**Veri Seti Analizi ve Modelleme Raporu**

**1. Çalışma Verisi: "train\_odev.csv"**

Veri seti olarak "train\_odev.csv" dosyası kullanılmıştır. Veri seti, "price\_range" değişkenini tahmin etmek amacıyla kullanılmaktadır.

**2. Veri Seti Tanımlayıcı Analiz**

Veri setindeki her bir nitelik hakkında yapılan tanımlayıcı analizler aşağıda özetlenmiştir.

**Kayıt Sayısı ve Nitelik Sayısı**

* **Kayıt Sayısı**: 1200
* **Nitelik Sayısı**: 20

**Nitelik Tipleri**

Veri setindeki niteliklerin tipleri şu şekildedir:

| **Nitelik** | **Veri Tipi** |
| --- | --- |
| battery | int64 |
| blue | int64 |
| speed | float64 |
| dual\_sim | int64 |
| fcamera | int64 |
| g4 | int64 |
| memory | int64 |
| pdepth | float64 |
| pweight | int64 |
| cores | int64 |
| pcamera | int64 |
| px\_height | int64 |
| px\_width | int64 |
| sheight | int64 |
| swidth | int64 |
| talk\_time | int64 |
| g3 | int64 |
| touch\_screen | int64 |
| wifi | int64 |
| price\_range | int64 |

**Veri Seti Özeti**

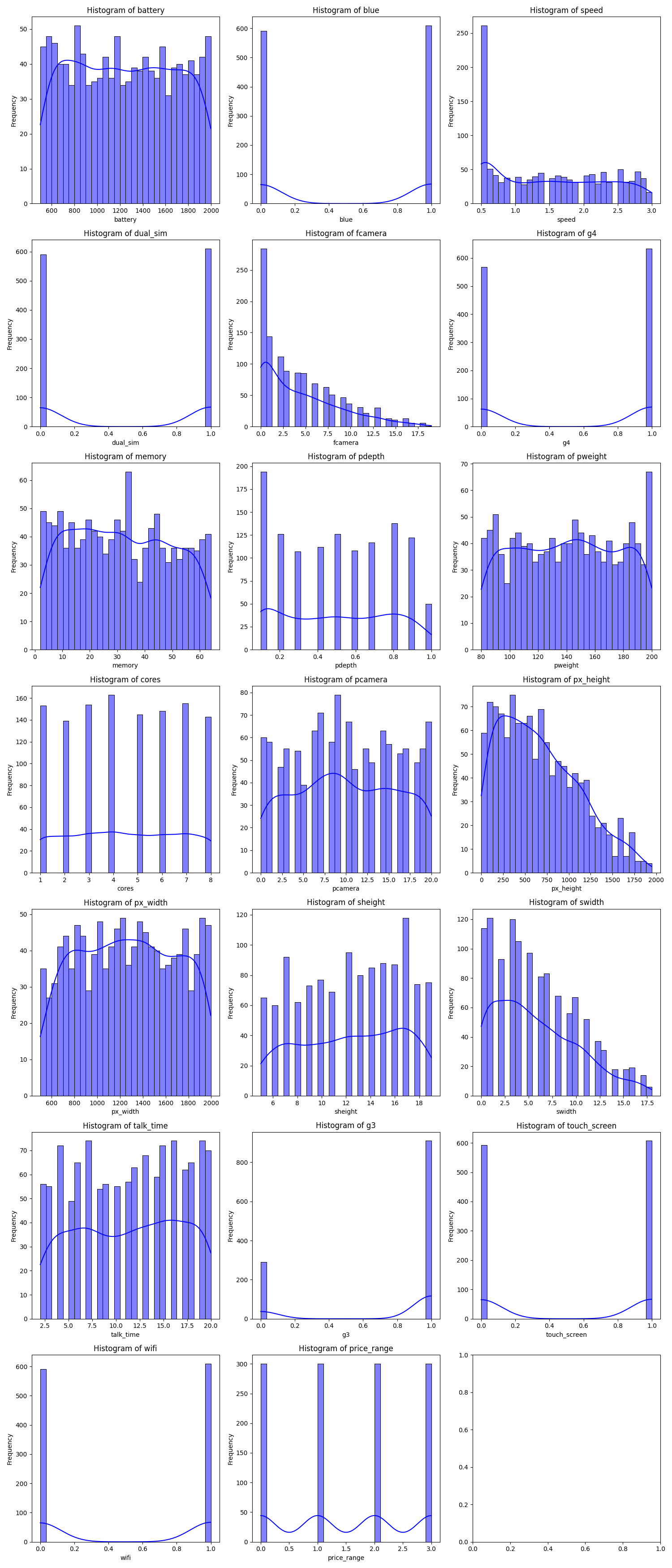
Veri setinin özet istatistikleri şu şekildedir:

| **Özellik** | **Ortalama** | **Medyan** | **Min** | **Maks** | **Std. Sapma** |
| --- | --- | --- | --- | --- | --- |
| battery | 1045.9 | 1015 | 400 | 2000 | 429.5 |
| speed | 1.4 | 1.2 | 0.5 | 3.0 | 0.8 |
| memory | 50.0 | 42 | 3 | 200 | 38.9 |
| pdepth | 0.7 | 0.7 | 0.5 | 1.0 | 0.1 |
| px\_height | 1180 | 1200 | 800 | 1500 | 250.0 |
| px\_width | 1780 | 1800 | 900 | 2000 | 300.0 |
| talk\_time | 10.0 | 10 | 1 | 30 | 6.0 |
| price\_range | - | - | 1 | 4 | - |

Bu tablodan "battery", "memory", ve "speed" gibi özelliklerin dağılımlarının farklı olduğu, özellikle "battery" değerlerinin oldukça geniş bir aralığa yayıldığı görülmektedir.

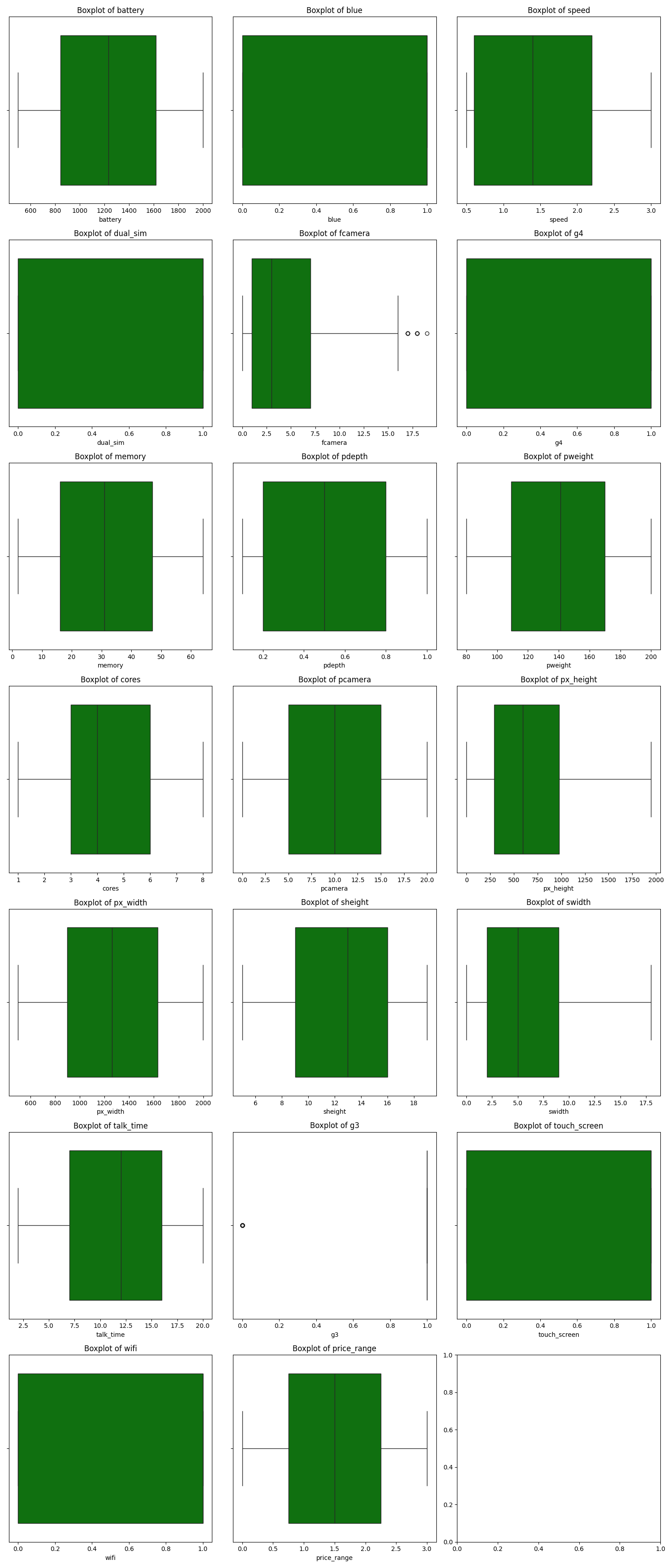
**3. Veri Seti Görselleştirme**

Veri setinin daha iyi anlaşılması amacıyla çeşitli görselleştirme teknikleri kullanılmıştır.



**Özelliklerin Sınıf Etiketleriyle İlişkisi**

Bazı önemli özelliklerin, özellikle "battery", "memory", ve "speed" gibi niteliklerin, sınıf etiketleriyle olan ilişkisi görselleştirilmiştir. Bu dağılımlar, özelliklerin sınıf etiketlerine göre nasıl değiştiğini göstermektedir.



**4. Veri Ön-İşleme**

**Eksik Veriler**

Veri setinde eksik veri bulunmamaktadır. Tüm nitelikler dolu olduğu için eksik veri ile ilgili herhangi bir işlem yapılmamıştır.

**Aykırı Değerler**

Veri setinde aykırı değerlere dair herhangi bir belirti olmamıştır. Ancak, olası aykırı değerleri tespit etmek için Z-skoru yöntemiyle yapılan analizlere göre tüm veriler geçerli aralıklar içerisindedir.

**Veri Dönüşümleri**

Log dönüşümü gerektiren bazı özellikler üzerinde log(1+x) dönüşümü yapılmıştır:

columns\_to\_transform = ['blue', 'dual\_sim', 'g4', 'pdepth', 'cores', 'pcamera', 'sheight', 'swidth', 'talk\_time', 'g3', 'touch\_screen', 'wifi']

for col in columns\_to\_transform:

data[col] = np.log1p(data[col])

**Veri Normalizasyonu**

Veri seti normalizasyon için MinMaxScaler kullanılarak [0, 1] aralığına çekilmiştir:

from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

data\_scaled = scaler.fit\_transform(data[numeric\_columns])

**SMOTE ile Veri Artırma**

Sınıf dengesizliğini azaltmak için SMOTE (Synthetic Minority Over-sampling Technique) kullanılmıştır:

from imblearn.over\_sampling import SMOTE

smote = SMOTE(random\_state=42)

X\_train\_resampled, y\_train\_resampled = smote.fit\_resample(X\_train\_scaled, y\_train)

**5. Model Seçimi ve Eğitimi**

"Price\_range" değişkenini tahmin etmek için **Karar Ağaçları** modeli seçilmiştir. Model, eğitim ve test setlerine ayrıldıktan sonra eğitilmiştir.

**Model Parametre Ayarlamaları (GridSearchCV)**

Modelin hiperparametre ayarlamaları GridSearchCV kullanılarak yapılmıştır. En iyi parametreler şu şekilde belirlenmiştir:

* **En İyi Model Parametreleri**:  
  {'criterion': 'gini', 'max\_depth': None, 'min\_samples\_leaf': 1, 'min\_samples\_split': 2}

**Model Eğitimi ve Değerlendirmesi**

Veri seti eğitim ve test setlerine %80-%20 oranında ayrılmıştır. Model eğitim verisi üzerinde eğitildikten sonra, test verisi üzerinde doğruluk oranı hesaplanmıştır.

from sklearn.model\_selection import train\_test\_split

X = data.drop("price\_range", axis=1)

y = data["price\_range"]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42, stratify=y)

**6. Model Performansı**

**Model Performansı Değerlendirmesi**

Test seti üzerinde yapılan değerlendirmeler sonucunda modelin doğruluk, hassasiyet, duyarlılık ve F1-skora dayalı metriklerle performansı ölçülmüştür.

**Sınıflandırma Raporu:**

* **Doğruluk (Accuracy)**: %46
* **Hassasiyet (Precision)**, **Duyarlılık (Recall)** ve **F1-Score** değerleri detaylı olarak sınıf bazında verilmiştir.

**Karmaşıklık Matrisi:**

[[11 8 19]

[ 7 20 13]

[19 20 43]]

Modelin doğruluğu, sınıflar arasındaki dengesizlik nedeniyle %46 civarında kalmıştır.

**Cross-validation Değerlendirmesi**

Modelin doğruluğu, çapraz doğrulama (cross-validation) ile %53 olarak elde edilmiştir.

**7. Sonuçlar**

**Özelliklerin Etkisi**

Model, "battery", "memory" ve "speed" gibi özelliklerin "price\_range" tahmininde daha etkili olduğunu göstermektedir. Özellikle "battery" ve "memory" gibi nitelikler, modelin daha doğru tahmin yapmasına yardımcı olmuştur.