İçindekiler

1) Projenin Tanımı

Projemizde WineQT.csv dosyasındaki veri setinden yararlandık. Öncelikle şeker hastaları gibi sağlık konusunda sıkıntı çeken hastaların hangi şarabı tüketebileceklerini inceleyecek ve cevap verebilecek bir program yazmak istedik ancak verilerimizi işledikten sonra regresyon tablosunu inceleyince şeker oranı ile ilişkisi yüksek bir değer ya da diyabet hastalarını tetikleyebilecek başka bir değişken olmadığı için sadece şeker oranını baz alarak bir çalışma yapacaktık ya da şeker oranı ile ilişkisi düşük başka bir değer kullanıp sonucun gerçekçiliğini düşürecektik bu yüzden bundan vazgeçip programa belli başlı niteliklerin(ph,alkol ve sülfat oranı gibi) verildiği zaman şarabın yaklaşık kalitesini tahmin eden bir program yazdık.

2) Araştırma (Ön çalışma)

Araştırmamızı data setini aldığımız yerden yani Kaggle'dan yaptık websitesinde data setinin yanı sıra diğer insanların yapmış olduğu projelere bakarak bu data setini işleyerek ne gibi sonuçlara erişebileceğimize ve erişirken hangi kütüphanelerden yardım alabileceğimiz gibi konulara baktık.

3) Kullanılan Ortam, Yöntem ve Kütüphaneler

Pythonda kodlamımızı yaptık kullandığımız başlıca kütüphaneler
-pandas -sklearn -seaborn -matplotlib -altair

4) Önerilen (Geliştirilen / Kullanılan) Yöntem

Confussionmatrixdt.py

```
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Veriyi yükle
df = pd.read_csv(r'C:\Users\yavuz\PycharmProjects\AIM_P2\WineQT.csv')

# Kalite eşiğini belirle ve quality_label sütununu oluştur
quality_threshold = 6
df['quality_label'] = ['kaliteli' if quality >= quality_threshold else 'kalitesiz' for quality in
df['quality']]

# 'quality' ve 'Id' sütunlarını düşür
df.drop(['quality', 'Id'], axis=1, inplace=True)

# Özellik matrisini (X) ve hedef değişkeni (y) oluştur
X = df.drop('quality_label', axis=1)
y = df['quality_label']

# Veriyi eğitim ve test kümelerine ayır
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Karar Ağacı sınıflandırıcısı modelini oluştur ve eğit
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)

# Test kümesi üzerinde tahminler yap
y_pred = model.predict(X_test)

# Karışıklık matrisini hesapla
cm = confusion_matrix(y_test, y_pred)

# Karışıklık matrisini görselleştir
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Kalitesiz', 'Kaliteli'],
yticklabels=['Kalitesiz', 'Kaliteli'])
plt.xlabel('Tahmin Edilen')
plt.ylabel('Gerçek')
plt.title('Karışıklık Matrisi')
plt.show()

# Modelin doğruluk skorunu hesapla ve yazdır
```

```
accuracy = accuracy_score(y_test, y_pred)
print(f'Model Accuracy: {accuracy:.2f}')
```

Kfold.py

```
import pandas as pd
from sklearn.model_selection import train_test_split, KFold
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
import altair as alt

# Veriyi yükle
df = pd.read_csv('../WineQT.csv')

# Kalite eşiğini belirle ve quality_label sütununu oluştur
quality_threshold = 6
df['quality_label'] = ['kaliteli' if quality >= quality_threshold else 'kalitesiz' for quality in
df['quality']]

# 'quality' ve 'Id' sütunlarını düşür
df.drop(['quality', 'Id'], axis=1, inplace=True)

# Özellik matrisini (X) ve hedef değişkeni (y) oluştur
X = df.drop('quality_label', axis=1)
y = df['quality_label']

# K-fold çapraz doğrulama için KFold nesnesi oluşturun (k=5)
kf = KFold(n_splits=5, shuffle=True, random_state=42)

accuracies = []
conf_matrices = []

# Her bir fold için işlemleri tekrarla
for fold, (train_index, val_index) in enumerate(kf.split(X)):
    X_train, X_val = X.iloc[train_index], X.iloc[val_index]
    y_train, y_val = y.iloc[train_index], y.iloc[val_index]

    # Karar Ağacı sınıflandırıcısı modelini oluştur ve eğit
    model = DecisionTreeClassifier(random_state=42)
    model.fit(X_train, y_train)

    # Doğrulama kümesi üzerinde tahminler yap
    y_pred = model.predict(X_val)
```

```
    # Doğruluk skorunu hesapla ve kaydet
    accuracy = accuracy_score(y_val, y_pred)
    accuracies.append(accuracy)

    # Karışıklık matrisini hesapla ve kaydet
    cm = confusion_matrix(y_val, y_pred)
    conf_matrices.append(cm)

    print(f'Fold {fold + 1}: Accuracy = {accuracy:.2f}')

# Ortalama doğruluk skorunu ve sınıflandırma raporunu yazdır
mean_accuracy = sum(accuracies) / len(accuracies)
print(f'\nMean Accuracy: {mean_accuracy:.2f}')

# Tüm fold'lar için ortalama karışıklık matrisini hesapla
mean_cm = sum(conf_matrices) / len(conf_matrices)

# Ortalama karışıklık matrisini görselleştir
plt.figure(figsize=(8, 6))
sns.heatmap(mean_cm, annot=True, fmt='.2f', cmap='Blues',
            xticklabels=['Kalitesiz', 'Kaliteli'], yticklabels=['Kalitesiz', 'Kaliteli'])
plt.xlabel('Tahmin Edilen')
plt.ylabel('Gerçek')
plt.title('Ortalama Karışıklık Matrisi (K-Fold Çapraz Doğrulama)')
plt.show()

# Her bir fold için accuracy değerlerini görselleştir
fold_names = [f"Fold {i + 1}" for i in range(kf.get_n_splits())]
accuracy_data = pd.DataFrame({'Fold': fold_names, 'Accuracy': accuracies})

chart = alt.Chart(accuracy_data).mark_bar().encode(
    x=alt.X('Fold:N', axis=alt.Axis(title='Fold')),
    y=alt.Y('Accuracy:Q', axis=alt.Axis(title='Accuracy')),
    tooltip=['Fold', 'Accuracy']
).properties(
    title='Accuracy for Each Fold'
).interactive()

chart.save('accuracy_per_fold_bar_chart.json')
```

Sınıflandırma.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
import altair as alt

# Veriyi yükle
df = pd.read_csv(r'C:\Users\yavuz\PycharmProjects\AIM_P2\WineQT.csv')

# İlk 5 satırı yazdır
print(df.head().to_markdown(index=False, numalign="left", stralign="left"))

# Sütun adlarını ve veri tiplerini yazdır
print(df.info())

# Kalite eşiğini belirle ve quality_label sütununu oluştur
quality_threshold = 6
df['quality_label'] = ['kaliteli' if quality >= quality_threshold else 'kalitesiz' for quality in
df['quality']]

# 'quality' ve 'Id' sütunlarını düşür
df.drop(['quality', 'Id'], axis=1, inplace=True)

# Özellik matrisini (X) ve hedef değişkeni (y) oluştur
X = df.drop('quality_label', axis=1)
y = df['quality_label']

# Veriyi eğitim ve test kümelerine ayır
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Karar Ağacı sınıflandırıcısı modelini oluştur ve eğit
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)

# Test kümesi üzerinde tahminler yap
y_pred = model.predict(X_test)

# Modelin doğruluğunu hesapla ve yazdır
accuracy = accuracy_score(y_test, y_pred)
print(f'Model Accuracy: {accuracy:.2f}')

# Sınıflandırma raporunu oluştur ve yazdır
report = classification_report(y_test, y_pred)
print("Classification Report:\n", report)

# Kalite metriklerini hesapla ve yazdır
quality_counts = df['quality_label'].value_counts()
```

```
quality_percentage = (quality_counts['kaliteli'] / quality_counts.sum()) * 100
print("\nQuality Metrics:")
print(quality_counts.rename_axis('Quality').rename('Total Wines').to_markdown(numalign="left",
stralign="left"))
print(f'Percentage of High Quality Wines: {quality_percentage:.2f}%')

# Feature İmportances
feature_importances = pd.DataFrame({'Feature': X.columns, 'Importance':
model.feature_importances_})

# Feature İmportances Bar Chart
chart = alt.Chart(feature_importances).mark_bar().encode(
   x=alt.X('Importance:Q', axis=alt.Axis(title='Importance')),
   y=alt.Y('Feature:N', sort='-x'),
   tooltip=['Feature', 'Importance']
).properties(title='Feature Importances').interactive()

chart.save('feature_importances_bar_chart_dtree.json')
```

Tahmindt.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report

# Veriyi yükle
df = pd.read_csv(r'C:\Users\yavuz\PycharmProjects\AIM_P2\WineQT.csv')

# Kalite eşiğini belirle ve quality_label sütununu oluştur
quality_threshold = 6
df['quality_label'] = ['kaliteli' if quality >= quality_threshold else 'kalitesiz' for quality in
df['quality']]

# 'quality' ve 'Id' sütunlarını düşür
df.drop(['quality', 'Id'], axis=1, inplace=True)

# Özellik matrisini (X) ve hedef değişkeni (y) oluştur
X = df.drop('quality_label', axis=1)
y = df['quality_label']

# Veriyi eğitim ve test kümelerine ayır
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Karar Ağacı sınıflandırıcısı modelini oluştur ve eğit
model = DecisionTreeClassifier(random_state=42)
model.fit(X_train, y_train)

# Test kümesi üzerinde tahminler yap
y_pred = model.predict(X_test)

# Modelin doğruluğunu hesapla ve yazdır
accuracy = accuracy_score(y_test, y_pred)
print(f'Model Accuracy: {accuracy:.2f}')

# Sınıflandırma raporunu oluştur ve yazdır
report = classification_report(y_test, y_pred)
print("Classification Report:\n", report)

# Kalite metriklerini hesapla ve yazdır
quality_counts = df['quality_label'].value_counts()
quality_percentage = (quality_counts['kaliteli'] / quality_counts.sum()) * 100
print("\nQuality Metrics:")
print(quality_counts.rename_axis('Quality').rename('Total Wines').to_markdown(numalign="left",
stralign="left"))
print(f'Percentage of High Quality Wines: {quality_percentage:.2f}%')

# Kullanıcının şarap bilgileri (örnek değerler)
user_wine = pd.DataFrame(0, index=[0], columns=X_train.columns)
user_wine['pH'] = 3.5
user_wine['alcohol'] = 10.5
user_wine['sulphates'] = 0.6

# Girilen değerleri kullanarak modelin tahminini yap
prediction = model.predict(user_wine)

# Tahmin edilen kaliteyi ekrana yazdır
print("Tahmin edilen şarap kalitesi:", prediction[0])
```

Randomforestconfmatrix.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
import altair as alt
```

```
import numpy as np

# Tüm satır ve sütunları göstermek için pandas ayarları
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

# CSV dosyasını DataFrame olarak oku
df = pd.read_csv(r'C:\Users\yavuz\PycharmProjects\AIM_P2\WineQT.csv')

# İlk 5 satırı göster
print(df.head().to_markdown(index=False, numalign="left", stralign="left"))

# Sütun adlarını ve veri türlerini yazdır
print(df.info())

# Eşik 6'ya göre `quality_label` sütunu oluştur
df['quality_label'] = df['quality'].apply(lambda x: 'kaliteli' if x >= 6 else 'kalitesiz')

# `quality` ve `Id` sütunlarını düşür
df.drop(['quality', 'Id'], axis=1, inplace=True)

# Özellik matrisi (X) ve hedef vektörü (y) oluştur
X = df.drop('quality_label', axis=1)
y = df['quality_label']

# Veriyi eğitim ve test kümelerine ayır
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Random Forest modelini oluştur ve eğit
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Test kümesi üzerinde tahmin yap
y_pred = model.predict(X_test)

# Karışıklık matrisini hesapla
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", cm)

# Karışıklık matrisi için DataFrame oluştur
cm_df = pd.DataFrame(cm, index=['Actual Kalitesiz', 'Actual Kaliteli'], columns=['Predicted
Kalitesiz', 'Predicted Kaliteli'])

# Isı haritası oluştur
heatmap = alt.Chart(cm_df.stack().reset_index().rename(columns={'level_0': 'Actual', 'level_1':
```

```
'Predicted', 0: 'Count'})).mark_rect().encode(
    x=alt.X('Predicted:O', axis=alt.Axis(title='Predicted')),
    y=alt.Y('Actual:O', axis=alt.Axis(title='Actual')),
    color=alt.Color('Count:Q', scale=alt.Scale(scheme='blues')),
    tooltip=['Actual', 'Predicted', 'Count']
).properties(
    title='Confusion Matrix for Random Forest Model'
).interactive()

# Isı haritasına metin etiketleri ekle
text = heatmap.mark_text(baseline='middle').encode(
    text='Count:Q',
    color=alt.condition(alt.datum.Count > np.mean(cm), alt.value('white'), alt.value('black'))
)

# Isı haritası ve metin katmanlarını birleştir
chart = heatmap + text

# Grafiği kaydet
chart.save('confusion_matrix_heatmap_random_forest.json')
```

randomforestsınıflandırma.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import altair as alt

# Veriyi yükle
df = pd.read_csv(r'C:\Users\yavuz\PycharmProjects\AIM_P2\WineQT.csv')

# Kalite eşiğini belirle ve quality_label sütununu oluştur
quality_threshold = 6
df['quality_label'] = ['kaliteli' if quality >= quality_threshold else 'kalitesiz' for quality in
df['quality']]

# 'Id' sütununu düşür
df.drop(['Id'], axis=1, inplace=True)

# Özellik matrisini (X) ve hedef değişkeni (y) oluştur
X = df.drop('quality_label', axis=1)
y = df['quality_label']
```

```
# Veriyi eğitim ve test kümelerine ayır
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)

# Random Forest sınıflandırıcısı modelini oluştur ve eğit
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Test kümesi üzerinde tahminler yap
y_pred = model.predict(X_test)

# Modelin doğruluğunu hesapla ve yazdır
accuracy = accuracy_score(y_test, y_pred)
print(f'Model Accuracy: {accuracy:.2f}')

# Sınıflandırma raporunu oluştur ve yazdır
report = classification_report(y_test, y_pred)
print("Classification Report:\n", report)

# Karışıklık matrisini hesapla ve ısı haritası olarak görselleştir
cm = confusion_matrix(y_test, y_pred)

# Karışıklık Matrisi Heatmap
cm_df = pd.DataFrame(cm, index=['Kalitesiz (Gerçek)', 'Kaliteli (Gerçek)'], columns=['Kalitesiz
(Tahmin)', 'Kaliteli (Tahmin)'])
heatmap_chart = alt.Chart(cm_df.reset_index().melt(id_vars='index')).mark_rect().encode(
   x=alt.X('variable:O', axis=alt.Axis(title='Tahmin')),
   y=alt.Y('index:O', axis=alt.Axis(title='Gerçek')),
   color=alt.Color('value:Q', scale=alt.Scale(scheme='blues')),
   tooltip=['value:Q']
).properties(
   title='Karışıklık Matrisi (Random Forest)'
).interactive()

# Heatmap üzerindeki sayıları ekle
text = heatmap_chart.mark_text(baseline='middle').encode(
   text='value:Q',
   color=alt.condition(alt.datum.value > 100, alt.value('white'), alt.value('black'))
)
chart = heatmap_chart + text
chart.save('confusion_matrix_heatmap_random_forest.json')

# Kaliteli ve kalitesiz şarapların sayısını ve yüzdesini hesapla ve yazdır
quality_counts = df['quality_label'].value_counts()
quality_percentage = (quality_counts['kaliteli'] / quality_counts.sum()) * 100
```

```
print("\nQuality Metrics:")
print(quality_counts.rename_axis('Quality').rename('Total Wines').to_markdown(numalign="left",
stralign="left"))
print(f'Percentage of High Quality Wines: {quality_percentage:.2f}%')

# Feature İmportances
feature_importances = pd.DataFrame({'Feature': X.columns, 'Importance':
model.feature_importances_})

# Feature İmportances Bar Chart
chart = alt.Chart(feature_importances).mark_bar().encode(
   x=alt.X('Importance:Q', axis=alt.Axis(title='Importance')),
   y=alt.Y('Feature:N', sort='-x'),
   tooltip=['Feature', 'Importance']
).properties(title='Feature Importances (Random Forest)').interactive()

chart.save('feature_importances_bar_chart_random_forest.json')
```

Randomforesttahmin.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Veriyi yükle
df = pd.read_csv(r'C:\Users\yavuz\PycharmProjects\AIM_P2\WineQT.csv')

# Kalite eşiğini belirle ve quality_label sütununu oluştur
quality_threshold = 6
df['quality_label'] = ['kaliteli' if quality >= quality_threshold else 'kalitesiz' for quality in
df['quality']]

# Özellik matrisini (X) ve hedef değişkeni (y) oluştur
X = df[['pH', 'alcohol', 'sulphates']]
y = df['quality_label']

# Veriyi eğitim ve test kümelerine ayır
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Random Forest modelini oluştur ve eğit
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

```
# Modelin doğruluğunu hesapla ve yazdır
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(f'Model Accuracy: {accuracy:.2f}')

# Kullanıcının şarap bilgileri (örnek değerler)
user_wine = pd.DataFrame([[3, 10.5, 0.6]], columns=['pH', 'alcohol', 'sulphates'])

# Tahmin yap
prediction = model.predict(user_wine)

# Tahmin edilen kaliteyi ekrana yazdır
print("Tahmin edilen şarap kalitesi:", prediction[0])
```

Clustering.py

```
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt

# Kmeans kümeleme

# Veri kümesini yükle
file_path = 'WineQT.csv'
data = pd.read_csv(file_path)

# Id sütununu düşür (modelleme için gereksiz)
data = data.drop(columns=['Id'])

# Özellikleri ölçeklendirme (Standardizasyon)
scaler = StandardScaler()
scaled_data = scaler.fit_transform(data)

# K-means kümeleme
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(scaled_data)

# Küme etiketlerini veri kümesine ekle
data['Cluster'] = kmeans.labels_

# Küme merkezlerini görselleştirme
centers = scaler.inverse_transform(kmeans.cluster_centers_)
```

```
center_df = pd.DataFrame(centers, columns=data.columns[:-1])
center_df['Cluster'] = range(3)

# Sonuçları görselleştirme
plt.figure(figsize=(10, 6))
plt.scatter(data['alcohol'], data['quality'], c=data['Cluster'], cmap='viridis')
plt.scatter(center_df['alcohol'], center_df['quality'], c='red', marker='X', s=200)
plt.xlabel('Alcohol')
plt.ylabel('Quality')
plt.title('Şarap Verileri Kümeleme Analizi')
plt.show()
```

Correlationmatrix.py

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt



pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)

# CSV dosyasını oku
veri = pd.read_csv('WineQT.csv')

# İlk 5 satırı göster
print("Veri setinden ilk 5 satır:")
print(veri.head().to_markdown(index=False, numalign="left", stralign="left"))

# Sütun adlarını ve veri tiplerini yazdır
print("\nSütun Adları ve Veri Tipleri:")
print(veri.info())

# 'Id' sütunu hariç korelasyon matrisini hesapla
korelasyon_matrisi = veri.drop(columns=['Id']).corr().round(3)

print("\nKorelasyon Matrisi:")
print(korelasyon_matrisi.to_markdown(numalign="left", stralign="left"))

# 'quality' ile diğer değişkenlerin korelasyonlarını (mutlak değer) bul ve sırala
kalite_korelasyonlari =
korelasyon_matrisi['quality'].drop('quality').abs().sort_values(ascending=False)
print("\nKalite ile Diğer Değişkenlerin Korelasyonları (Mutlak Değer):")
print(kalite_korelasyonlari.to_markdown(numalign="left", stralign="left"))

# Load the dataset
```

```
wine_data = veri

# Generate a heatmap of the correlation matrix
plt.figure(figsize=(12, 8))
correlation_matrix = wine_data.corr()
heatmap = sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', linewidths=0.5)

# Set the title of the heatmap
plt.title('Heatmap of Wine Quality Dataset Correlations')
plt.show()
```

Tahmin.py

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, mean_absolute_error
import altair as alt

# Veriyi yükle
df = pd.read_csv('WineQT.csv')

# Kalite eşiğini belirle ve quality_label sütununu oluştur
quality_threshold = 6
df['quality_label'] = ['kaliteli' if quality >= quality_threshold else 'kalitesiz' for quality in
df['quality']]

# Özellik matrisini (X) ve hedef değişkeni (y) oluştur
X = df[['pH', 'alcohol', 'sulphates']]
y = df['quality']  # Hedef değişken olarak quality kullanılacak

# Veriyi eğitim ve test kümelerine ayır
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# SVR modelini oluştur ve eğit
model = SVR(kernel='linear')
model.fit(X_train, y_train)

# Modelin performansını değerlendir (MSE ve MAE)
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
print(f'Mean Squared Error (MSE): {mse:.2f}')
print(f'Mean Absolute Error (MAE): {mae:.2f}')
```

```
# Kullanıcının şarap bilgileri (örnek değerler)
user_wine = [[3.5, 10.5, 0.6]]

# DataFrame'e dönüştürme
user_wine_df = pd.DataFrame(user_wine, columns=['pH', 'alcohol', 'sulphates'])

# Tahmin yapma
prediction = model.predict(user_wine_df)

# Tahmin edilen kaliteyi ekrana yazdır
print("Tahmin edilen şarap kalitesi (puan):", prediction[0])

# Grafik Çizimi

# pH Değeri Dağılımı - Kaliteye Göre
chart1 = alt.Chart(df).mark_boxplot().encode(
   x='quality_label:N',
   y='pH:Q',
   color='quality_label:N'
).properties(
   title='pH Değeri Dağılımı - Kaliteye Göre'
).interactive()

chart1.save('ph_dagilimi_kaliteye_gore_boxplot.json')

# Sülfat Oranı Dağılımı - Kaliteye Göre
chart2 = alt.Chart(df).mark_boxplot().encode(
   x='quality_label:N',
   y='sulphates:Q',
   color='quality_label:N'
).properties(
   title='Sülfat Oranı Dağılımı - Kaliteye Göre'
).interactive()

chart2.save('sulfat_orani_dagilimi_kaliteye_gore_boxplot.json')

# Alkol Oranı Dağılımı - Kaliteye Göre
chart3 = alt.Chart(df).mark_boxplot().encode(
   x='quality_label:N',
   y='alcohol:Q',
   color='quality_label:N'
).properties(
   title='Alkol Oranı Dağılımı - Kaliteye Göre'
).interactive()
```

```
chart3.save('alkol_orani_dagilimi_kaliteye_gore_boxplot.json')

# pH, Sülfat ve Alkol Oranları Arasındaki İlişki
chart4 = alt.Chart(df).mark_circle().encode(
    x='pH:Q',
    y='sulphates:Q',
    size='alcohol:Q',
    color='quality_label:N',
    tooltip=['pH:Q', 'sulphates:Q', 'alcohol:Q', 'quality_label:N']
).properties(
    title='pH, Sülfat ve Alkol Oranları Arasındaki İlişki'
).interactive()

chart4.save('ph_sulfat_alkol_iliskisi_scatter_plot.json')
```

5) Deneysel Çalışmalar

Çalışmamızda 3 tane eğitim modülü kullandık(Decision tree, Destek vektörlü makine ve random forest) bu eğitim modüllerinde Random forest sınıflandırmasında aşırı öğrenime maruz kaldığı için model accuracy 1.00 olarak gözükmekte ancak Random forest tahmininde ve ya diğerlerinde aşırı öğrenim durumuyla karşılaşmadık.

Aşırı öğrenim

```
Model Accuracy: 1.00
Classification Report:
              precision    recall  f1-score   support

    kaliteli       1.00      1.00      1.00       124
   kalitesiz       1.00      1.00      1.00       105

    accuracy                           1.00       229
   macro avg       1.00      1.00      1.00       229
weighted avg       1.00      1.00      1.00       229
```

Destek Vektörlü

```
Mean Squared Error (MSE): 0.39
Mean Absolute Error (MAE): 0.47
Tahmin edilen şarap kalitesi (puan): 5.481558053352909
```

Decision Tree

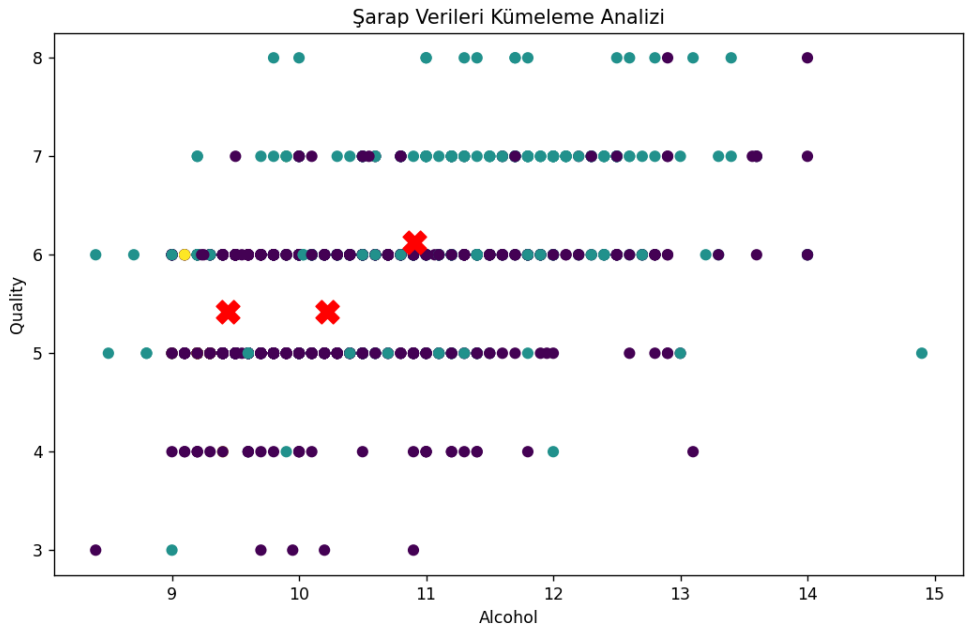
```
Model Accuracy: 0.68
Classification Report:
              precision    recall  f1-score   support

    kaliteli       0.71      0.71      0.71       127
   kalitesiz       0.64      0.65      0.64       102
```

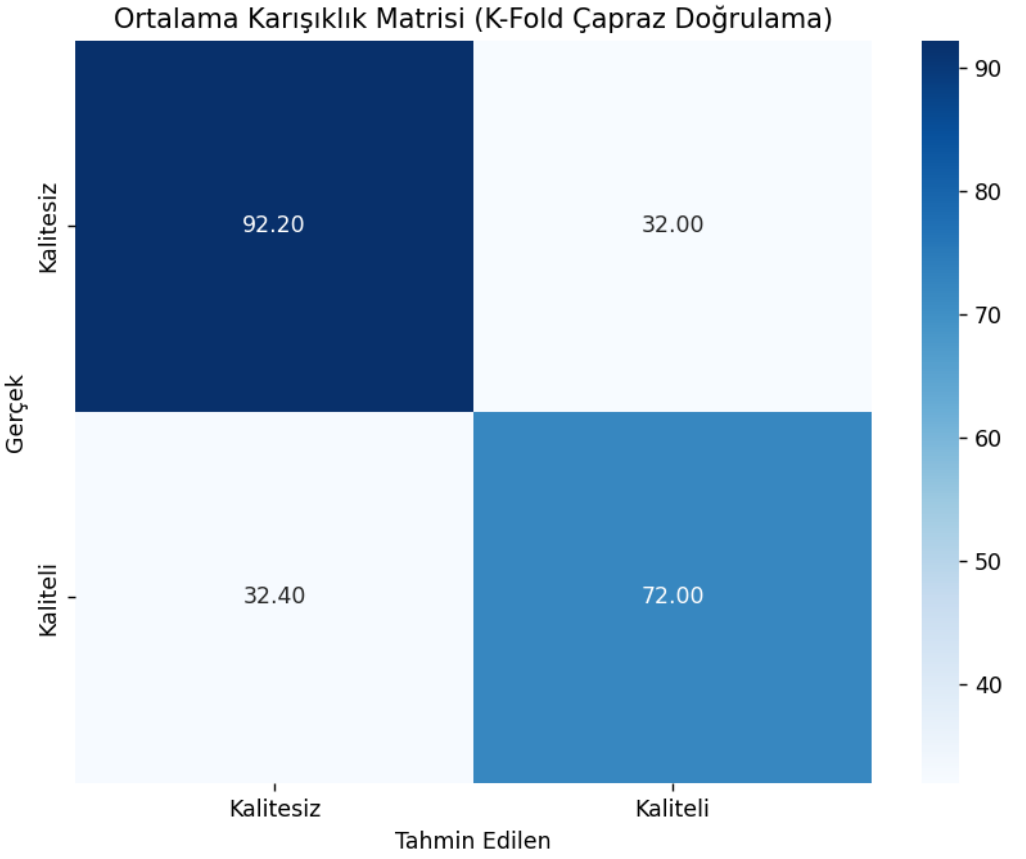
Random Forest Tahmin

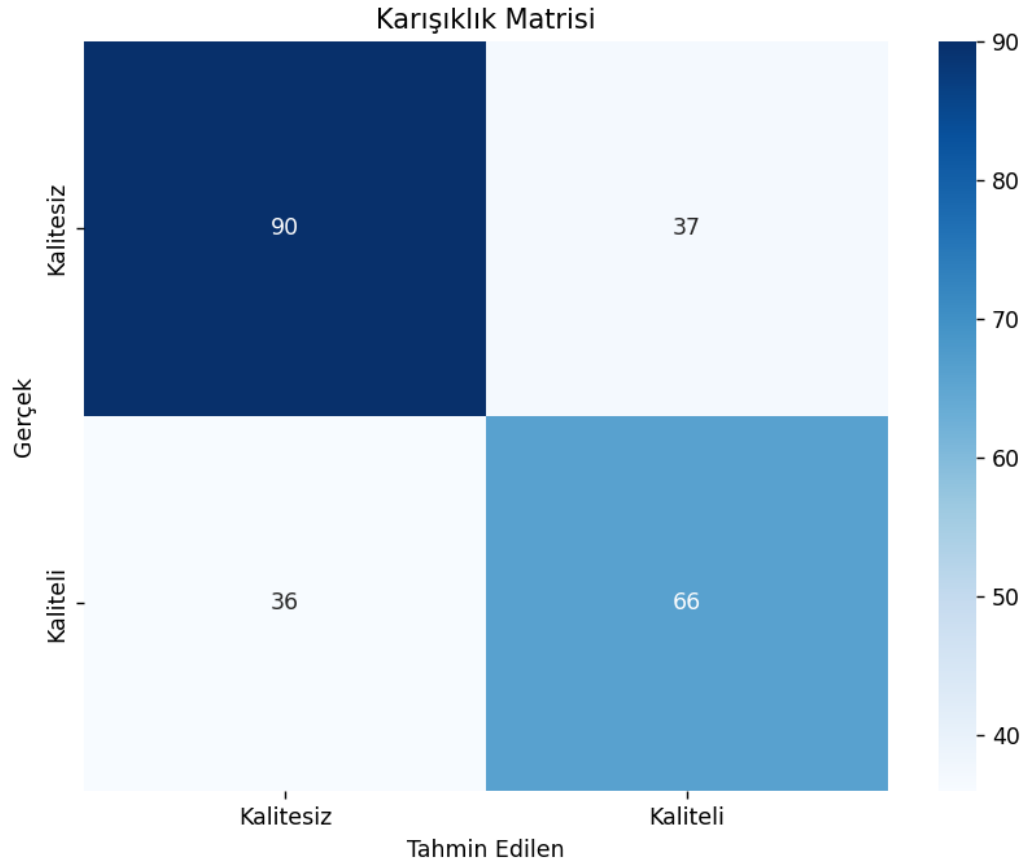
```
Model Accuracy: 0.77
Tahmin edilen şarap kalitesi: kalitesiz
```

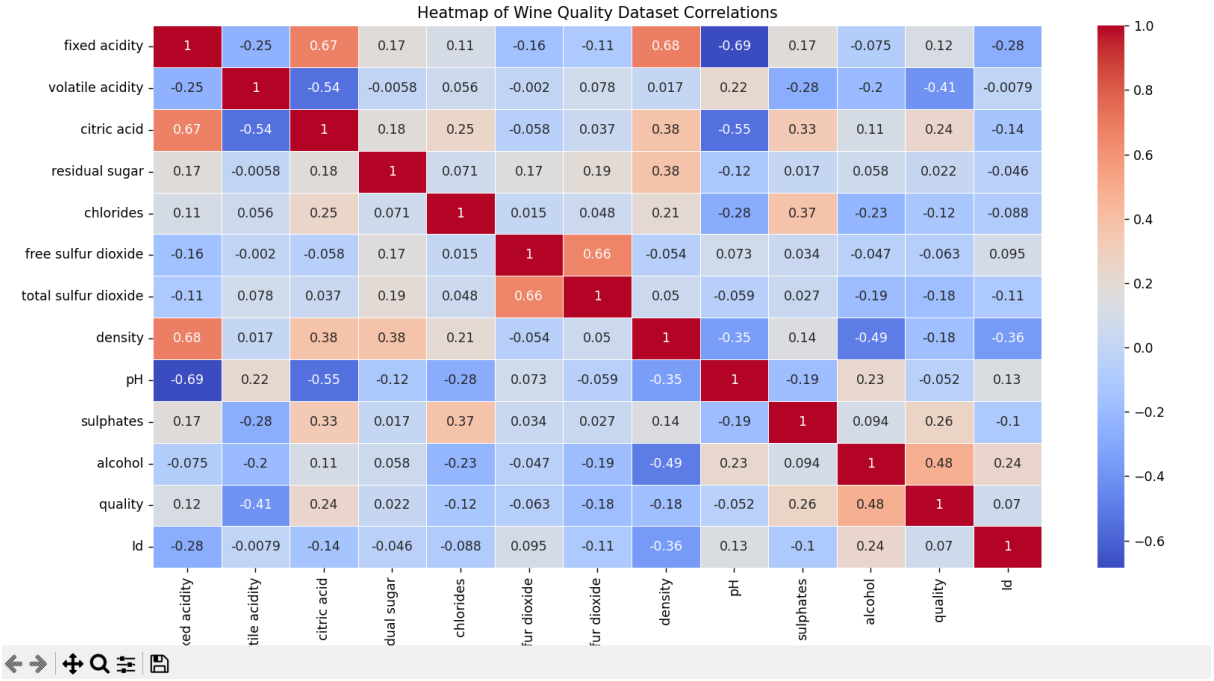
6) Sonuç

Data setini işleyerek Programa belli değerlerin(ph, alkol, sülfat) girilmesi durumunda program bize ürünün kalitesini verebilmekte




Ortalama Karışıklık Matrisi (K-Fold Çapraz Doğrulama)
Gerçek
Kalitesiz
Kaliteli
92.20
32.00
32.40
72.00
Kalitesiz
Kaliteli
Tahmin Edilen
90
80
70
60
50
40


Karışıklık Matrisi
Gerçek
Kalitesiz
Kaliteli
90
37
36
66
Kalitesiz
Kaliteli
Tahmin Edilen
90
80
70
60
50
40



Kalitenin direkt olarak herhangi bi nitelikle yüksek bir ilişkisi bulunmasa da başka değerlerinde de yardımıyla ürünün tahimini kalitesini projemiz aracılığıyla bulabiliyoruz özellikle alkol ve sülfat gibi ürünün kalitesiyle korelasyonu yüksek olan niteliklerin yanında bir de rastgele bir nitelik seçtik(ph)

7) Ek 1: Başarım İyileştirme

Başarım iyileştirebilmek için tahminlemeyi 3 farklı eğitim modülü ile yaptık

8)Ek 2: Literatür Katkısı

Literatüre katkısı

Uygulamada SVR, Random forest ve Decision Tree gibi farklı metotlar sayesinde tahminleme yapabildik
Plot ve altair kullanarak işlediğimiz verileri görselleştirdik.
Regresyon hesabıyla niteliklerin birbiriyle ilişkilerini gösterik
Regresyon tablosuna bakarak kalite ile ilişkisi yüksek 2 tane değer ve bir tane de ilişkisi daha düşük nitelikle bir tahminleme yaptırdık burada nitelikleri farklı seçebilirdik ve tabii ki farklı niteliklerde farklı başarı oranları elde ederdik.

Tek bir tahminleme modeline bağımlı kalmadan farklı modellerden de yardım alınarak daha doğru bir sonuca yaklaşılabilir

Bu çalışmayı özellikle şarap üreticileri ve ya şarap restaurantları kullanabilir. Tahminleme sayesinde üreticilere ve restaurantlara yardımı dokunabilir.
Çalışmamız farklı eğitim modülleriyle geliştirebilir ve ya daha geniş bir veri setiyle daha doğru bir hale getirilebilir

9) Ek 3:Faydanılan Kaynaklar

Öncelikle Kaggle'dan hem data setinde hem de örnek çalışmalar ve o çalışmalardaki kodlardan yardım aldık daha sonrasında veriyi işlerken, kütüphaneleri kullanırken ve sonuçlarımı görselleştirmeye çalışırken geeksforgeeks gibi bazı kaynaklardan yardım aldık.

10) Ek 4: İş Paketleri ve İşbölümü

Data setinin seçimi hepimizin ortak kabulüydü.
Daha sonrasında Topluma katkısı konusunda birkaç fikir sunduk en sonunda kabul gören fikir üzerine kodlamaya başladıktan sonra her birimizin farklı eğitim modüllerini kullanarak kodlama yapmasına böylece hem proje hakkında daha iyi bilgi sahibi olabileceğimizi hem de hepimizin bi yardımı olsun diye farklı eğitim modülleriyle kodlamaları yaptık
Decision tree – Ceyda Yapar
Random Forest – Mert Erarslan
Destek vektörlü makine – Yasir Yavuz

Programımızın tahmin yapabilmesi için bu eğitim modüllerini kullandık.

11) Öz değerlendirme Tablosu

	İstenen Madde	Var	Açıklama	Tahmini Not
1	**Kapak Sayfası, Problemin Tanımı, Kullanılan Ortam, Yöntem ve Kütüphaneler, Araştırma (10)**	☒	10	10
2	**Önerilen Yöntem (10)**	☒		10
3	**Deneysel Çalışmalar (10)**	☒		10
4	**Proje Rapor Biçimi, Organizasyonu, Boyutu, Kalitesi, Kaynakça ve atıflar (10)**	☒	**Boyutu çok büyük değil ve Kaynakçalar tam olarak hatırlanmadığından net belirtilemedi**	7
5	**Sonuç (10)**	☒		10
6	**Ek 1: Başarım İyileştirme (10)**	☒	3 farklı eğitim modülü kullanıldı	10

7	**Ek 2: Literatür Katkısı (10)**	☒	Tam olarak doğru doldurulmamış olabilir	7
8	**Ek 3: Faydanılan Kaynaklar (10)**	☒	Tam olarak kaynak belirtilmedi	8
9	**Ek 4: İş Paketleri ve İşbölümü (10)**	☒		10
10	**Özdeğerlendirme Tablosu (10)**	☒		10
			100 üzerinden Toplam Not:	90