```
fcamera_lower_bound = 0
fcamera_upper_bound = 15
battery_lower_bound = 0
battery_upper_bound = 4000

columns = ['fcamera', 'battery_feature_2']
# columns = ['fcamera']


for col in columns:
    if col == 'fcamera':
        df[col] = df[col].apply(lambda x: fcamera_lower_bound if x < fcamera_lower_bound else (fcamera_upper_bound if x > fcamera_upper_bound else x))
    elif col == 'battery_feature_2':
        df[col] = df[col].apply(lambda x: battery_lower_bound if x < battery_lower_bound else (battery_upper_bound if x > battery_upper_bound else x))
```

Şekil-11 Aykırı değerlerin tespiti ve baskılanması

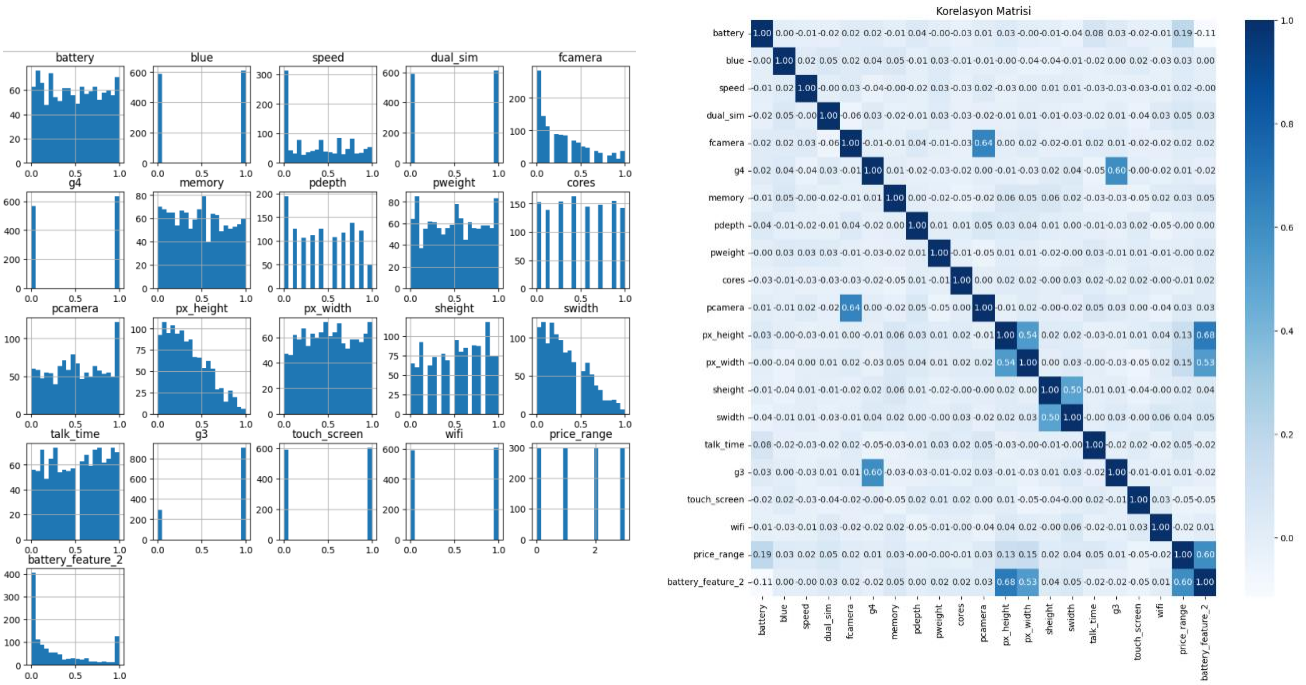
Normalizasyon: Veri setindeki sürekli değişkenlere, değerlerin daha iyi ölçeklenmesi ve model performansının artırılması için min-maks normalizasyonu uygulanmıştır *(Şekil 12).*

```
continuous_features = ['battery', 'speed', 'fcamera', 'memory', 'pdepth', 'pweight',
                       'cores', 'pcamera', 'px_height', 'px_width', 'sheight', 'swidth', 'talk_time',"battery_feature_2"]

#

scaler = MinMaxScaler()
df[continuous_features] = scaler.fit_transform(df[continuous_features])

df.head()
```

	battery	blue	speed	dual_sim	fcamera	g4	memory	pdepth	pweight	cores	...	px_height	px_width	sheight	swidth	talk_time
0	0.227789	0	0.68	0	0.066667	0	0.080645	0.555556	0.900000	0.142857	...	0.010262	0.171009	0.285714	0.388889	0.944444
1	0.347361	1	0.00	1	0.000000	1	0.822581	0.666667	0.466667	0.285714	...	0.464341	0.993988	0.857143	0.166667	0.277778
2	0.041416	1	0.00	1	0.133333	1	0.629032	0.888889	0.541667	0.571429	...	0.648025	0.812291	0.428571	0.111111	0.388889
3	0.076152	1	0.80	0	0.000000	0	0.129032	0.777778	0.425000	0.714286	...	0.623910	0.859051	0.785714	0.444444	0.500000
4	0.881764	1	0.28	0	0.866667	1	0.677419	0.555556	0.508333	0.142857	...	0.619805	0.475618	0.214286	0.111111	0.722222

Şekil-12 sürekli değerler için normalizasyon yapılması

Ön işleme adımları tamamlandıktan sonra veri seti histogram tabloları, korelasyon ilişkileri ve diğer görselleştirmeler kullanılarak yeniden analiz edilmiştir *(Şekil 13).*



Şekil-13 ön işlenmiş veri setinin histogram tablosu ve korelasyon matrisi

4. Model Seçimi ve Eğitimi

Bu projede gerek deneyimlemelerden gerekse esnek ve değiştirilebilir yapısından dolayı yapay sinir ağı olan "CNN" modeli tercih edilmiştir. Model tasarımında ilk olarak veri setinde hedef değişkeni ayırarak ve hedef değişkeni kategorik hale getirerek işleme başlandı *(şekil 14).*

```
# Veriyi ve hedefi ayırma
x = df.drop(columns=["price_range"])
y = df["price_range"]
```

```
# Hedef değişkeni kategorik hale getirme
num_classes = len(np.unique(y))
y = to_categorical(y, num_classes=num_classes)
```

Şekil-14 hedef değişkeni ayırma ve kategorik hale getirme

Ardından veri seti %25 i test verisi olmak üzere ayrıldı. Ve standartlaştırma işlemi uygulandı *(Şekil 15).*

```
# Veriyi eğitim ve test setlerine ayırma
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=42)
```

```
# Veriyi standartlaştırma
scaler = MinMaxScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.transform(x_test)
```

Şekil-15 veri setini ayırma ve standartlaştırma işlemi

Modelin tasarlanması: oluşturulan model biri giriş birisi çıkış ve 3 ara katman olmak üzere 5 katmandan oluşmaktadır. İlk ara katmanda 128 nöron ikinci ara katmanda 64 nöron ve üçüncü ara katmanda 32 nöron kullanılmıştır. ilk gizli katmanda ezberlemeyi engellemek adına 0.3 "dropout" uygulanmıştır. gizli katmanlar için "elu" aktivasyon fonksiyonu, çıkış katmanı için çok sınıflı hedef değer olmasından dolayı "softmax" aktivasyon fonksiyonu tercih edilmiştir.

Derleme adımında kayıp fonksiyonu olarak "categorical_crossentropy", optimizer olarak "adam" tercih edilmiştir. *(Şekil 16).*

```
# Modelin tanımlanması
model = models.Sequential()
model.add(layers.Input(shape=(x_train.shape[1],)))
model.add(layers.Dense(128, activation='elu'))
model.add(layers.Dropout(0.3))
model.add(layers.Dense(64, activation='elu'))
# model.add(layers.Dropout(0.3))
model.add(layers.Dense(32, activation='elu'))
# model.add(layers.Dropout(0.3))
model.add(layers.Dense(num_classes, activation='softmax'))

# Modeli derleme
model.compile(loss='categorical_crossentropy',
              optimizer='adam',
              metrics=['categorical_accuracy'])
```

Şekil-16 cnn modelinin tasarlanması

Model eğitimi: Model tasarımında olduğu gibi birçok deme yanılma yolu ile testler yapılmış ve en iyi sonuçların "epoch sayısı" 50 ve "batch_size" değerinin 4 ile alındığı kanısına varılmıştır. *(Şekil 17).*

```
# Modeli eğitme
history = model.fit(x_train, y_train,
                    epochs=50,
                    batch_size=4,
                    validation_data=(x_test, y_test))

# Modelin özetini yazdırma
model.summary()
```

Şekil-17 eğitim verilerinin atanması

5. Model Performansı

Model performansı: Model tasarımında *(şekil 16)*'de yapılan birçok deneme-yanılma yöntemi ile en iyi model performansı elde edilmiştir

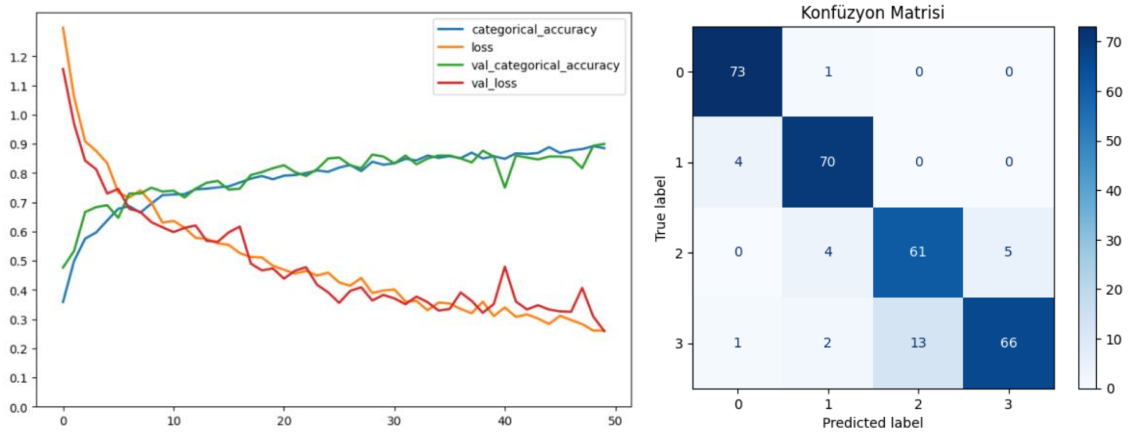
En iyi doğruluk değeri %90 olmak üzere duyarlılık ve hassasiyet gibi metrikler aşağıda belirtilmiştir. *(Şekil 18).*

Classification Report:

	precision	recall	f1-score	support
0	0.94	0.99	0.96	74
1	0.91	0.95	0.93	74
2	0.82	0.87	0.85	70
3	0.93	0.80	0.86	82
accuracy			0.90	300
macro avg	0.90	0.90	0.90	300
weighted avg	0.90	0.90	0.90	300

(Şekil 18). En iyi eğitim sonucuna ait metrik değerler

Son olarak modelin eğitim ve test aşamalarındaki eğrisel grafikleri ve model sonucunda dair konfüzyon matrisleri incelenmiştir *(Şekil 19).*



(Şekil 19). Model eğrileri ve konfüzyon matrisi

6. Sonuç

Bu projede, mobil cihazların fiyat aralığını tahmin etmeye yönelik bir sınıflandırma modeli geliştirilmiştir. Çalışmanın ana bulguları ve sonuçları şu şekildedir**:**

Bu projede, mobil cihazların fiyat aralığını tahmin etmeye yönelik bir sınıflandırma modeli geliştirilmiştir. Veri setinin analizi ve ön işleme sürecinde, eksik değer bulunmaması çalışmayı önemli ölçüde kolaylaştırmıştır. Özellik mühendisliği kapsamında eklenen "battery_feature_2" değişkeni, model performansına kayda değer bir katkı sağlamıştır. Ayrıca, aykırı değerlerin başarılı bir şekilde tespit edilip baskılanması ve uygulanan normalizasyon işlemleri, model güvenilirliğini artırmış ve eğitim sürecinin daha etkin gerçekleşmesini sağlamıştır.

Geliştirilen CNN modeli, %90 doğruluk oranı ile oldukça başarılı bir performans sergilemiştir. Model mimarisinde kullanılan dropout katmanı (0.3) aşırı öğrenmeyi engellemede etkili olurken, seçilen hiperparametreler (epoch=50, batch_size=4) optimum sonuçlar vermiştir. Konfüzyon matrisi sonuçları incelendiğinde, modelin tüm sınıflar için dengeli bir tahmin yeteneği geliştirdiği açıkça görülmektedir. Bu sonuçlar, seçilen model mimarisinin ve hiperparametrelerin problemin çözümü için uygun olduğunu göstermektedir.

Korelasyon analizi ve özellik önem dereceleri incelendiğinde, fiyat aralığı tahmini üzerinde en etkili özelliklerin sırasıyla:

- "battery_feature_2"
- "battery"
- "px_height"
- "px_widht"

Şeklinde olmuştur.

İleriki çalışmalarda modelin performansını daha da artırmak için bazı geliştirmeler yapılabilir. Öncelikle, veri seti boyutunun artırılması modelin genelleme yeteneğini güçlendirebilir. Farklı model mimarileri (örneğin LSTM, GRU) ile karşılaştırmalı analizler yapılarak en optimal çözüm bulunabilir. Bununla birlikte, daha fazla özellik mühendisliği tekniği uygulanarak model hassasiyeti artırılabilir ve ensemble öğrenme teknikleri kullanılarak tahmin başarısı yükseltilebilir.

Bu çalışma, derin öğrenme tekniklerinin mobil cihaz fiyat tahmininde etkili bir şekilde kullanılabileceğini göstermiştir. Elde edilen sonuçlar, benzer sınıflandırma problemleri için bir referans teşkil edebilir ve gelecek çalışmalara ışık tutabilir. Özellikle e-ticaret ve fiyatlandırma stratejileri alanında yapılacak çalışmalar için önemli bir temel oluşturabilir.