**Veri Madenciliği- Proje Raporu**

**1. Proje Amacı**

1912 yılında batan Titanic gemisindeki yolcuların hayatta kalıp kalmadığını makine ögrenmesi modelleri ile tahmin etmeyi amaçladık. Model çıktıları; kurtarma senaryoları, güvenlik protokollerinin analizi ve akademik amaçlı sınıflandırma örnekleri için değerli görüşler sağlar.

**2. Veri Seti Tanımı**

* **Kaynak dosya:** Titanic-Dataset.csv
* **Veri Sayısı :** 891 yolcu kaydı

o Bağımlı Değişkenler: Kategorik(Sex, Embarked, Pclass (1-3), SibSp, Parch) , Sayısal( Age, Fare)  
o Hedef: Survived (1 = Hayatta, 0 = Öldü)

**3. Veri Ön İşleme**

Age sütunundaki boş gözlemler **medyan** (≈ 28 yaş) ile, Embarked sütunundaki boş gözlemler en sık liman olan **'S'** değeri ile dolduruldu. Sex ve Embarked degişkenleri **One-Hot Encoding** ile numerik forma dönüştürüldü.

Survived hedef değişken olarak ayrıldı; geri kalan sütunlar **X** veri matrisini oluşturdu.

**4. Modelleme Sonuçları**

Lojistik Regresyon:

Doğruluk = %81, ROC AUC = 0.88

Hızlı ve yorumlanabilir; cinsiyet ve bilet sınıfı katsayıları öne çıktı.

Karar Ağacı:

Doğruluk = %79, ROC AUC = 0.88

Basit kurallar üretti; derinlik sınırı aşırı uyumu engelledi.

Random Forest :

Doğruluk = %82, ROC AUC = 0.89

En yüksek genel performans; değişken önem sıralaması tutarlı.

**4.1 Confusion Matrix Örnekleri**

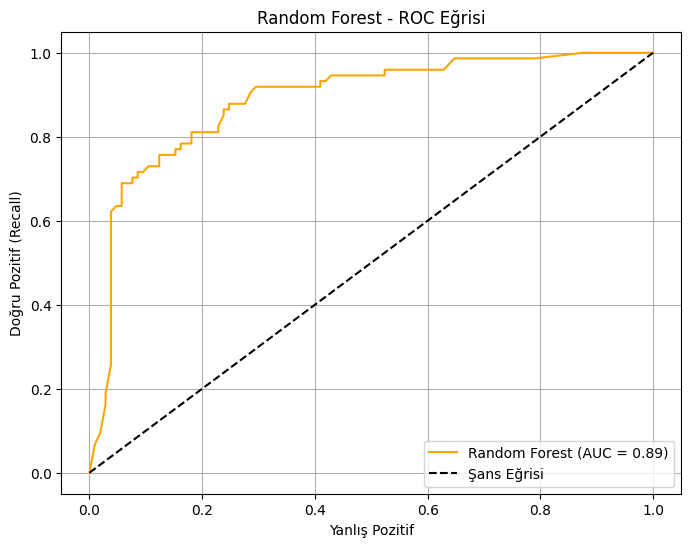
Aşağıdaki çizimler proje sırasında elde edildi. Örneğin Lojistik Regresyon için tipik bir matris:

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Yapay zeka tarafından oluşturulmuş içerik yanlış olabilir.

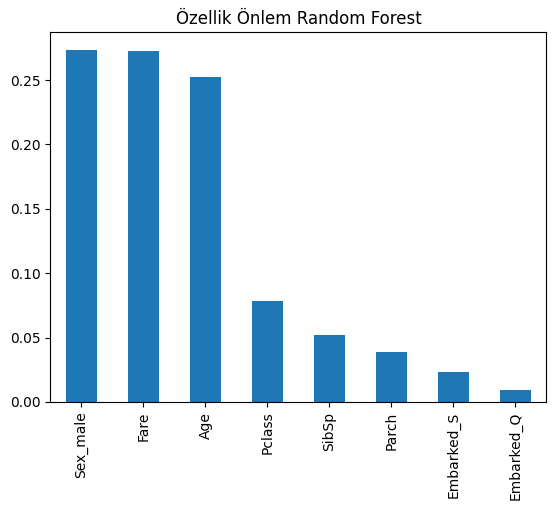
**4.2 ROC Eğrileri**

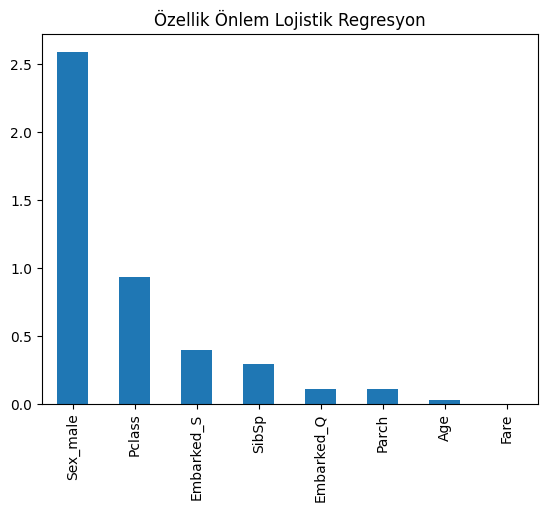
Her model için ROC eğrisi çizilerek **AUC** değerleri karşılaştırıldı. Random Forest eğrisi, rastgele tahmin çizgisine en uzak kalan alanı (≈ 0.89) kapsayarak en iyi ayrıştırmayı sağladı.



**4.3 Özellik Önem Analizi**

Random Forest ve Karar Ağacı modellerine göre en etkili degişkenler sırasıyla:





**5. Sonuç̧**

1. **Performans:** Random Forest test doğruluğu %80+ seviyeye ulaştı; lojistik regresyon da rekabetçidir.
2. **Model Karmaşıklığı:** İhtiyaç̧ duyulan açıklama seviyesine göre model seçimi yapılmalıdır.

KODLAR(.ipynb)

df = pd.read\_csv('Titanic-Dataset.csv')

pd.DataFrame(df)

df.describe()

df.shape

df.info()

df1 = df[['Survived','Pclass','Sex','Age','SibSp','Parch','Fare','Embarked']]

df1.isnull().sum()

df1['Age'].fillna(df1['Age'].median(),inplace=True) #Boş yerleri ortalama ile dolduruyoruz

df1['Embarked'].fillna(df1['Embarked'].mode()[0],inplace=True) #Boş yerleri en çok tekrar eden veri ile dolduruyoruz

df1 = pd.get\_dummies(df1,columns=['Sex','Embarked'],drop\_first=True) # Nümerik Verilere Çevirme

df1.head()

df1.info()

X = df1.drop('Survived',axis = 1) # Hedef Değişkenleri Silince Bağımsız Değişkenler kalır

y = df1['Survived'] # Hedef Değişken

from sklearn.model\_selection import train\_test\_split # Modeli Test ve Eğitim Olarak Bölme

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

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import confusion\_matrix , accuracy\_score , classification\_report

log\_model = LogisticRegression(max\_iter=1000)

log\_model.fit(X\_train , y\_train)

Tahminde kaldık

y\_tahmin\_log = log\_model.predict(X\_test)

accuracy\_score(y\_test,y\_tahmin\_log)

classification\_report(y\_test,y\_tahmin\_log)

from sklearn.tree import DecisionTreeClassifier

agac\_model = DecisionTreeClassifier(max\_depth=4,random\_state=42)

agac\_model.fit(X\_train,y\_train)

y\_tahmin\_tree = agac\_model.predict(X\_test)

accuracy\_score(y\_test,y\_tahmin\_tree)

classification\_report(y\_test,y\_tahmin\_tree)

from sklearn.metrics import confusion\_matrix, ConfusionMatrixDisplay

cm\_hsp = confusion\_matrix(y\_test,y\_tahmin\_log)

import matplotlib.pyplot as plt

gorsel = ConfusionMatrixDisplay(confusion\_matrix=cm\_hsp,display\_labels=['Öldü','Hayatta'])

gorsel.plot(cmap=plt.cm.Blues)

plt.ylabel('Gerçek Etiket')

plt.xlabel('Tahmini Etiket')

plt.title('Lojistik Regresyon')

from sklearn.ensemble import RandomForestClassifier

rfc\_model = RandomForestClassifier(n\_estimators=100,random\_state=42) # 100 ağaçlık orman

rfc\_model.fit(X\_train,y\_train)

y\_tahmin\_rfc = rfc\_model.predict(X\_test)

accuracy\_score(y\_test,y\_tahmin\_rfc)

classification\_report(y\_test,y\_tahmin\_rfc)

cm\_rfc = confusion\_matrix(y\_test, y\_tahmin\_rfc)

gorsel\_rfc = ConfusionMatrixDisplay(confusion\_matrix=cm\_rfc, display\_labels=["Öldü", "Hayatta"])

gorsel\_rfc.plot(cmap=plt.cm.Oranges)

plt.title("Random Forest - Confusion Matrix")

plt.ylabel('Gerçek Etiket')

plt.xlabel('Tahmini Etiket')

plt.show()

from sklearn.metrics import roc\_auc\_score , roc\_curve

roc\_tahmin = rfc\_model.predict\_proba(X\_test)[ : , 1]

fpr\_rfc , tpr\_rfc , \_ = roc\_curve(y\_test,roc\_tahmin)

auc\_rfc = roc\_auc\_score(y\_test,roc\_tahmin)

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

plt.plot(fpr\_rfc, tpr\_rfc, label=f'Random Forest (AUC = {auc\_rfc:.2f})', color='orange')

plt.plot([0, 1], [0, 1], 'k--', label='Şans Eğrisi')

plt.xlabel('Yanlış Pozitif')

plt.ylabel('Doğru Pozitif (Recall)')

plt.title('Random Forest - ROC Eğrisi')

plt.legend(loc='lower right')

plt.grid(True)

plt.show()

ozellik\_onlem = pd.Series(rfc\_model.feature\_importances\_,index=X.columns)

gorsel = ozellik\_onlem.sort\_values(ascending=False)

gorsel.plot(kind = 'bar',title='Özellik Önlem Random Forest')

roc\_tahmin\_log = log\_model.predict\_proba(X\_test)[ : , 1]

fpr\_log , tpr\_log , \_ = roc\_curve(y\_test , roc\_tahmin\_log)

auc\_log = roc\_auc\_score(y\_test,roc\_tahmin\_log)

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

plt.plot(fpr\_log, tpr\_log, label=f'Lojistik Regresyon (AUC = {auc\_log:.2f})', color='red')

plt.plot([0, 1], [0, 1], 'k--', label='Şans Eğrisi')

plt.xlabel('Yanlış Pozitif')

plt.ylabel('Doğru Pozitif (Recall)')

plt.title('Lojistik Regresyon - ROC Eğrisi')

plt.legend(loc='lower right')

plt.grid(True)

plt.show()

ozellik\_onlem\_log = pd.Series(abs(log\_model.coef\_[0]) , index = X.columns)

gorsel\_log = ozellik\_onlem\_log.sort\_values(ascending=False)

gorsel\_log.plot(kind = 'bar',title='Özellik Önlem Lojistik Regresyon')

cm\_agac = confusion\_matrix(y\_test , y\_tahmin\_tree)

gorsel\_rfc = ConfusionMatrixDisplay(confusion\_matrix=cm\_rfc, display\_labels=["Öldü", "Hayatta"])

gorsel\_rfc.plot(cmap=plt.cm.Oranges)

plt.title("Karar Ağaç - Confusion Matrix")

plt.ylabel('Gerçek Etiket')

plt.xlabel('Tahmini Etiket')

plt.show()

roc\_tahmin\_agac = agac\_model.predict\_proba(X\_test)[:,1]

fpr\_tree , tpr\_tree , \_ = roc\_curve(y\_test,roc\_tahmin\_agac)

auc\_tree = roc\_auc\_score(y\_test,roc\_tahmin\_agac)

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

plt.plot(fpr\_log, tpr\_log, label=f'Karar Ağaçları (AUC = {auc\_log:.2f})', color='green')

plt.plot([0, 1], [0, 1], 'k--', label='Şans Eğrisi')

plt.xlabel('Yanlış Pozitif')

plt.ylabel('Doğru Pozitif (Recall)')

plt.title('Karar Ağaçları - ROC Eğrisi')

plt.legend(loc='lower right')

plt.grid(True)

plt.show()

ozellik\_onlem\_tree = pd.Series(agac\_model.feature\_importances\_ , index=X.columns)

gorsel\_tree = ozellik\_onlem\_tree.sort\_values(ascending=False)

gorsel\_tree.plot(kind = 'bar',title='Özellik Önlem Karar Ağaçları')

Eren Uçar 20224029028

Nişantaşı Üniversitesi Yönetim Bilişim Sistemleri