# Kredi Riski Sınıflandırılması Proje Raporu

Hazırlayan :Melike Su Koçyiğit

# Proje Metodolojisi

**Problem Tanımı ve Amaç:**
Bu projenin amacı, bireyleri bir dizi özellik temelinde "iyi" veya "kötü" kredi riski olarak sınıflandırmaktır. Lojistik Regresyon ve KNN Sınıflandırma modellerini uygulayarak bu sınıflandırmaları gerçekleştirip performansları değerlendirilenecek ve sonuçları karşılaştırılanacaktır.

## 1. Veri Keşfetme (EDA)

Veriyi anlamak ve keşfetmek için Exploratory Data Analysis (EDA) yapacağız. Bu aşamada çeşitli görselleştirmeler ve istatistiksel analizler gerçekleştirilecektir.

`pandas (read_csv )` fonksiyonu kullanılarak veri seti yüklendi. Veri seti, 1000 gözlem ve 20 değişkenden oluşmaktadır.

Veriyi daha iyi tanımak için:

```
data.head()

data.info()

data.columns
```

fonksiyonları kullanıldı.

Kullanılmayan sütunlar (Unnamed 0.1 ve Unnamed 0 ) `(drop )` metodu ile veri setinden çıkarıldı..

**A. Eksik Değer Analizi**
Eksik değerler data.isnull().sum() fonksiyonuyla tespit edildi.Aykırı değerler tespit edildi. Eksik değerler "Saving accounts" ve "Checking account" sütunlarında gözlenmiştir.

**B. Dağılım Analizleri ve Veri Görselleştirme**

Bu aşamada `Matplotlib` ve `Seaborn` kütüphanelerini kullandım.

- **Sayısal Değişkenlerin Dağılımı**: `Age`, `Credit amount`, `Duration` değişkenleri için histogram veya kutu grafikleri çizildi. Aykırı değerleri belirlendi.

# AGE


**Histogram Grafiği Yorumlaması**

- **Dağılım Yapısı:** Age değişkeni sağa çarpıktır, genç yaşlarda daha fazla veri varken, yaş arttıkça veri sayısı azalır.

- **Yoğun Yaş Aralıkları:** En yoğun veri 25-30 yaş aralığındadır; 20-40 yaş arasında yoğun bir veri kümesi bulunmaktadır.

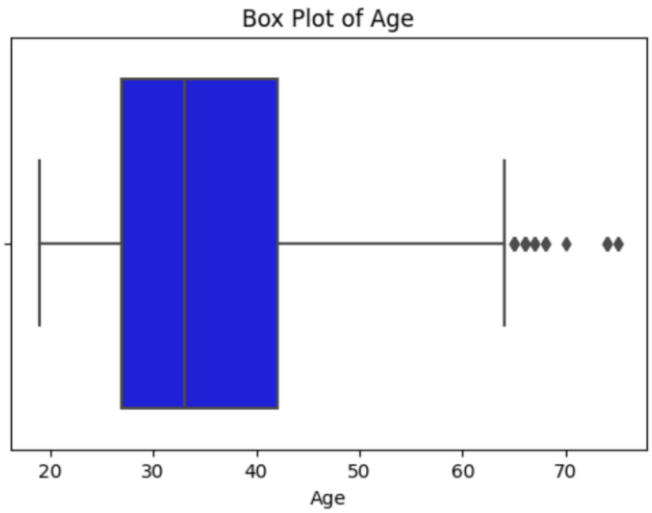
- **Düşük Yoğunluk:** 50 yaş ve üzerindeki bireylerin sayısı belirgin şekilde azalmaktadır, özellikle 60 yaş üstü çok düşük.

- **KDE Çizgisi:** Yaş arttıkça yoğunluk azalır.

- **Aykırı Değerler:** 20 yaş altı ve 60 yaş üstü bireyler potansiyel aykırı değerlerdir.

- **Yoğunluk ve Veri Dengesi:** Veri çoğunluğu 20-50 yaş aralığındadır.

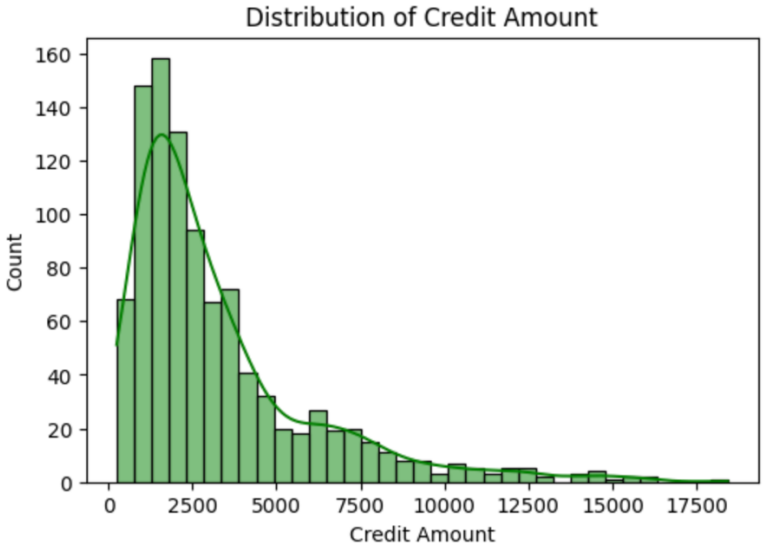
- **Eksiklikler:** 70 yaş ve üzeri bireylerin sayısı çok azdır, yaşlı bireyler yeterince temsil edilmemiştir.




**Kutu Grafiği Yorumlaması**

1. **Merkezi Eğilim ve Yayılım**:
   - Medyan yaklaşık 30-35 yaş arasındadır.
   - Kutunun alt ve üst kenarları, verinin %50'sini temsil eder (Q1 ve Q3).
2. **İç ve Dış Sınırlar**:
   - Çizgiler, minimum ve maksimum değerleri gösterir (aykırı değerler hariç).
3. **Aykırı Değerler**:
   - 60 yaş ve üzerindeki bazı bireyler aykırı değer olarak görülür.
4. **Veri Yoğunluğu**:
   - Verilerin çoğu 20-40 yaş aralığında yoğunlaşmıştır.

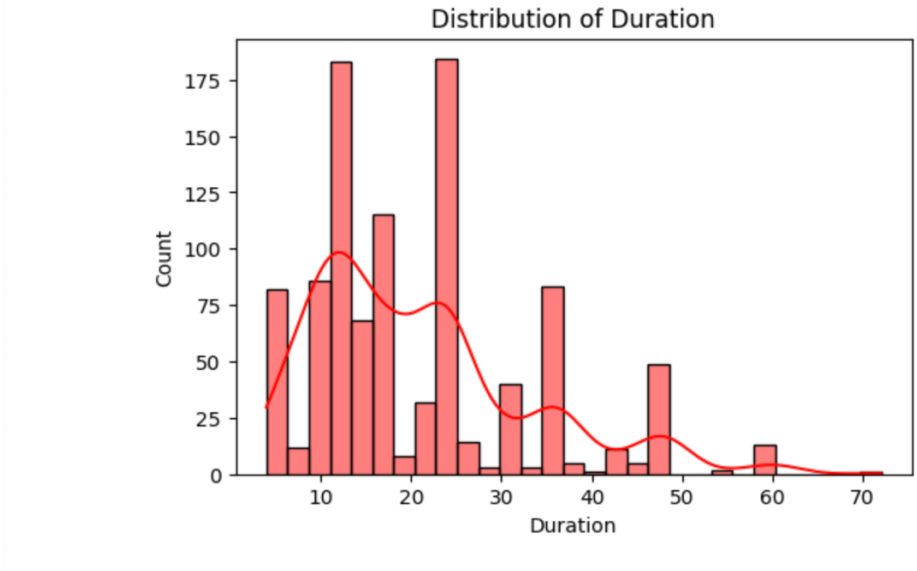
**CREDİT AMOUNT**



**Histogram Grafiği Yorumlaması**

- **Dağılım Yapısı**: Credit Amount (Kredi Miktarı) sağa çarpık bir dağılıma sahiptir; düşük kredi miktarları daha yoğundur, yüksek miktarlar ise azalmaktadır.
- **Yoğun Kredi Aralıkları**: En yoğun veri 2000-3000 aralığında bulunmakta, 0-5000 arasında yoğun bir küme gözlenmektedir.
- **Düşük Yoğunluk**: 10.000 ve üzeri kredi miktarları nadir olup düşük bir yoğunluk sergilemektedir.
- **KDE Çizgisi**: Kredi miktarı arttıkça yoğunluk düzenli şekilde azalmaktadır.
- **Aykırı Değerler**: 10.000 üzerindeki kredi miktarları potansiyel aykırı değer olarak değerlendirilebilir.
- **Yoğunluk ve Veri Dengesi**: Veri ağırlığı düşük kredi miktarlarına yoğunlaşmıştır.

**DURATION**




- **Dağılım Yapısı**: "Duration" değişkeni sağa çarpık bir dağılıma sahiptir; düşük süreler daha yaygındır, yüksek süreler ise azalmaktadır.

- **Yoğun Süre Aralıkları**: En yoğun veri 10-20 ay aralığındadır; özellikle 12 ve 18 aylık süreler dikkat çekicidir.

- **Düşük Yoğunluk**: 40 ay ve üzerindeki sürelerin frekansı oldukça düşüktür.

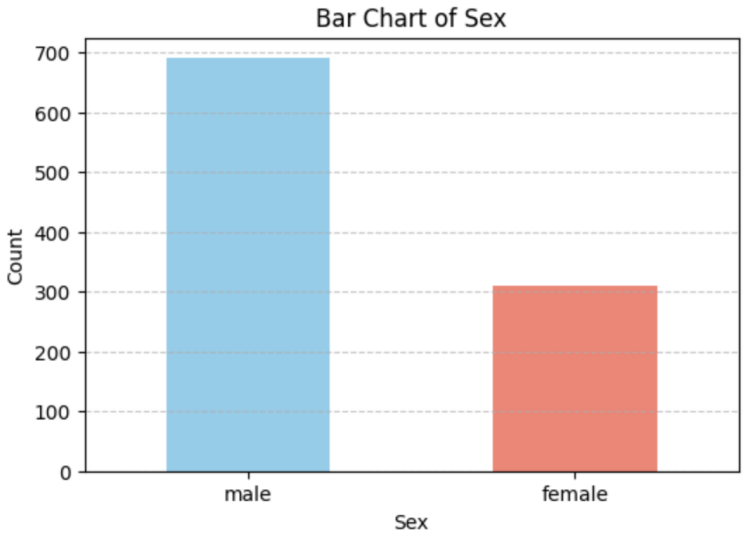
- **KDE Çizgisi**: Süre arttıkça yoğunluk genelde azalmaktadır, ancak bazı sürelerde küçük tepe noktaları vardır.

- **Aykırı Değerler**: 50 ay ve üzerindeki süreler potansiyel aykırı değer olarak değerlendirilebilir.

- **Yoğunluk ve Veri Dengesi**: Veri çoğunluğu 10-40 ay aralığında yoğunlaşmıştır.

**Kategorik Değişkenlerin Sayımları**: `Sex`, `Housing`, `Purpose` değişkenleri için çubuk grafikler oluşturuldu.

**SEX**

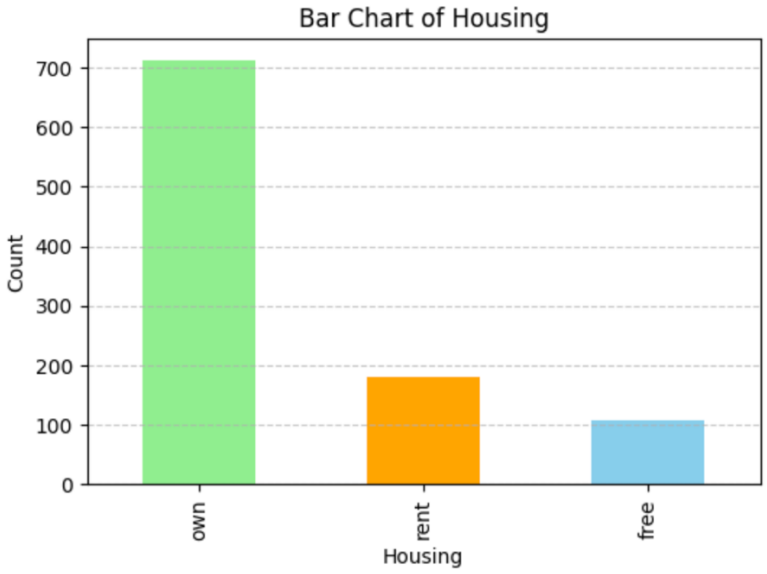



- **Dağılım Yapısı**: Cinsiyet değişkeninde "male" kategorisi "female" kategorisinden daha fazla bireyi kapsamaktadır.

- **Yoğunluk Farkı**: Erkek birey sayısı yaklaşık 700 civarında, kadın birey sayısı ise yaklaşık 300 civarındadır.

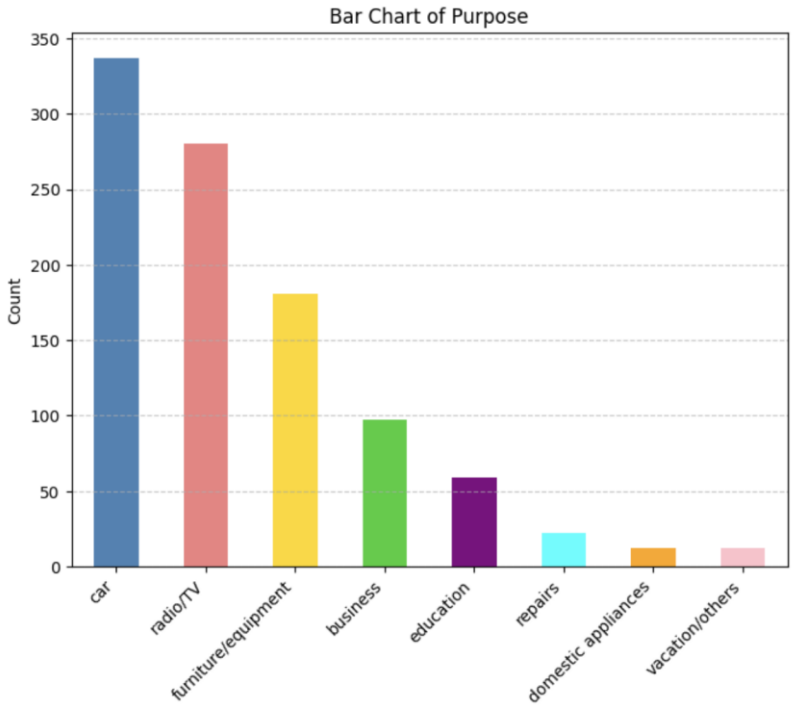
- **Dengesizlik**: Veride erkek bireylerin oranı kadınlara kıyasla belirgin şekilde fazladır.

## HOUSING



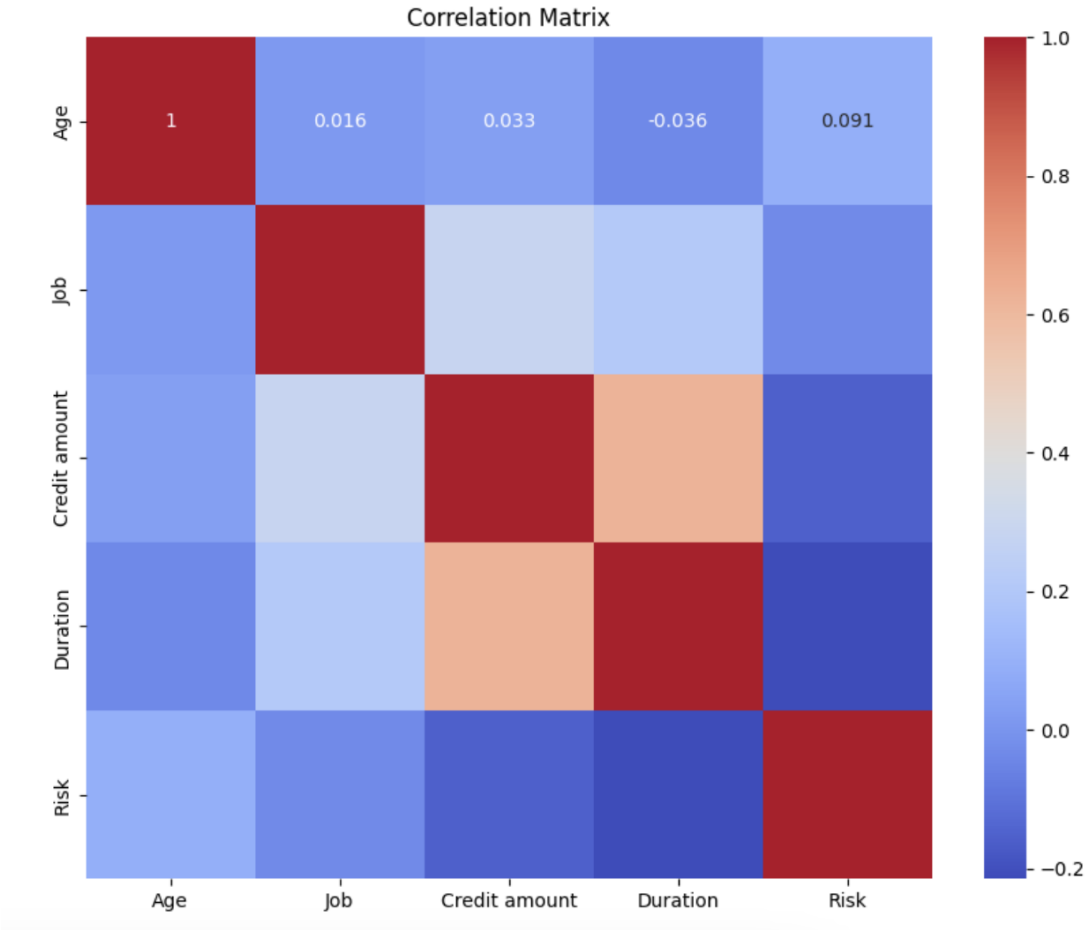
- **Dağılım Yapısı**: "own" kategorisi (ev sahibi) en yüksek sayıya sahiptir ve yaklaşık 700 bireyi kapsamaktadır.
- **Yoğunluk Farkı**: "rent" (kiralık) kategorisi yaklaşık 200 bireyi temsil ederken, "free" (ücretsiz) kategorisi yaklaşık 100 bireyi kapsamaktadır.
- **Dengesizlik**: Veride ev sahipleri belirgin şekilde daha fazla temsil edilmiştir.

## PURPOSE




- **Dağılım Yapısı**: En fazla tercih edilen amaç "car" (araba) olup yaklaşık 350 bireyi kapsamaktadır.
- **Yoğunluk Sıralaması**: İkinci sırada "radio/TV" (radyo/televizyon) yaklaşık 300 birey ile yer alırken, bunu "furniture/equipment" (mobilya/ekipman) yaklaşık 200 birey ile takip etmektedir.
- **Düşük Yoğunluk**: "vacation/others" (tatil/diğer) ve "domestic appliances" (ev aletleri) gibi kategoriler çok düşük bir oranda temsil edilmiştir.
- **Dengesizlik**: "car", "radio/TV" ve "furniture/equipment" amaçları, verinin büyük çoğunluğunu oluşturarak diğer kategorilere göre belirgin bir dengesizlik sergilemektedir.

**Korelasyon Haritası**: Sayısal özellikler arasındaki ilişkileri anlamak için korelasyon haritası oluşturuldu.




**Age (Yaş):**

- Yaşın diğer değişkenlerle olan korelasyonu oldukça düşük (en fazla 0.091), yani yaşın bu değişkenlerle anlamlı bir doğrusal ilişkisi yok.

**Job (İş):**

- İş değişkeni ile diğer değişkenler arasında belirgin bir ilişki yok (korelasyon değerleri ~0).

**Credit Amount (Kredi Tutarı):**

- **Credit Amount ve Duration:** 0.33 gibi orta seviyede pozitif bir ilişki var. Kredi tutarının artması genellikle kredi süresinin de artmasıyla ilişkili olabilir.
- Diğer değişkenlerle olan korelasyon oldukça düşük.

**Duration (Süre):**

- Süre ile Risk arasında anlamlı bir ilişki yok (korelasyon ~0).
- Süre ve Kredi Tutarı arasında pozitif bir ilişki mevcut.

**Risk:**

- Risk değişkeninin diğer değişkenlerle olan korelasyonu oldukça düşük. Bu, Risk değişkeninin bu faktörlerle doğrusal olarak fazla etkilenmediğini gösterebilir.

## Genel Değerlendirme:

- Çoğu değişken arasında korelasyon yok veya çok düşük. Bu durum, değişkenlerin birbirinden bağımsız olabileceğini veya doğrusal olmayan ilişkilerin olabileceğini gösteriyor. Bunu çözmek için özellik mühendisliği ilerleyen adımlarda kullanacağım.

### C. Soruların Cevaplanması

1. Veri setinde eksik değerler var mı? Hangi kolonlarda var ve bunlarla nasıl başa çıkılacak?

   Eksik değerleri data.isnull().sum() fonksiyonuyla tespit ettim ve sonuçlarda Saving acounts'da 183 tane, Checking account'da 394 tane eksik değerler olduğunu fark ettim. Bunun önüne ilerleyen adımlarda eksik değerleri medyan ile doldurarak başa çıkılacak.

2. `Age`, `Credit amount`, `Duration` değişkenlerinin dağılımları nedir? Aykırı değerler var mı?

`data.describe()` fonksiyonu ile temel istatistikleri ve değişken dağılımlarını inceledim.

İstatistiksel Özet:

```
- Age: Ortalama 35.5, minimum 19, maksimum 75

- Credit amount: Ortalama 3271.3, minimum 250, maksimum
18424

- Duration: Ortalama 20.9 ay, minimum 4 ay, maksimum 72 ay

- Risk: %70 iyi kredi riski, %30 kötü kredi riski
```

Interquartile Range (IQR) yöntemi ile aykırı değerler belirlendi. Bu analiz `Age`, `Credit amount` ve `Duration` sütunları için yapıldı.Bunun metodunu chatgpt'den destek alarak gerçekleştirdim. Aynı şekilde grafikler üzerinden de yorumlandı.

Sonuçlar:

```
Age Outliers Analysis:
Lower Bound: 4.5
Upper Bound: 64.5
Number of Outliers: 23
```

```
Credit Amount Outliers Analysis:
Lower Bound: -2544.625
Upper Bound: 7882.375
Number of Outliers: 72
```

Lower Bound'un Negatif olması veri dağılımının asimetrik olmasındandır.

```
Duration Outliers Analysis:
Lower Bound: -6.0
Upper Bound: 42.0
Number of Outliers: 70
```

3. Hedef kolonunda (`iyi` ve `kötü` kredi riski) oran nasıldır? Veri dengesiz mi?

   `data['Risk'].value_counts(normalize=True)` ile oranları hesapladım.

   ```
   1    0.7
   0    0.3
   ```

   Bu durum, verinin dengesiz olduğunu gösteriyor çünkü "iyi kredi riski" (1) çok daha yaygın ve "kötü kredi riski" (0) daha az temsil ediliyor. Bu tür veri setlerinde sınıf dengesizliği (class imbalance) olabilir, bu da modelin kötü kredi riskini (0) daha az öğrenmesi veya tahmin etmesi anlamına gelebilir.

4. `İyi` kredi riski kategorisindeki bireylerin ortalama `Credit amount` değeri nedir?

   ```
   mean_credit_good_risk = data[data['Risk'] == 1]['Credit
   amount'].mean()
   ```

   ile sonucu `2985.457142857143 buldum.`

5. `Free` (bedava) konut kategorisindeki bireylerin `Saving accounts` değişkeninin dağılımı nasıldır?

   ```
   free_housing_saving_accounts = data[data['Housing'] ==
   'free']['Saving accounts'].value_counts(normalize=True)
   ```

   Sonuç:

   ```
   little        0.788235
   moderate      0.117647
   quite rich    0.070588
   rich          0.023529
   ```

   Bireyler en çok ‘Little’ kategorisindedir.

6. İyi ve kötü kredi riski grupları arasında Duration farklılık gösteriyor mu?

```
duration_good = data[data['Risk'] == 1]['Duration']

duration_bad = data[data['Risk'] == 0]['Duration']

# Temel istatistiklerin hesaplanması

stats_good = duration_good.describe()

stats_bad = duration_bad.describe()
```

kodlarını kullanarak ekrana yazdırdım.

```
İyi Kredi Riski (Risk = 1) - Duration İstatistikleri:
count    700.000000
mean      19.207143
std       11.079564
min        4.000000
25%       12.000000
50%       18.000000
75%       24.000000
max       60.000000
Name: Duration, dtype: float64

Kötü Kredi Riski (Risk = 0) - Duration İstatistikleri:
count    300.000000
mean      24.860000
std       13.282639
min        6.000000
25%       12.000000
50%       24.000000
75%       36.000000
max       72.000000
```

- Kötü kredi riski grubu, genellikle daha uzun vadeli ve daha değişken kredi sürelerine sahip.

- Kısa vadeli krediler (özellikle 18 ay ve altı), genellikle iyi kredi riski grubunda daha yaygın.

Bu durum, uzun vadeli kredilerin daha yüksek bir riskle ilişkilendirilebileceğini ve kredi verenlerin bu sürelerde daha dikkatli olmaları gerektiğini gösterebilir.

7. Yüksek kredi miktarına sahip bireylerin (75. yüzde dilimi üzerinde) en sık kullandığı 3 `Purpose` kategorisi nedir?

   Burada kodda atladığım bazı noktalar olduğundan chatten destek aldım.

```
 high_credit = data[data['Credit amount'] > data['Credit
amount'].quantile(0.75)]

top_purposes =
high_credit['Purpose'].value_counts().head(3)

print(top_purposes)
```

```
car         108
radio/TV     39
business     39
```

## 2. Veri Temizleme ve Ön İşleme

### A. Eksik Değerlerin Ele Alınması

- **Kategorik Değişkenler**: `Saving accounts` ve `Checking account` gibi kategorik değişkenlerde eksik değerler dolduruldu ve kategoriler sayısal değerlere dönüştürüldü.

`fillna('unknown')` yöntemiyle eksik değerler "unknown" kategorisiyle dolduruldu.

`map()` fonksiyonuyla kategorik veriler, belirli bir sözlük (`account_mapping`) kullanılarak sayısal değerlere dönüştürüldü.

- **Sayısal Değişkenler**: Sayısal değişkenlerde eksik değerler medyan ile doldurulacaktır.

  `select_dtypes(include='number')` yöntemiyle veri küpündeki yalnızca sayısal (numeric) kolonlar seçildi.

  `fillna()` yöntemiyle sayısal değişkenlerdeki eksik değerler, o kolonun **medyan** değeriyle dolduruldu.

### B. Kategorik Değişkenlerin Kodlanması

Burada data.dtypes yaptığımda normalde kategorik fakat sayısal gibi davranan değişkenler fark ettim. Örneğin

```
Saving accounts     float64
Checking account      int64
```

**One-Hot Encoding**: `Purpose` , `Sex`, `Saving acounts`, `Checking acount` ve `Housing` gibi birden fazla kategoriye sahip değişkenlerde one-hot encoding uygulandı.

Bunun için:

`from sklearn.preprocessing import OneHotEncoder` kütüphanesi kullanıldı.

### C. Özellik Ölçekleme

**StandardScaler**: `Age`, `Credit amount`, `Duration` gibi sayısal özellikler aynı aralıkta olacak şekilde ölçeklenecektir.

StandartScaler, verilerin ortalamasını ve standart sapmasını hesaplayarak her değeri standart bir forma çevirir.

```
scaler = StandardScaler()
```

`fit_transform()`: Verilerin ortalaması ve standart sapmasını hesaplar (**fit**), ardından bu bilgileri kullanarak ölçekler (**transform**).

Seçilen değişkenler: `'Age'`, `'Credit amount'`, `'Duration'`.

```
data[['Age', 'Credit amount', 'Duration']] =
scaler.fit_transform(data[['Age', 'Credit amount',
'Duration']])
```

## 3. Özellik Mühendisliği

### A. Yeni Özellikler Oluşturma

- **Aylık Kredi Miktarı**: `Credit amount` özelliği `Duration` ile bölünerek hesaplandı.
- **Yaş Kategorileri**: Opsiyonel olarak yaşlar kategorilere ayrıldı

**Yaş Aralıkları Belirleme (`bins`)**:

- `bins` değişkeni, yaşları farklı kategorilere ayırmak için kullanılan sınır değerlerini belirtir.
  - Örneğin:benim projemde
  - Çocuk → 0
  - Genç→ 1
  - Orta → 2
  - Yetişkin → 3
  - Yaşlı → 4

**Kategorilere Etiket Atama (`labels`)**:

- `labels` değişkeni, her kategoriye bir numaralı etiket atanmasını sağlar.
  - Örneğin: 0 → Çocuk, 1 → Genç, vb.

**Kesikli Veriler Oluşturma (`pd.cut`)**:

- `pd.cut()` fonksiyonu, yaş değerlerini belirlenen sınırlar (`bins`) ve etiketler (`labels`) kullanarak kategorilere ayırır.
- `right=False`: Üst sınırın dahil olmadığını belirtir.

**Yeni Kolon Eklenmesi (`Age Category`)**:

- `data['Age Category']`: Kategorilere ayrılmış yaş verileri bu yeni kolona eklenmiştir.

## B. Yüksek Korelasyona Sahip Özellikler

- **Korelasyon Analizi**: Korelasyon katsayısı > 0.9 olan özellikler belirlenip kaldırıldı.

  **Korelasyon matrisi:**

  Yüksek korelasyon, bilgi tekrarına işaret eder. Modelleme sırasında bu tekrar sorun yaratabilir.Bu yüzden bu aşama daha doğru, basit ve genelleştirilebilir modeller oluşturmak için önemli bir adımdır.

  ```
  correlation_matrix = data.corr(numeric_only=True)

  correlation_matrix
  ```

  ile korelasyon matrisi oluşturuldu.

  **0.9'un üzerinde korelasyonlar:**

  ```
  high_correlation = correlation_matrix[(correlation_matrix
  > 0.9) & (correlation_matrix < 1.0)]

  high_correlation
  ```

  bu adımları uyguladığımda:

| | Age | Credit amount | Duration | Risk | Purpose_car | Purpose_domestic appliances | Purpose_education | Purpose_fur |
|---|---|---|---|---|---|---|---|---|
| **Age** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Credit amount** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Duration** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Risk** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Purpose_car** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Purpose_domestic appliances** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Purpose_education** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **_furniture/equipment** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Purpose_radio/TV** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |

Görüldüğü üzere tüm değerlerim NaN çıktı.

Bu da sonuç olarak:

- Verimdeki özellikler birbirinden bağımsız bir şekilde iyi dağılmıştır. Hiçbir iki değişken arasında yüksek düzeyde ilişki olmadığı için, veri temizleme aşamasında bu bağlamda bir çıkarma işlemi yapmam gerekmez.

- Bu matris, özellikler arasında düşük ilişkiler olduğunu (genellikle 0'a yakın değerler) ve bunların modelin performansı için problem yaratmayacağını gösterir.

# 4. Model Uygulaması

## A. Lojistik Regresyon

- **Model Eğitimi**: Ön işlenmiş veri seti kullanılarak bir Lojistik Regresyon modeli eğitilecektir.
-

**Feature ve Target değerleri belirleme**:

```
X = data.drop('Risk', axis=1)

y = data['Risk']
```

**Veriyi Bölme:**

```
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

`X_train.shape ve X_test.shape` ile test ve trainlerin boyutlarının doğruluğunu kontrol edildi.

**Lojistik Regresyon Modeli:**

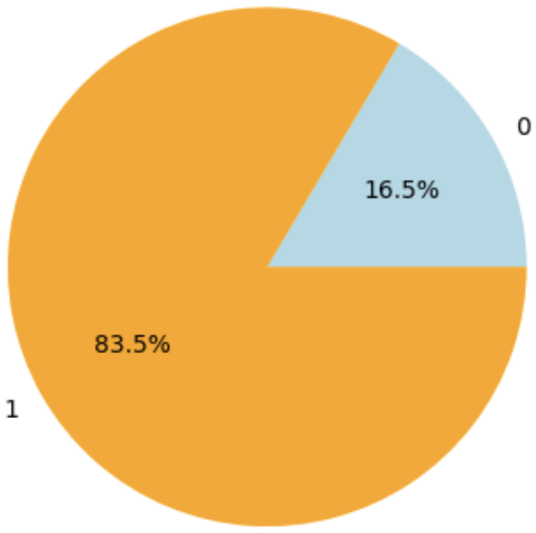
```
log_reg = LogisticRegression(max_iter=1000)

log_reg.fit(X_train, y_train)
```

**Lojistik Regresyon Tahminleri:**

```
y_pred = log_reg.predict(X_test)
```

Daha sonra bu tahmin edilen değerlerin yüzdesel dağılımını net görmek için bir pie chart oluşturuldu.

```
plt.pie([sum(y_pred == 0), sum(y_pred == 1)],
labels=['0', '1'], autopct='%1.1f%%',
colors=['lightblue', 'orange'])

plt.title('Tahmin Değerlerinin Yüzdesel Dağılımı')

plt.show()
```



- **Model Değerlendirmesi**: Doğruluk (Accuracy), Kesinlik (Precision), Duyarlılık (Recall) ve F1-Skoru metrikleri ile model performansı değerlendirilecektir.

```
accuracy_log_reg = accuracy_score(y_test, y_pred)

precision_log_reg = precision_score(y_test, y_pred)

recall_log_reg = recall_score(y_test, y_pred)

f1_log_reg = f1_score(y_test, y_pred)



print("Accuracy:", accuracy_log_reg)

print("Precision:", precision_log_reg)

print("Recall:", recall_log_reg)

print("F1-Skoru:", f1_log_reg)
```

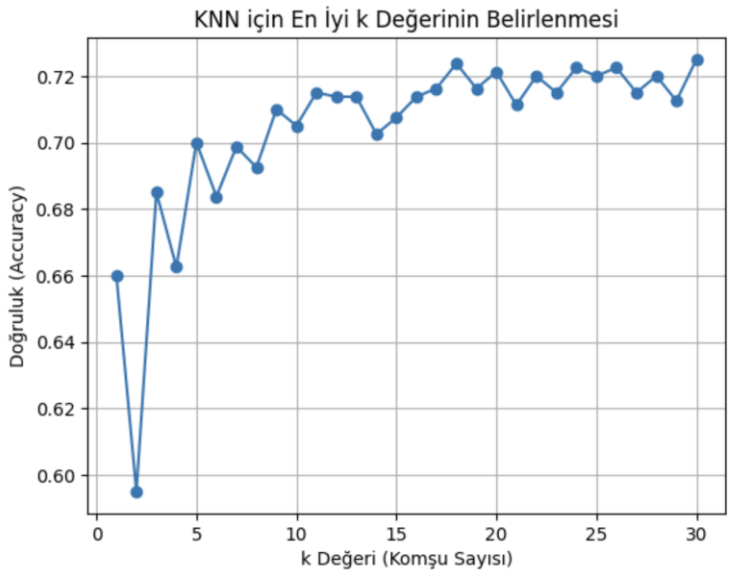
```
Accuracy: 0.75
Precision: 0.7724550898203593
Recall: 0.9148936170212766
F1-Skoru: 0.8376623376623377
```

**B. KNN Sınıflandırma**

- **Hiperparametre Optimizasyonu**: Çapraz doğrulama kullanarak en iyi `k` değeri belirlenecektir.

```
k_degerleri = range(1, 31)

cv_skorlari = []



for k in k_degerleri:

  knn = KNeighborsClassifier(n_neighbors=k)

  skorlar = cross_val_score(knn, X_train, y_train, cv=10,
scoring='accuracy')

  cv_skorlari.append(skorlar.mean())
```

**Çizgi grafiği ile k değerlerini görselleştirme:**




**En iyi k değeri:**

```
en_iyi_k =
k_degerleri[cv_skorlari.index(max(cv_skorlari))]

print(f"En iyi k değeri: {en_iyi_k}")
```

```
En iyi k değeri:
```

- **Model Eğitimi ve Değerlendirmesi**: En iyi `k` değeri ile model eğitilecek ve değerlendirilecektir.

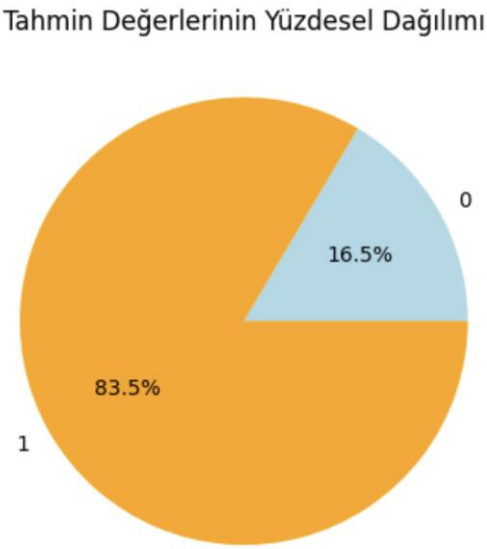
```
knn = KNeighborsClassifier(n_neighbors=en_iyi_k)
knn.fit(X_train, y_train)
```

**Tahminlerde bulunma:**

```
y_pred_knn = knn.predict(X_test)
```

Tahmin edilen değerlerin yüzdesel dağılımını net görmek için bir pie chart oluşturuldu.

```
plt.pie([sum(y_pred_knn == 0), sum(y_pred_knn == 1)],
labels=['0', '1'], autopct='%1.1f%%',
colors=['lightblue', 'orange'])

plt.title('Tahmin Değerlerinin Yüzdesel Dağılımı')

plt.show()
```




- **Model Değerlendirmesi**: Doğruluk (Accuracy), Kesinlik (Precision), Duyarlılık (Recall) ve F1-Skoru metrikleri ile model performansı değerlendirilecektir.

```
accuracy_knn = accuracy_score(y_test, y_pred_knn)

precision_knn = precision_score(y_test, y_pred_knn)

recall_knn = recall_score(y_test, y_pred_knn)

f1_knn = f1_score(y_test, y_pred_knn)


print(f'Doğruluk (KNN): {accuracy_knn}')

print(f'Kesinlik (KNN): {precision_knn}')

print(f'Duyarlılık (KNN): {recall_knn}')

print(f'F1-Skoru (KNN): {f1_knn}')
```

```
Doğruluk (KNN): 0.75
Kesinlik (KNN): 0.7382198952879581
Duyarlılık (KNN): 1.0
F1-Skoru (KNN): 0.8493975903614458
```

## 5. Model Değerlendirmesi

### A. Metrikler ile Değerlendirme

- **Karışıklık Matrisi**: Her iki modelin karışıklık matrisi oluşturulacaktır.

```
from sklearn.metrics import confusion_matrix

print("Lojistik Regresyon - Karışıklık Matrisi")

confusion_matrix(y_test, y_pred)
```

```
array([[ 21,  38],
[ 12, 129]])
```

- True Negatives (0,0): 21
- False Positives (0,1): 38
- False Negatives (1,0): 12
- True Positives (1,1): 129

```
print("KNN - Karışıklık Matrisi")

confusion_matrix(y_test, y_pred_knn)
```

```
array([[  9,  50],
[  0, 141]])
```

- True Negatives (0,0): 9
- False Positives (0,1): 50
- False Negatives (1,0): 0
- True Positives (1,1): 141

- **Sınıflandırma Raporu**: Kesinlik, Duyarlılık, F1-Skoru metrikleri ile sınıflandırma raporu hazırlanacaktır.

```
from sklearn.metrics import classification_report
```

**Lojistik Regresyon için:**

```
print("Lojistik Regresyon Sınıflandırma Raporu:")

print(classification_report(y_test, y_pred))
```

```
Lojistik Regresyon Sınıflandırma Raporu:
```

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.64 | 0.36 | 0.46 | 59 |
| 1 | 0.77 | 0.91 | 0.84 | 141 |
| accuracy | | | 0.75 | 200 |
| macro avg | 0.70 | 0.64 | 0.65 | 200 |
| weighted avg | 0.73 | 0.75 | 0.73 | 200 |

**Lojistik Regresyon Sonuçları:**

Sınıf 0 (Negative) Performansı:

- Precision: 0.64

- Recall: 0.36 (Düşük duyarlılık, yani sınıf 0'ı ayırt etmekte zorlanıyor)

- F1-Score: 0.46

Sınıf 1 (Positive) Performansı:

- Precision: 0.77

- Recall: 0.91 (Sınıf 1 için yüksek duyarlılık)

- F1-Score: 0.84

Genel Performans:

- Accuracy: 75%

- Macro Avg F1-Score: 0.65 (Sınıflar arasında dengesiz bir performans var)

- Weighted Avg F1-Score: 0.73

**KNN için:**

```
print("KNN Sınıflandırma Raporu:")

print(classification_report(y_test, y_pred_knn))
```

```
KNN Sınıflandırma Raporu:
              precision    recall  f1-score   support

           0       1.00      0.15      0.26        59
           1       0.74      1.00      0.85       141

    accuracy                           0.75       200
   macro avg       0.87      0.58      0.56       200
weighted avg       0.82      0.75      0.68       200
```

**KNN Sonuçları:**

Sınıf 0 (Negative) Performansı:

- Precision: 1.00 (Hiç yanlış pozitif yok, ancak bu yanıltıcı olabilir)

- Recall: 0.15 (Çok düşük, çoğu sınıf 0 örneğini kaçırıyor)

- F1-Score: 0.26

Sınıf 1 (Positive) Performansı:

- Precision: 0.74

- Recall: 1.00 (Hiç sınıf 1 örneğini kaçırmıyor)

- F1-Score: 0.85

Genel Performans:

- Accuracy: 75%

- Macro Avg F1-Score: 0.56 (Sınıflar arasında büyük bir dengesizlik var)

- Weighted Avg F1-Score: 0.68

KNN, sınıf 1 için yüksek performans gösterirken, sınıf 0'ı neredeyse tamamen ihmal ediyor. Bu nedenle sınıflar arasındaki dengesizlik daha belirgin.

**B. Modellerin Karşılaştırılması**

- **Performans Karşılaştırması**: Lojistik Regresyon ve KNN sonuçları karşılaştırılacaktır. Hangi modelin daha iyi performans gösterdiği ve nedenleri tartışılacaktır.

**Doğruluk (Accuracy):**

- Her iki modelin doğruluk oranı %75 olarak eşit.

**Kesinlik (Precision):**

- Lojistik Regresyon'un macro kesinliği %70, KNN'nin ise %87.Ancak, KNN modelinin sınıf 0 için kesinliği %100, sınıf 1 için %74. Sınıf 0'daki yüksek kesinlik, sınıf 0'ı tamamen sınıflandırma eksikliğinden kaynaklanıyor olabilir.

**Duyarlılık (Recall):**

- Lojistik Regresyon'un macro duyarlılığı %64, KNN'ninki ise %58.
- KNN modelinde sınıf 1'in duyarlılığı yüksekken (%100), sınıf 0'da düşük bir değer (%15) var. Lojistik Regresyon'da ise sınıflar arasında daha dengeli.

**F1-Score:**
- Lojistik Regresyon'un F1-Skoru (macro): %65, KNN'nin ise %56

## İki Modelin Avantajları ve Sınırlılıkları

**Lojistik Regresyon (Logistic Regression)**

Lojistik regresyon, kredi risk değerlendirme problemlerinde genellikle başlangıç modeli olarak kullanılır. Bu model, sınıflandırma yaparken doğrusal bir karar sınırına dayanır.

### *Avantajları:*

1. **Dengeli Performans:**
    - Lojistik regresyon, sınıflar arasında dengeli bir performans sağlar. Özellikle raporda verilen metriklerden **precision** (kesinlik) ve **recall** (duyarlılık) değerleri sınıflar arasında aşırı sapma göstermiyor. Bu, her iki sınıfın (iyi ve kötü kredi riski) makul şekilde temsil edildiğini gösterir.

    - Örneğin:
        - Sınıf 1 için precision: %77, recall: %91
        - Sınıf 0 için precision: %64, recall: %36
    - Sınıf 0'da performans düşük olsa da, bu durum K-NN'ye kıyasla daha dengelidir.

2. **Yorumlanabilirlik:**

    - Lojistik regresyonun her özelliğe verdiği önem (coefficients), bu özelliklerin kredi riskine nasıl etki ettiğini anlamak için kullanılabilir. Bu da modeli daha kolay açıklanabilir hale getirir.

3. **Aşırı Uyum Riskinin Az Olması:**

    - Lojistik regresyonun basit yapısı, modelin karmaşıklığını artırmadığı için, özellikle küçük ve orta ölçekli veri setlerinde aşırı uyuma (overfitting) karşı dayanıklıdır.

4. **Hızlı Eğitim ve Tahmin:**

    - Model hızlı bir şekilde eğitilebilir ve yeni örnekler için hızlı tahminlerde bulunabilir.

## *Sınırlılıkları:*

1. **Doğrusal Karar Sınırları:**

    - Lojistik regresyon, değişkenler arasındaki ilişkiler doğrusal olmadığında performans kaybı yaşayabilir. Örneğin, kredi riskini belirleyen faktörler genelde karmaşık ve doğrusal olmayan bir yapı sergiler.

2. **Sınıf Dengesizliği:**

    - Veri setinde **iyi kredi riski** oranı %70 ve **kötü kredi riski** oranı %30. Bu dengesizlik, lojistik regresyonun küçük sınıfı (kötü kredi riski) öğrenmede yetersiz kalmasına neden olabilir. Bu durum sınıf 0 için düşük **recall** değerinde gözlemlenmiştir (%36).

## K-NN (K-Nearest Neighbors)

K-NN, örüntü tanıma ve sınıflandırma problemlerinde kullanılan esnek ve doğrusal olmayan bir modeldir. Bu model, tahmin yaparken eğitim veri noktalarına olan mesafeyi kullanır.

### *Avantajları:*

1. **Doğrusal Olmayan Karar Sınırları:**

    - K-NN, değişkenler arasındaki doğrusal olmayan ilişkileri modelleyebilir. Örneğin, kredi miktarı ve kredi süresi arasındaki karmaşık ilişkileri daha iyi yakalayabilir.

2. **Sınıf 1 İçin Yüksek Performans:**

    - K-NN, pozitif sınıfı (iyi kredi riski) tespit etme konusunda oldukça başarılıdır:

    Sınıf 1 için recall: %100

    Bu, modelin iyi kredi riskine sahip tüm bireyleri doğru bir şekilde tanımladığını gösterir. Özellikle kötü kredi riskinin yanlış sınıflandırılmasının ciddi maliyetlere yol açtığı durumlarda avantaj sağlar.

3. **Hiperparametrelerle Özelleştirilebilirlik:**

    - k değeri gibi parametreler optimize edilerek performans artırılabilir. Bu, modele problem özelinde esneklik kazandırır.

### *Sınırlılıkları:*

1. **Sınıf Dengesizliği ve Düşük Performans:**

    - Sınıf 0 için recall sadece %15'tir. Yani, kötü kredi riskine sahip bireylerin çoğunu doğru bir şekilde sınıflandıramaz. Bu, K-NN'nin veri dengesizliği nedeniyle düşük performans göstermesine neden olur.

    - Örneğin:

      Sınıf 0 için precision: %100 (Ancak, neredeyse tüm sınıf 0 örnekleri yanlış pozitif olarak sınıflandırıldığı için bu yanıltıcıdır.)

2. **Yüksek Hesaplama Maliyeti:**

- Eğitim sırasında hızlı olsa da, tahmin aşamasında K-NN modeli tüm veri seti üzerindeki mesafeleri hesaplamak zorundadır. Büyük veri setlerinde bu maliyet hızla artabilir.

3. **Hassasiyet:**

   - Hiperparametrelerin (örneğin kk değeri) doğru seçilmesi performans için kritik öneme sahiptir. Yanlış seçim, modelin performansını önemli ölçüde düşürebilir

# Hangi Model Daha İyi?

**Dengeli Performans Gerekiyorsa:**

- Lojistik Regresyon, tüm metriklerde dengeli bir sonuç verdiği için genel olarak daha uygun görünüyor. Duyarlılık ve kesinlik oranlarının makul bir şekilde eşit olması, sınıflar arası dengeyi daha iyi sağlıyor.

**Sınıf 1'in Kritik Olduğu Durumlarda:**

- Eğer sınıf 1'in (örneğin yüksek riskli bir durumun) tüm pozitif örneklerinin doğru tespit edilmesi gerekiyorsa, KNN'nin sınıf 1 için duyarlılığı (%100) büyük bir avantaj sunabilir.

**Sınıf 0 İçin Yüksek Doğruluk Gerekiyorsa:**

- KNN modeli, sınıf 0 için %100 kesinlik sağlayarak bu sınıfa odaklanan senaryolarda tercih edilebilir.

# Sonuç:

Lojistik Regresyon, genellikle daha dengeli bir performans sağladığı için çoğu senaryo için daha iyi bir seçimdir. Ancak, hangi modelin daha iyi olduğu problemin gereksinimlerine ve hangi hataların daha kritik olduğuna bağlıdır. Örneğin:

- Sınıf 1'i yanlış sınıflandırmanın maliyeti yüksekse → KNN tercih edilebilir.
- Dengeli bir performans istiyorsanız → Lojistik Regresyon daha uygundur.

## Kullanılan Araçlar, Teknolojiler ve Kütüphaneler

Bu proje kapsamında kredi risk değerlendirmesi için veri işleme, modelleme ve analiz süreçlerinde çeşitli araçlar, teknolojiler ve kütüphaneler kullanılmıştır. Bu bileşenler, projenin her aşamasında bir arada kullanılarak etkin ve kapsamlı bir analiz ortamı sağlamıştır.

*1. Python*

- Projenin tamamında kullanılan ana programlama dili olarak, esnekliği ve geniş veri bilimi ekosistemi nedeniyle tercih edilmiştir.
- Python, veri manipülasyonu, modelleme, görselleştirme ve makine öğrenimi süreçlerinde güçlü bir altyapı sunar.

*2. Jupyter Notebook*

- Veri analizinin interaktif bir şekilde gerçekleştirilmesi için kullanılmıştır.
- Proje adımları (veri yükleme, analiz, görselleştirme, modelleme) burada yürütülmüş ve her adımda kolayca geri dönüş yapılabilmiştir.

*3. Sklearn (scikit-learn)*

- **Makine öğrenimi modelleri ve değerlendirme metrikleri** için kullanılan temel kütüphane olmuştur.
- Kullanılan modeller:
  - **LogisticRegression**: Lojistik regresyon modeli.
  - **KNeighborsClassifier**: K-NN algoritması.
- Değerlendirme ve doğrulama araçları:
  - `accuracy_score`, `precision_score`, `recall_score`, `f1_score`: Model performans metrikleri.
  - `train_test_split`: Veri setini eğitim ve test olarak bölmek için.
  - `cross_val_score`: K-NN modelinde en iyi KK değerini belirlemek için çapraz doğrulama yöntemiyle kullanılmıştır.

*4. Matplotlib ve Seaborn*

- Veri görselleştirme aşamalarında iki önemli kütüphane kullanılmıştır:
  - **Matplotlib**: Histogramlar, kutu grafikleri ve pasta grafikleri gibi temel görselleştirmeler için.
  - **Seaborn**: Daha gelişmiş görselleştirme ihtiyaçları (korelasyon haritaları ve kategorik değişken analizleri) için.

*5. Pandas ve NumPy*

- Veri manipülasyonu ve temel istatistiksel analizler için iki güçlü kütüphane kullanılmıştır:
  - **Pandas**: Veri yükleme, sütun manipülasyonu, eksik değer analizi ve temel istatistiksel özetler.
  - **NumPy**: Hızlı ve verimli sayısal işlemler için.

*6. Veri Ön İşleme Araçları*

- **Eksik Değer Doldurma:**
  - Eksik kategorik veriler doldurulmuş, sayısal değişkenler için medyan kullanılmıştır.

- **Özellik Ölçekleme (StandardScaler):**
  - Yaş, kredi tutarı ve kredi süresi gibi özellikler ölçeklendirilerek modellerin daha doğru çalışması sağlanmıştır.
- **OneHotEncoder:**
  - Kategorik değişkenlerin sayısal değerlere dönüştürülmesi için kullanılmıştır.

*7. Aykırı Değer Tespiti*

- **Interquartile Range (IQR):**
  - Kredi tutarı, yaş ve kredi süresi gibi değişkenlerde potansiyel aykırı değerleri belirlemek için bu yöntem uygulanmıştır.

*8.Görselleştirme Araçları*

- Grafiklerin oluşturulması ve verilerin dağılımlarının analizi için aşağıdaki araçlar kullanılmıştır:
  - Histogramlar ve kutu grafikleri ile sayısal değişkenler.
  - Korelasyon haritaları ile değişkenler arasındaki ilişkilerin görselleştirilmesi.
  - Pasta grafikleri ile sınıflandırma sonuçlarının yüzdesel dağılımları.

## Projenin Sunumu:

https://www.youtube.com/watch?v=RBfNydtSah8

Hazırlayan:

Melike Su Koçyiğit