

# **Yemek Malzemelerine Göre Ülke Tahmini**

**Halil Cüneyt TURHAN**  
**030122019**



# Yemeklere göre ülke tahmini

Yemek malzemerinden hangi ülke mutfağına ait olduğunu tahmin eden ve bunu elimizdeki veri setinden karşılaştıran bir model

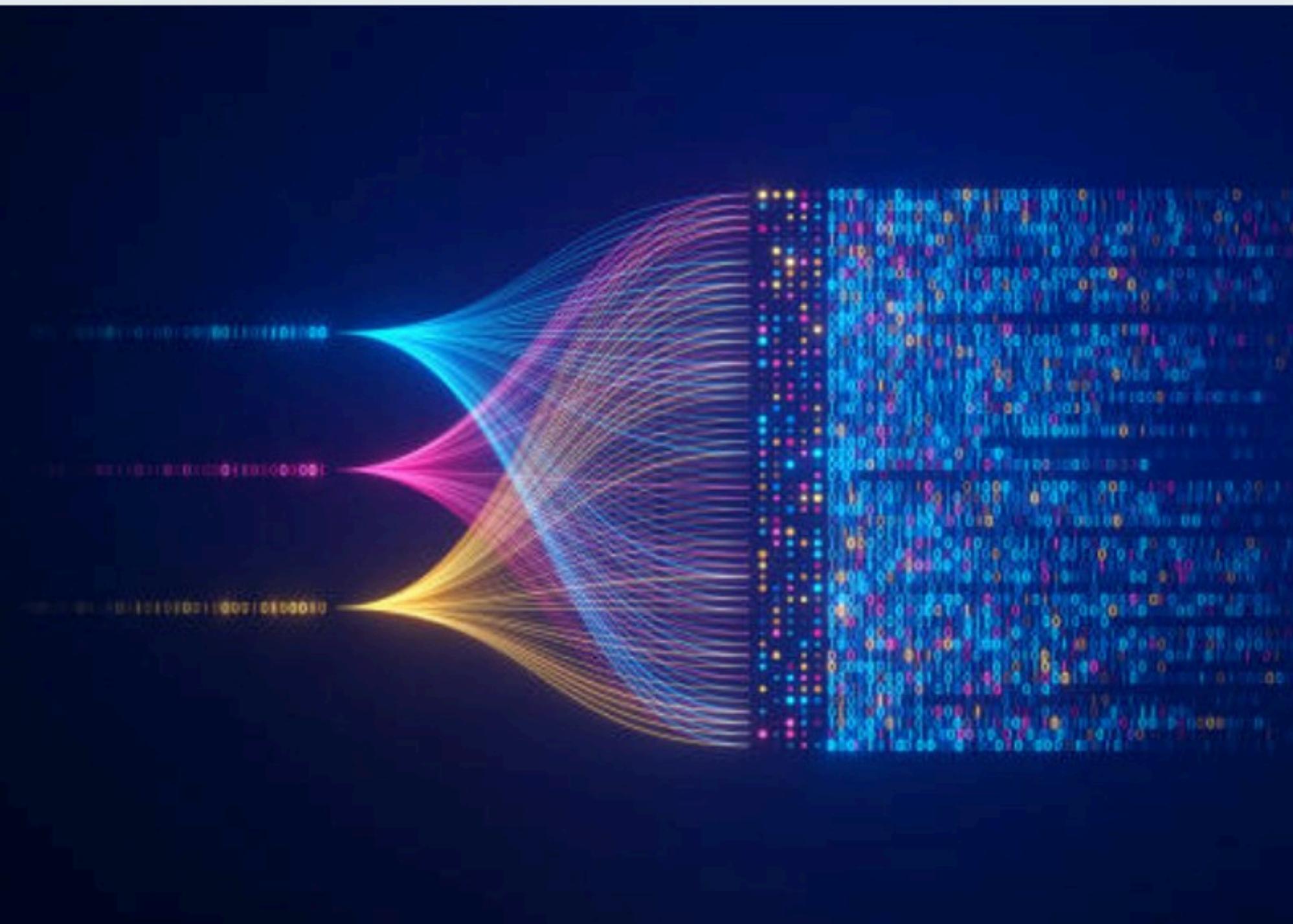
# Hangi yemek hangi ülkeye ait?

Yüzyıllardır yaşadığımız coğrafya da bizim kültürümüze yerleşmiş yemeklerin asıl hangi ülkelerden çıktığını ve malzemelerin kullanılarak yemek kültürlerinin genişlemesinden dolayı mutfaklarda yer edinen yemeklerin asıl hangi ülkelere ait olduğunu gösteren bir model.



# Kullanılan Modeller

- *Naive Bayes*
- *SVM*
- *Random Forest*
- *kNN*



# Data Set Filtreleme İşlemi

Veri setimizde 600.000'den fazla veri bulunuyordu ancak o kadar dağınık ve fazla mutfak türü bulunuyordu ki filtreleme işlemi yaptık. Bazı ülke mutfaklarını çok az örneğe sahipti bu yüzden modelin daha doğru öğrenmesi ve dengeli tahmin yapabilmesi için filtreleme yaptık.



”

# Data Set

Kullanılan veri seti

<https://www.kaggle.com/datasets/kaggle/recipe-ingredients-dataset/data>

Veri setimizde  
**southern\_us**  
**spanish**  
**italian**  
**french**  
**japanese**  
**turkish** verileri bulunmaktadır.



# Yapılan İşlemler

- Kütüphaneleri import ettik
- Veri seti filtreleme işlemi
- Model karşılaştırılması
- Test ve Eğitim,matplotlib
- Normalizasyon
- Özellik seçimi uygulandı
  - SelectKBest
  - RFE
  - Tree-Based
- Model eğitimi ve değerlendirilmesi,
- Çıktıların gösterilmesi

# Sunumda Veri Ön İşleme

## METİN TEMİZLEME (TEXT CLEANING)

- Malzemeleri metne dönüştürme
- Küçük harfe çevirme
- Noktalama ve özel karakterleri silme
- Etiketleri sayısal hale getirme (Label Encoding)
- Metinleri sayısal vektörlere dönüştürme (TF-IDF)
- Özellik seçimi (SelectKBest + chi2)
- Eğitim/Test veri setine ayırma



```
import matplotlib
import pandas as pd
import numpy as np
import re
import joblib
import matplotlib.pyplot as plt
import seaborn as sns
import json

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_selection import SelectKBest, chi2, RFE
from sklearn.preprocessing import LabelEncoder, MaxAbsScaler
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import LinearSVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score
from imblearn.over_sampling import SMOTE
```

```
# VERİ SETİ FILTRELEME
import pandas as pd
allowed_cuisines = ["southern_us", "spanish", "italian", "french", "japanese", "turkish"]
train_df = pd.read_json("veri_seti.json")
train_filtered = train_df[train_df["cuisine"].isin(allowed_cuisines)]
train_filtered.to_json("train.json", orient="records", indent=2)
print("✓ Filtreleme tamamlandı!")
```

```
#MODEL KARŞILAŞTIRILMASI
def save_metric_comparison_plots(results_df):
    metrics = ["Accuracy(Doğruluk)", "F1-Score", "Precision(Kesinlik)", "Recall(Duyarlılık)"]
    for metric in metrics:
        plt.figure(figsize=(12, 6))
        sns.barplot(x="Modeller", y=metric, hue="Özellik seçimi", data=results_df, palette="Set2")
        plt.title(f"Model Başarı Karşılaştırması - {metric}")
        plt.ylabel(metric)
        plt.xticks(rotation=45)
        plt.tight_layout()
        plt.savefig(f"{metric.lower()}_ws.png")
        plt.close()
```

```
#Veri setini kaç kaç böldüğümüzü gösteren grafik
def save_data_split_pie_chart(train_size, test_size):
    sizes = [train_size, test_size]
    labels = ['Eğitim Verisi (%80)', 'Test Verisi (%20)']
    colors = [■ '#00008b', ■ '#ff0000']
    plt.figure(figsize=(6, 6))
    plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=140)
    plt.title('Veri Seti Dağılımı (Toplam %100)')
    plt.axis('equal')
    plt.tight_layout()
    plt.savefig("veri_dagilimi.png")
    plt.close()
```

```
def main():
    #veri okuma
    df = pd.read_json("train.json")
    df["text"] = df["ingredients"].apply(lambda ing: re.sub(r'([a-zA-Z]+)(\d+)', r'\1 \2', ing))
    #temiz veri
    X_raw = df["text"]
    y = df["cuisine"]
    #label encoding
    le = LabelEncoder()
    y_encoded = le.fit_transform(y)
    #TF-IDF
    vectorizer = TfidfVectorizer()
    X_vect = vectorizer.fit_transform(X_raw)
    #MaxAbsScaler
    scaler = MaxAbsScaler()
    X_scaled = scaler.fit_transform(X_vect)
```

# Özellik Seçimleri

## SelectKBest:

SelectKBest, her özelliği tek tek hedef değişkenle istatistiksel olarak değerlendirir. En yüksek skora sahip K tane özelliği seçerek gereksiz değişkenleri elimine eder. Büyük veri setlerinde bu yöntem, hızlı bir ön eleme sağlar. Böylece sadece anlamlı özellikler modele girer, eğitim süresi ve karmaşıklık azalır.

## RFE:

RFE, bir tahmin modeli kullanarak tüm özelliklerini sıralı şekilde değerlendirir. Her iterasyonda en az katkı yapan özelliği eleyerek en iyi alt kümeyi bulmaya çalışır. Özellikle çok boyutlu veri setlerinde, etkisiz özellikleri adım adım azaltarak daha dengeli bir model oluşturmayı sağlar. Bu yöntem daha hesaplamalıdır ama sonuç olarak daha iyi genelleme performansı verir.

## Tree-Based:

Ağaç tabanlı yöntemler (örneğin Random Forest) her özelliğin model kararlarına olan katkısını skorlayarak önem derecelerini belirler. Bu önem skorlarına göre en etkili değişkenler seçilir. Karar aacı yapısı sayesinde etkileşimli ve doğrusal olmayan ilişkiler de yakalanabilir. Büyük veri setlerinde hem hızlı çalışır hem de yüksek doğruluk sağlar.

PS C:\Users\cavus\Desktop\umutunyaptigi\malzemeyuzdekarhangimutfakta> python main.py

--- Özellik Seçimi: SelectKBest ---

Naive Bayes - Doğruluk: %77.45, F1: %78.43, Precision: %80.80, Recall: %77.45

SVM - Doğruluk: %81.28, F1: %81.92, Precision: %83.35, Recall: %81.28

Random Forest - Doğruluk: %81.46, F1: %81.34, Precision: %81.39, Recall: %81.46

KNN - Doğruluk: %65.00, F1: %66.24, Precision: %75.55, Recall: %65.00

--- Özellik Seçimi: RFE ---

Naive Bayes - Doğruluk: %79.09, F1: %79.95, Precision: %82.38, Recall: %79.09

SVM - Doğruluk: %82.74, F1: %83.29, Precision: %84.52, Recall: %82.74

Random Forest - Doğruluk: %82.01, F1: %82.01, Precision: %82.10, Recall: %82.01

KNN - Doğruluk: %72.44, F1: %74.17, Precision: %80.81, Recall: %72.44

--- Özellik Seçimi: Tree-Based ---

Naive Bayes - Doğruluk: %77.12, F1: %78.09, Precision: %80.47, Recall: %77.12

SVM - Doğruluk: %81.49, F1: %82.05, Precision: %83.26, Recall: %81.49

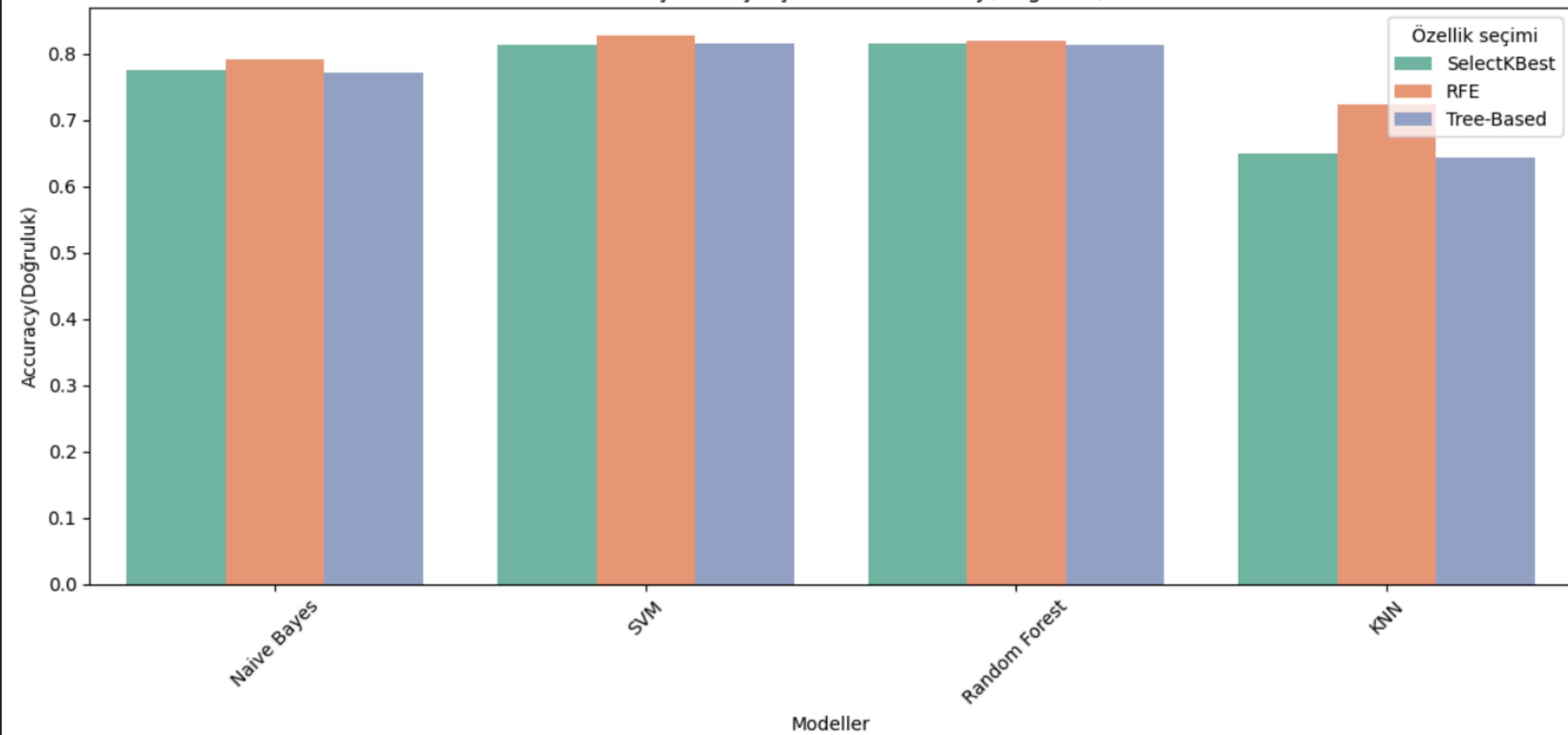
Random Forest - Doğruluk: %81.43, F1: %81.21, Precision: %81.28, Recall: %81.43

KNN - Doğruluk: %64.39, F1: %65.63, Precision: %76.61, Recall: %64.39

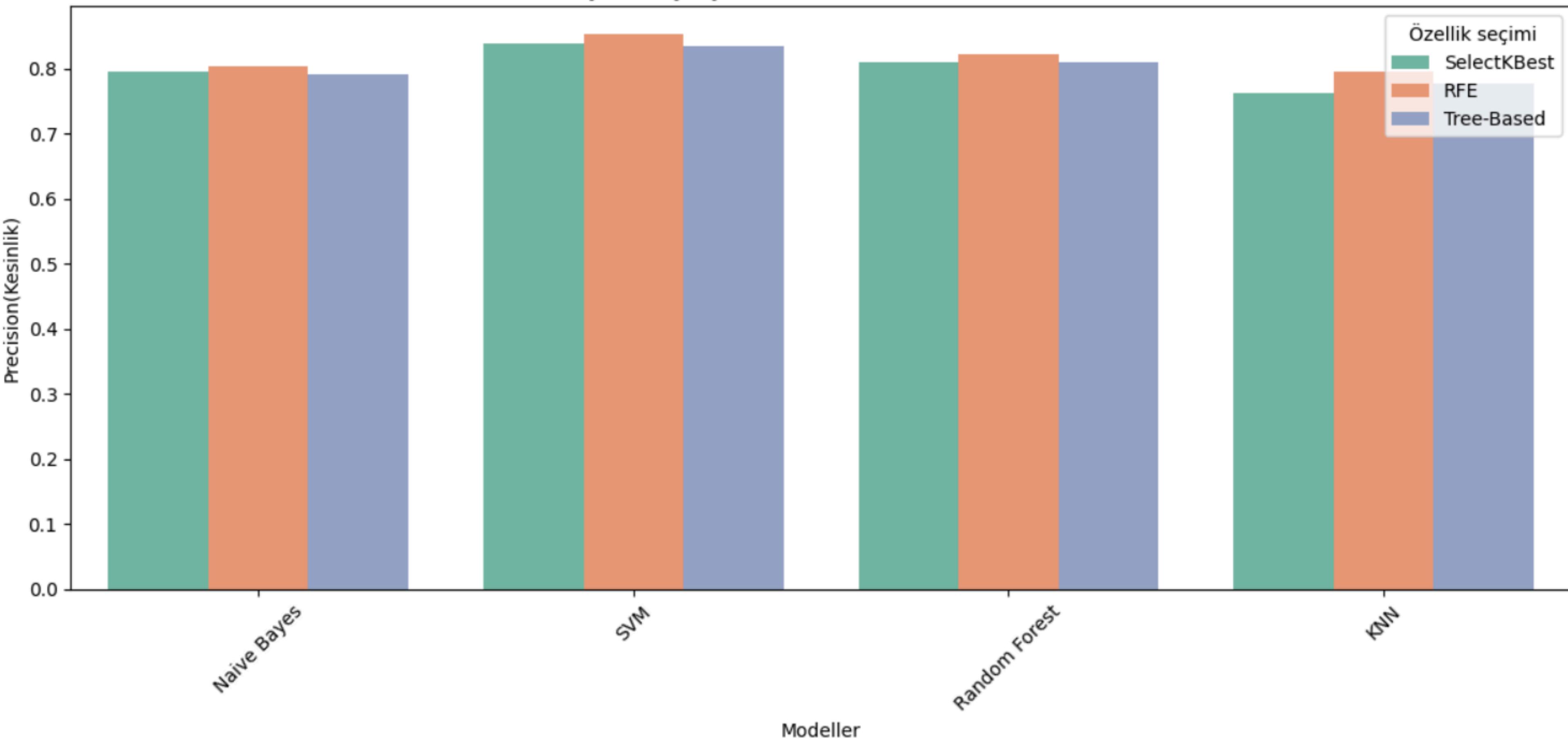
```
#Özellik seçimi yöntemleri 3 özellik 4 model ile karşılaştırılıyor.
feature_selectors = {
    "SelectKBest": SelectKBest(score_func=chi2, k=1000),
    "RFE": RFE(estimator=LinearSVC(max_iter=5000), n_features_to_select=1000, step=0.1),
    "Tree-Based": SelectKBest(score_func=lambda X, y: RandomForestClassifier(n_estimators=100, random_state=42).fit(X, y).feature_importances_, k=1000)
}

models = {
    "Naive Bayes": MultinomialNB(),
    "SVM": LinearSVC(max_iter=5000),
    "Random Forest": RandomForestClassifier(n_estimators=100, n_jobs=-1, random_state=42),
    "KNN": KNeighborsClassifier()
}
#baslangic da hepsini bos atadık en iyi model gelince burada kayitlanacak.
best_acc = 0
best_model = None
best_name = ""
best_selector_name = ""
best_metrics = {}
results = []
```

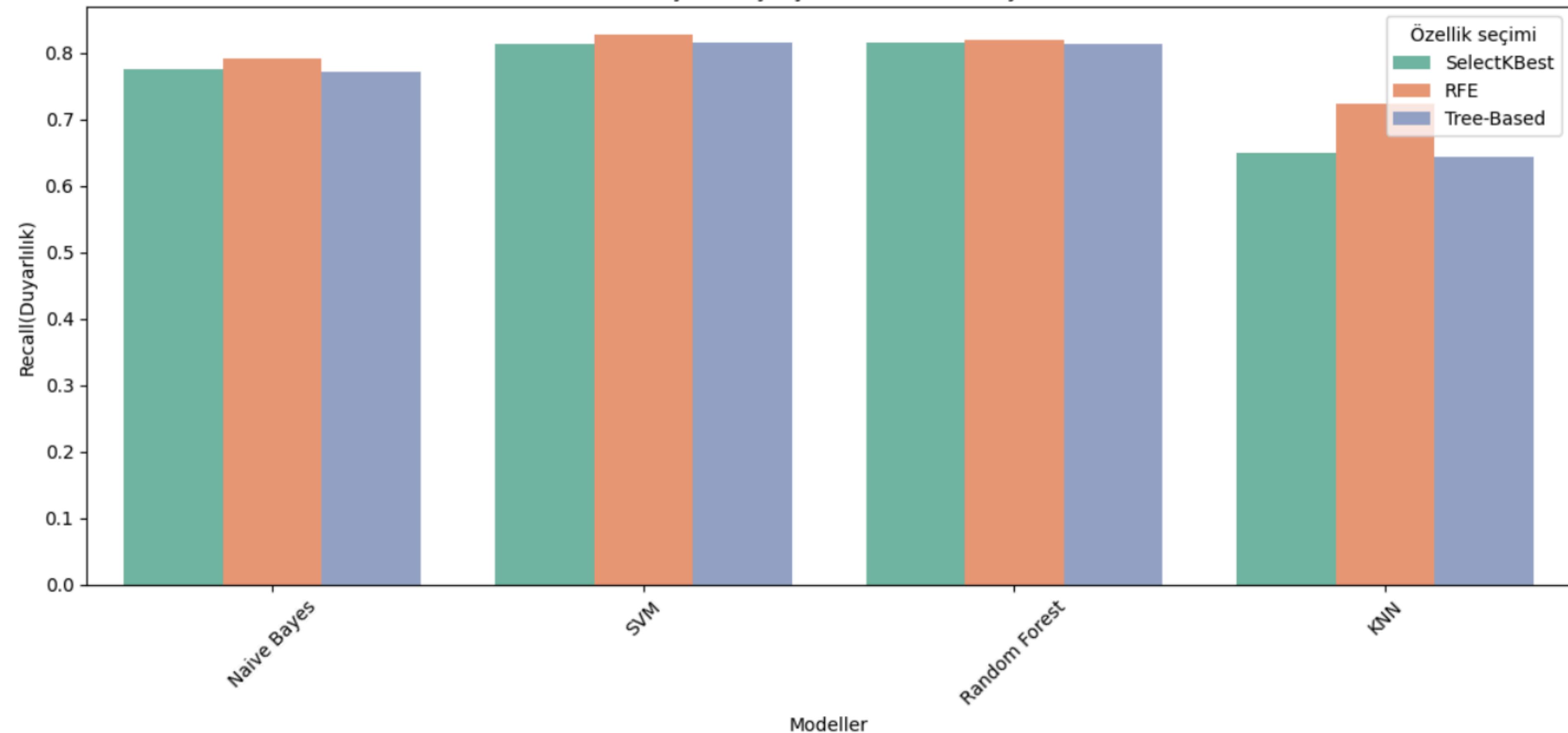
## Model Başarı Karşılaştırması - Accuracy(Doğruluk)



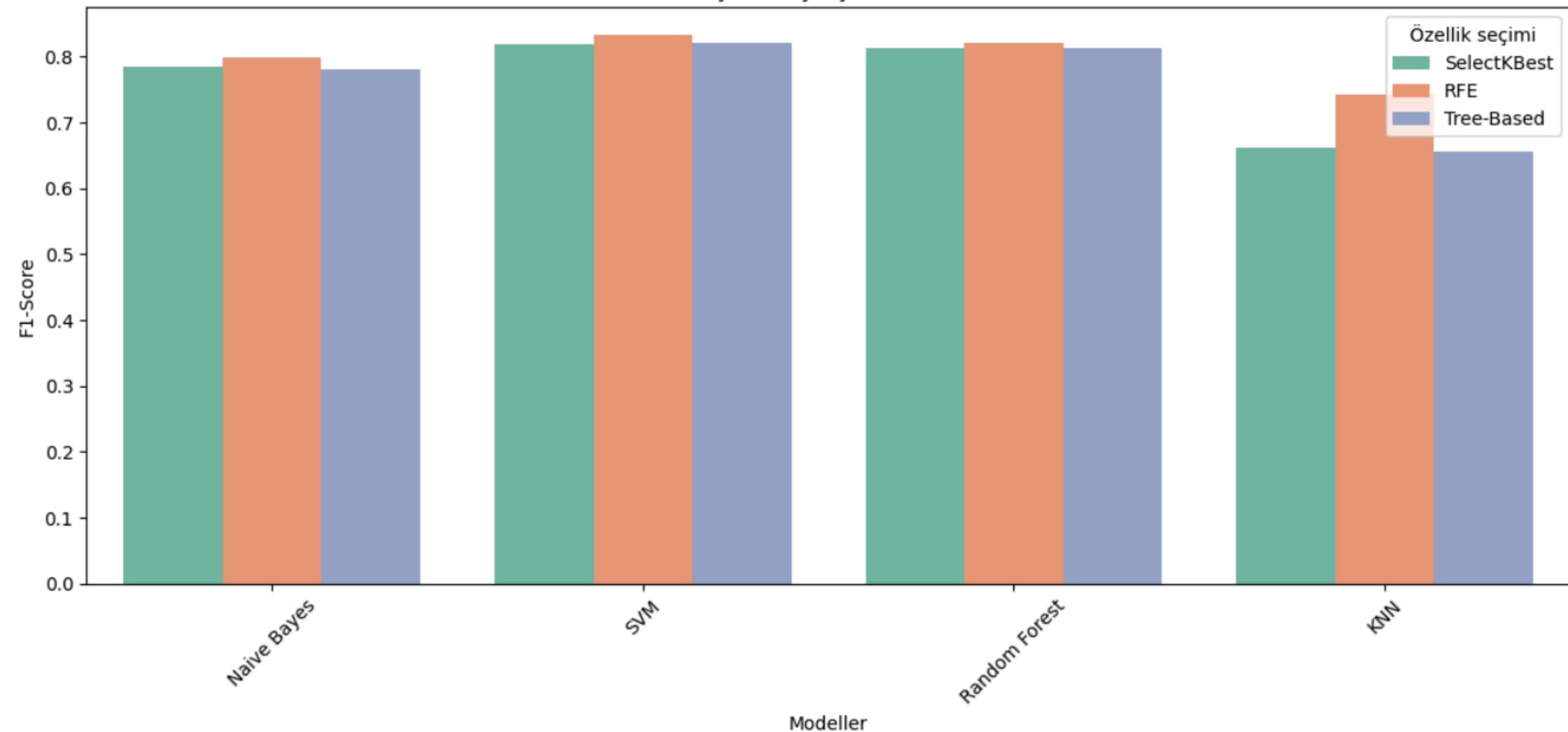
## Model Başarı Karşılaştırması (Test Verisi) - Precision(Kesinlik)

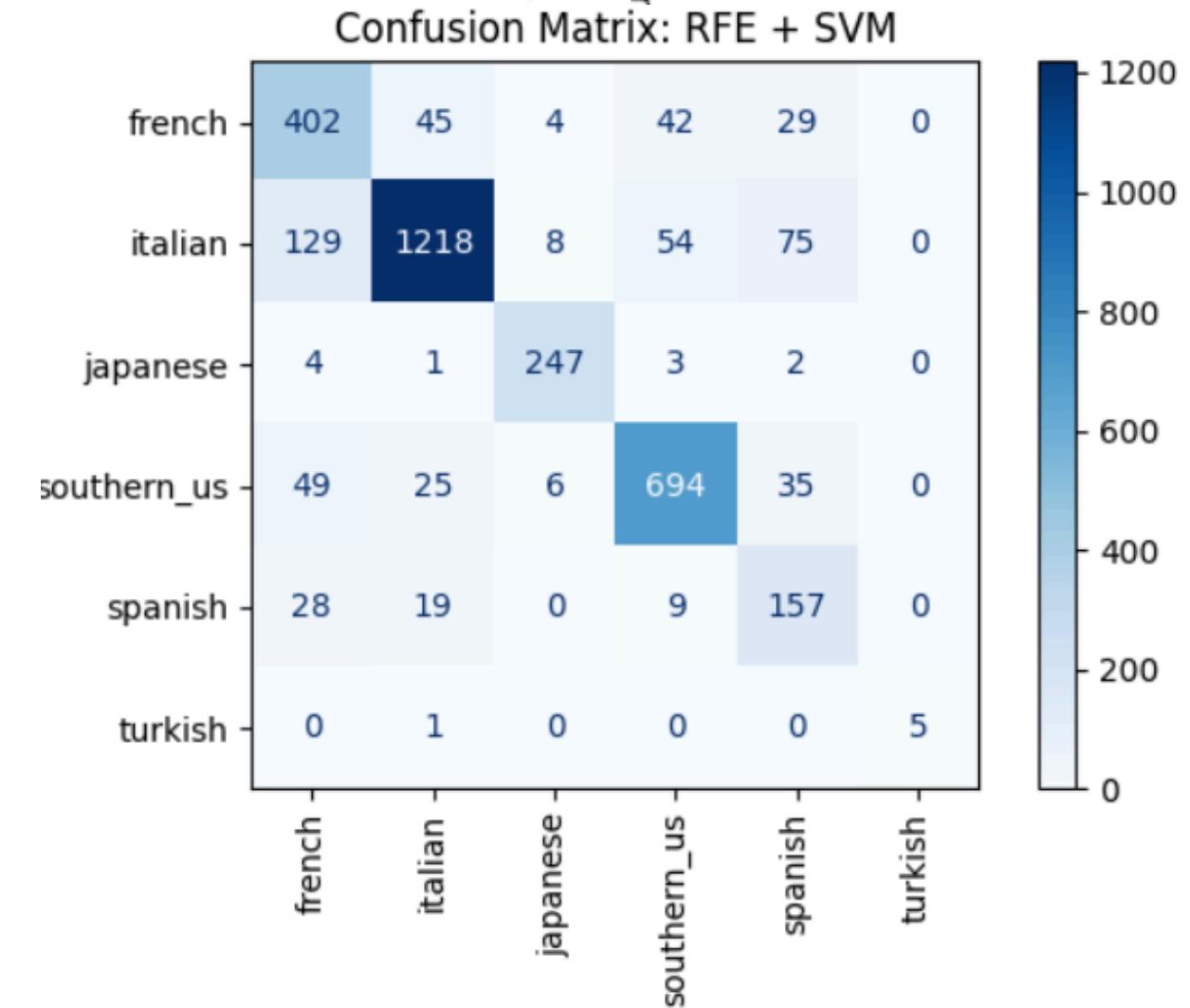
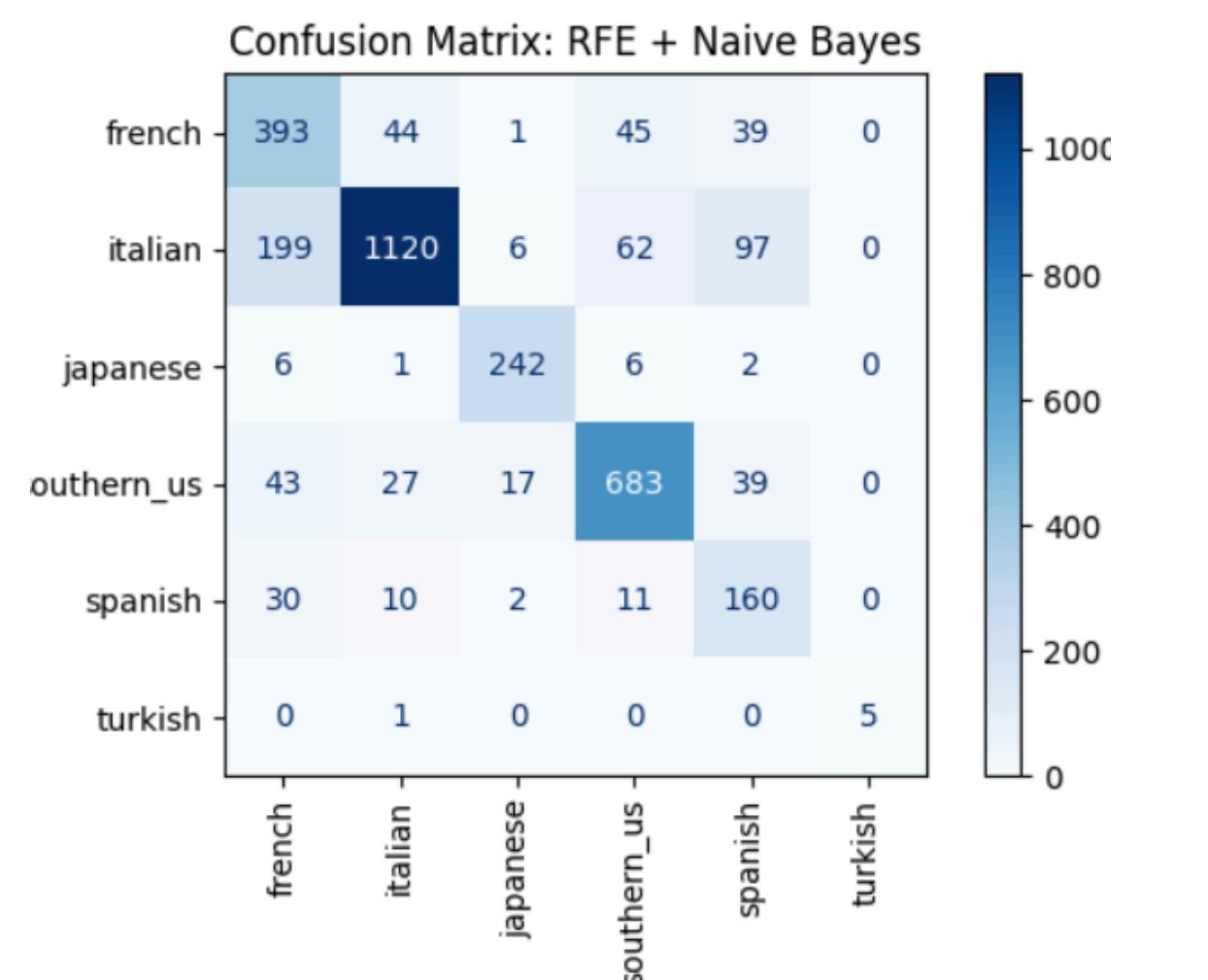
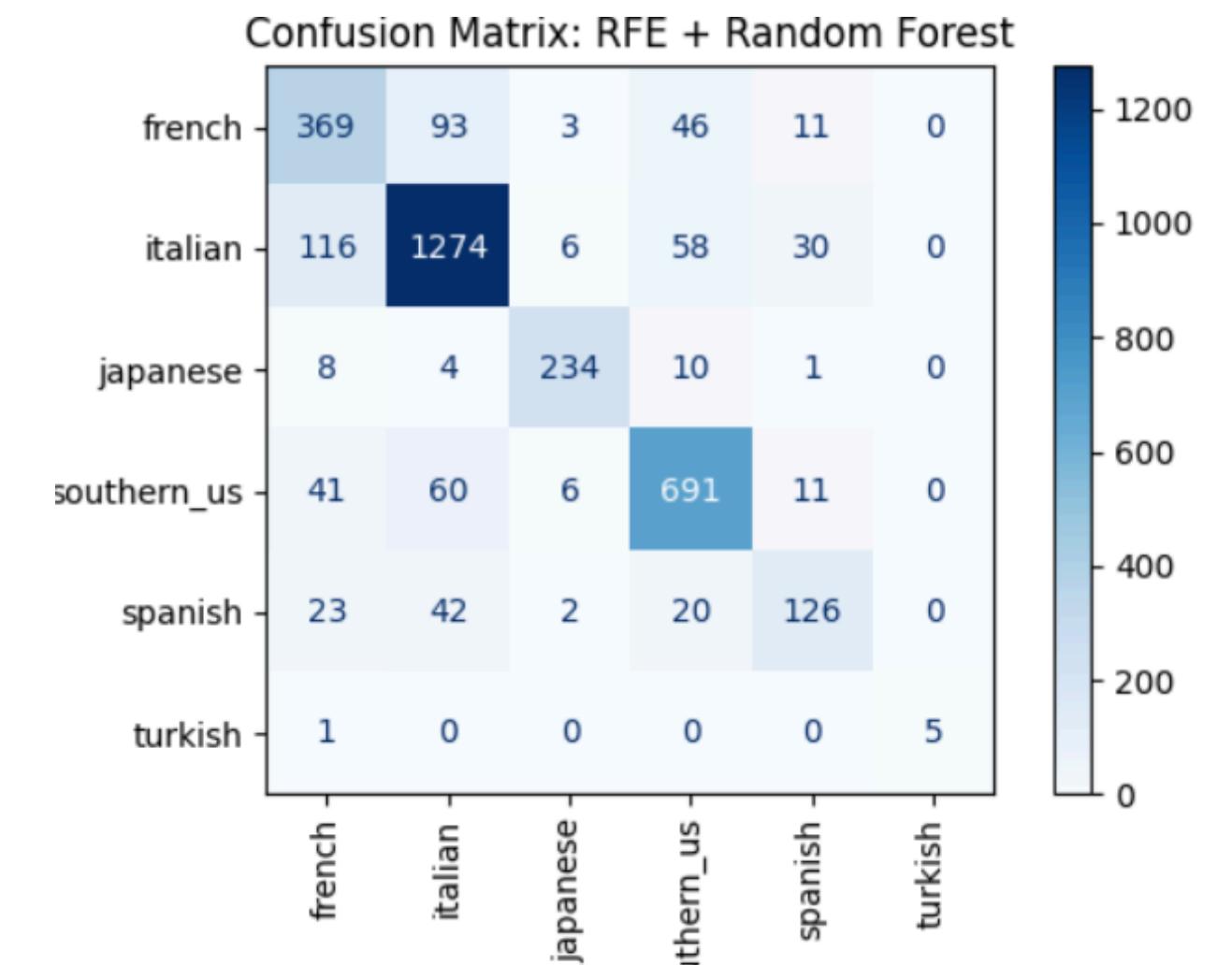
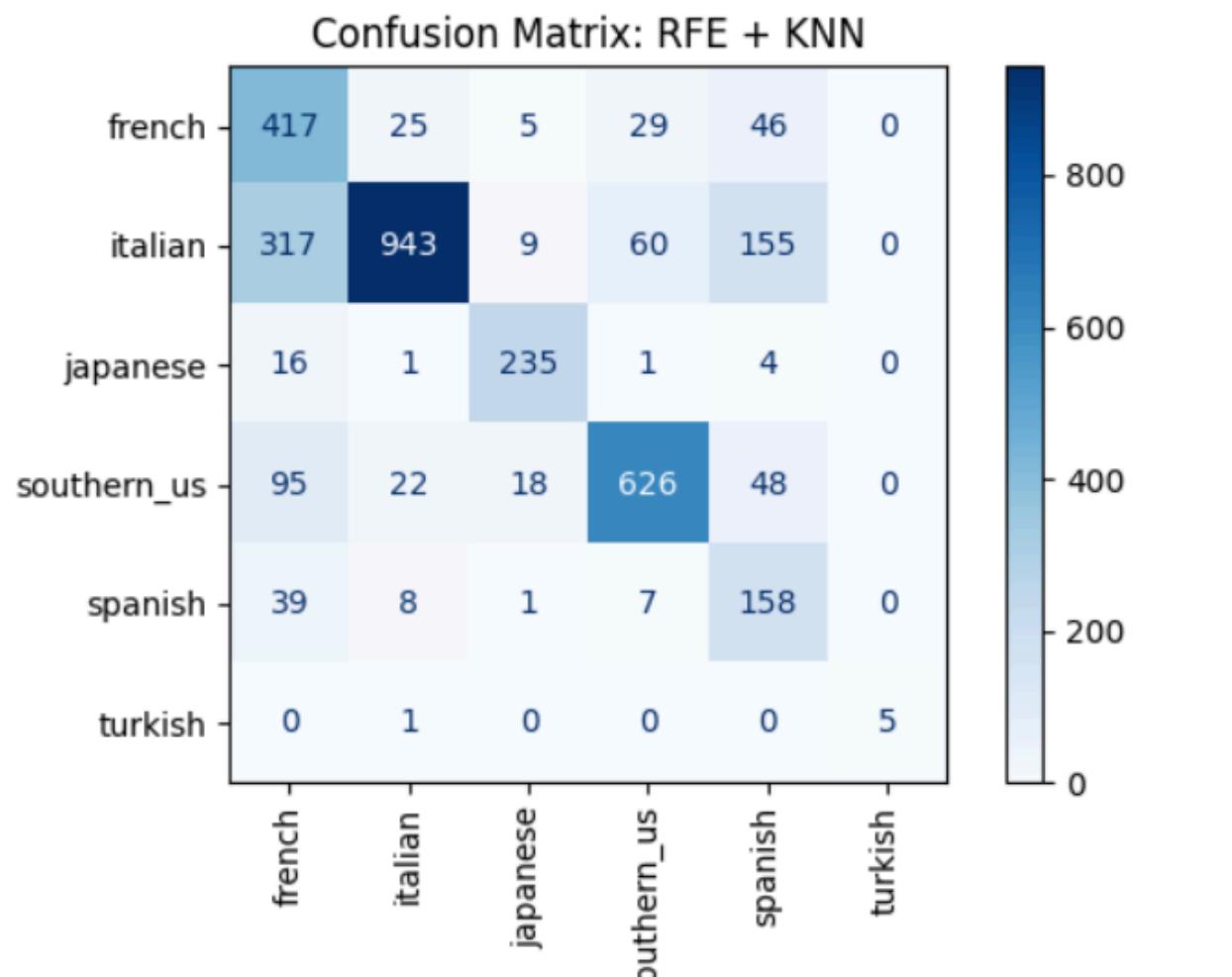


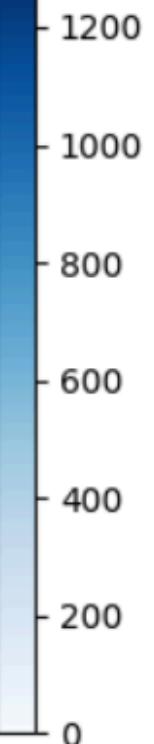
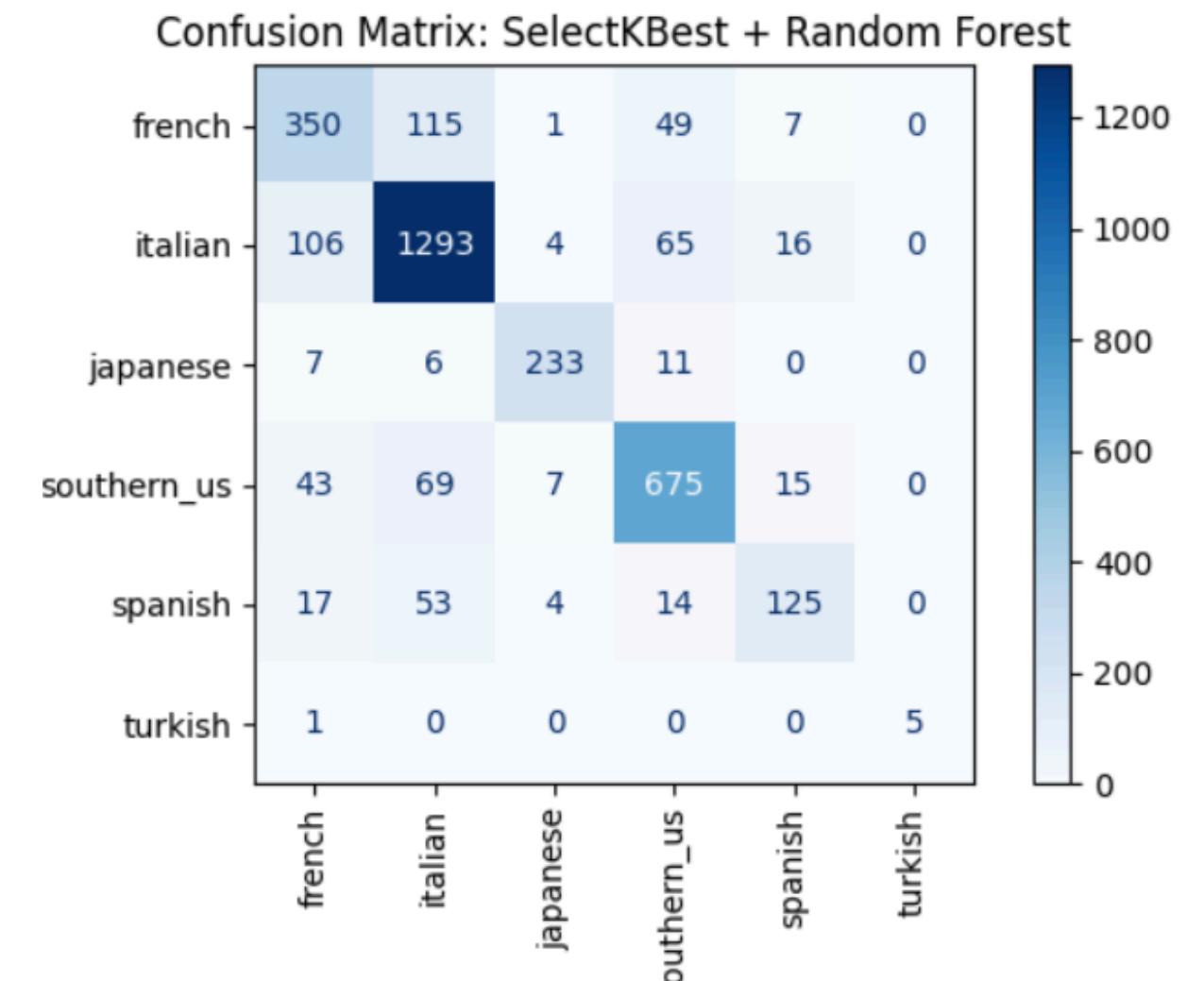
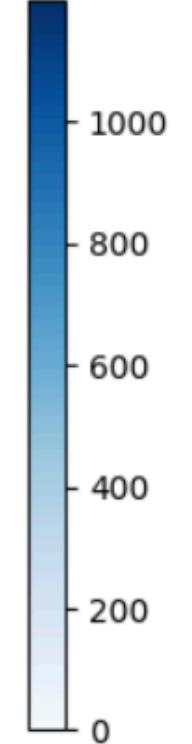
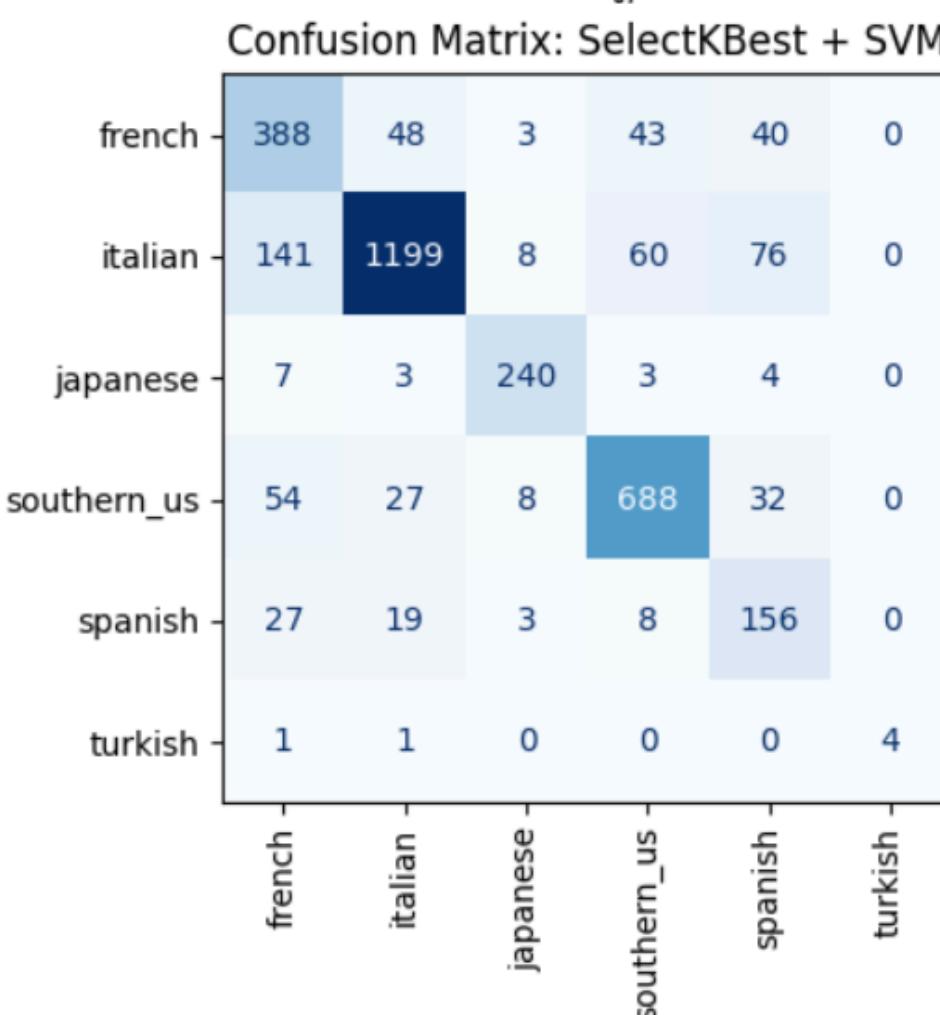
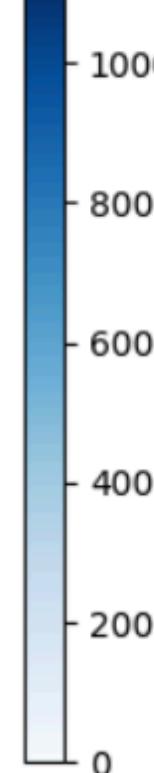
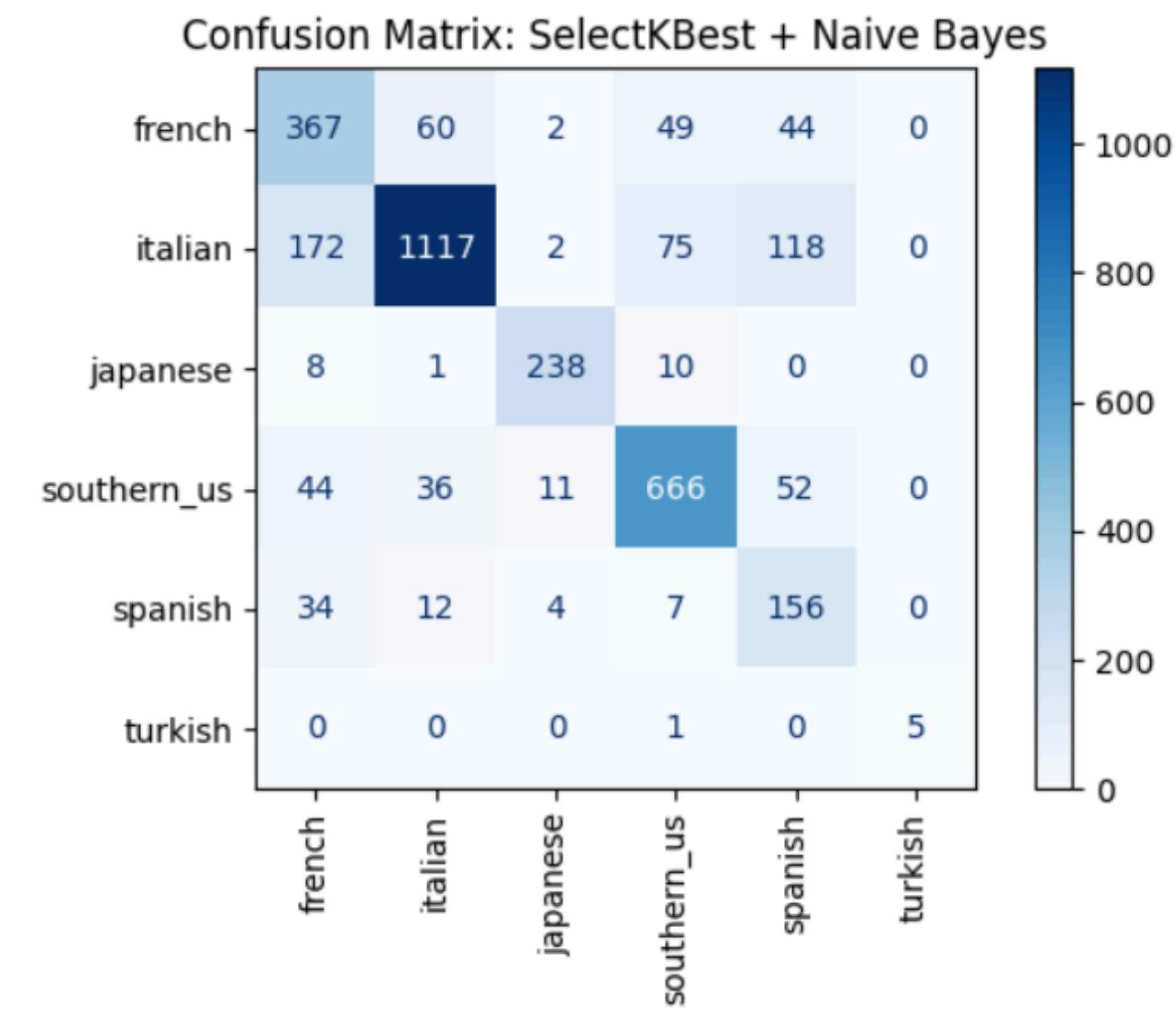
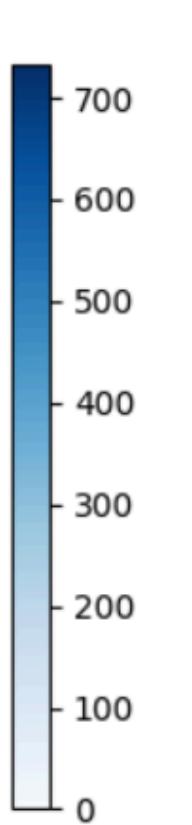
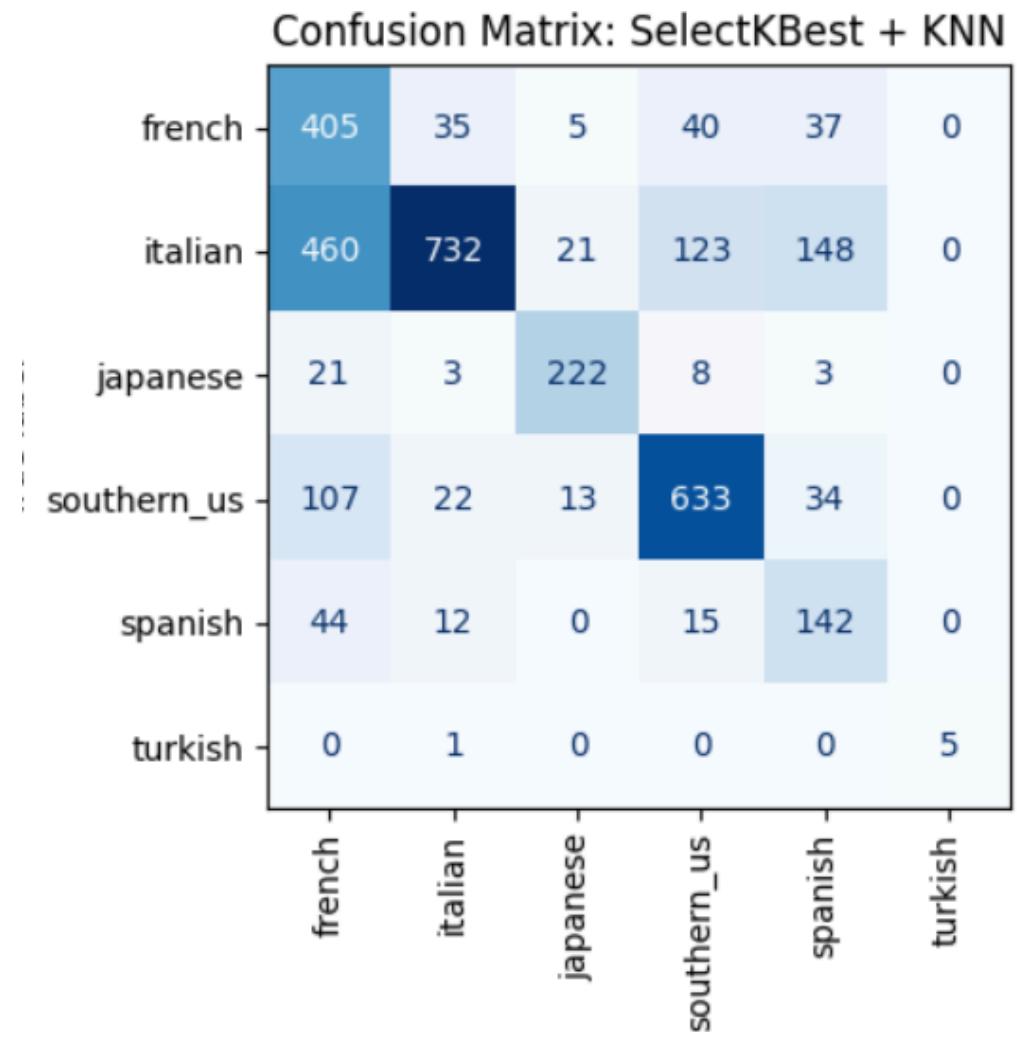
### Model Başarı Karşılaştırması - Recall(Duyarlılık)



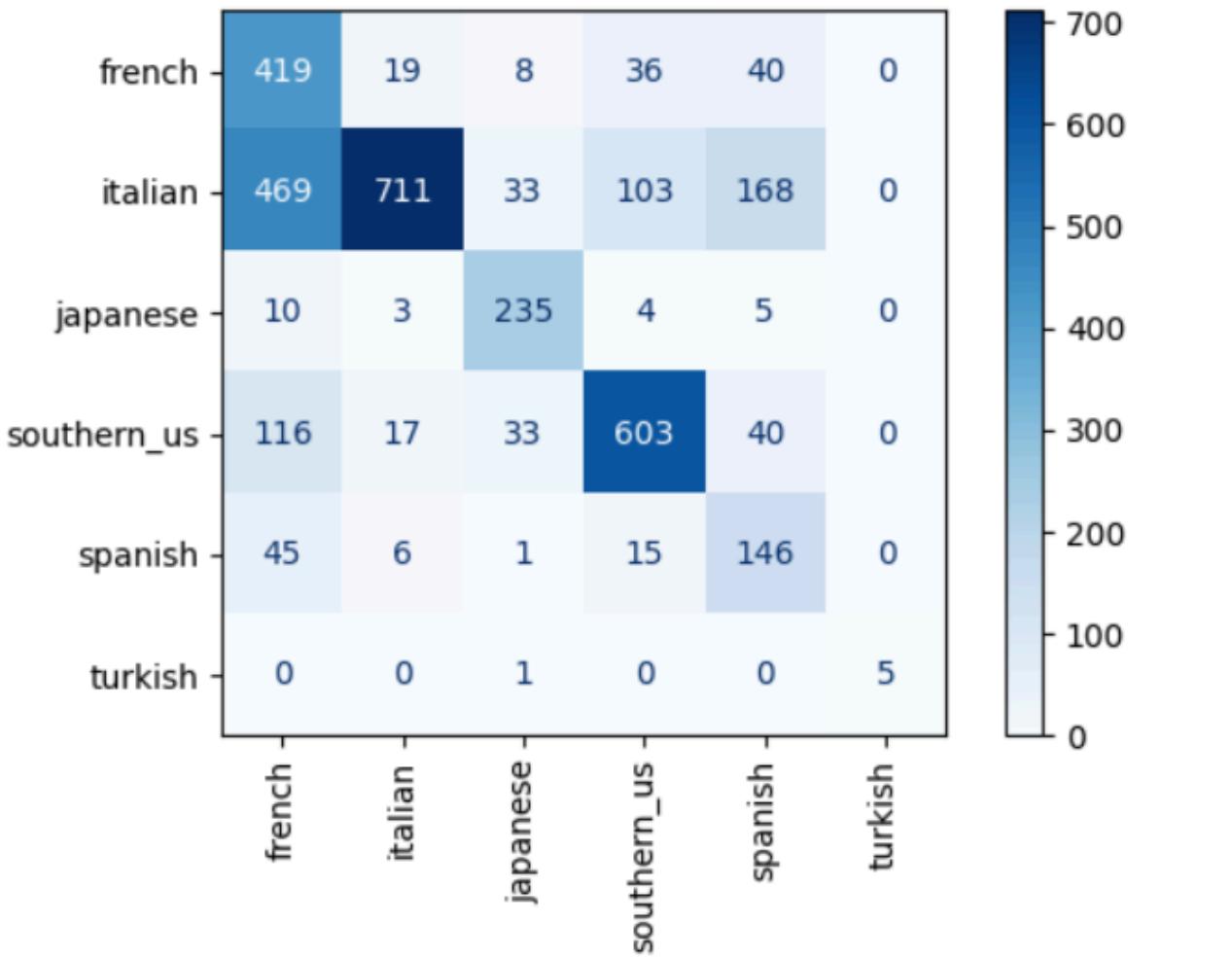
## Model Başarı Karşılaştırması - F1-Score



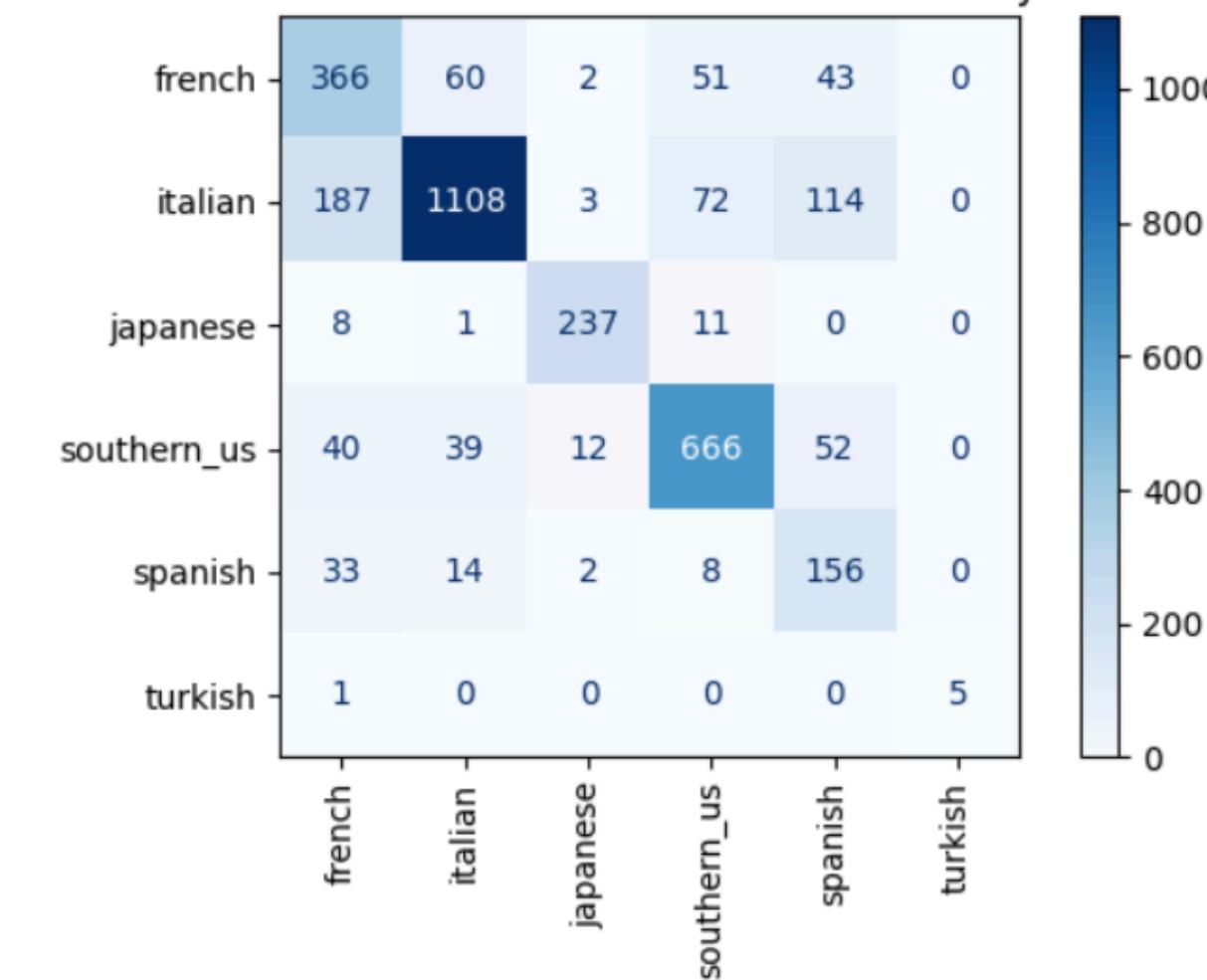




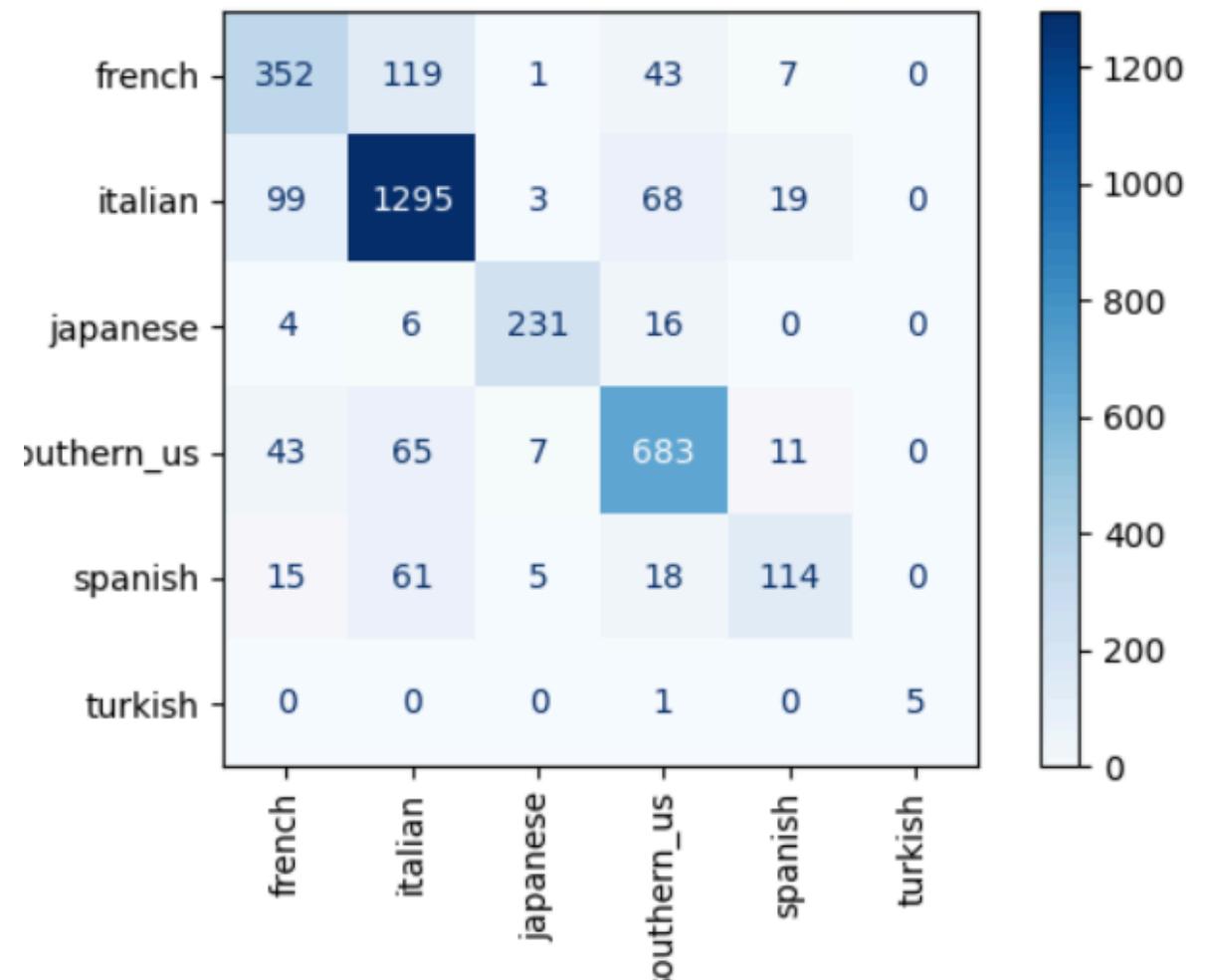
Confusion Matrix: Tree-Based + KNN



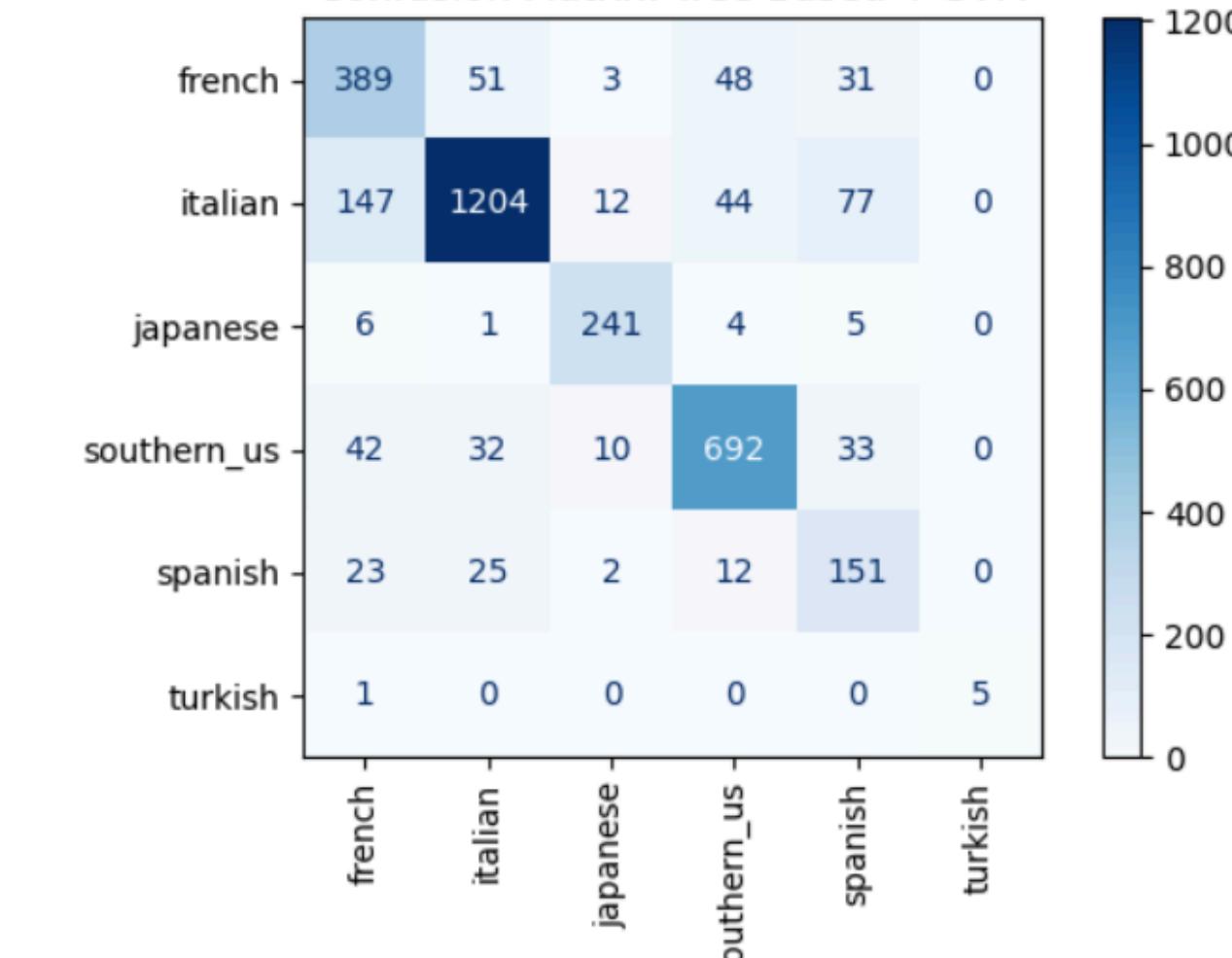
Confusion Matrix: Tree-Based + Naive Bayes



Confusion Matrix: Tree-Based + Random Forest



Confusion Matrix: Tree-Based + SVM

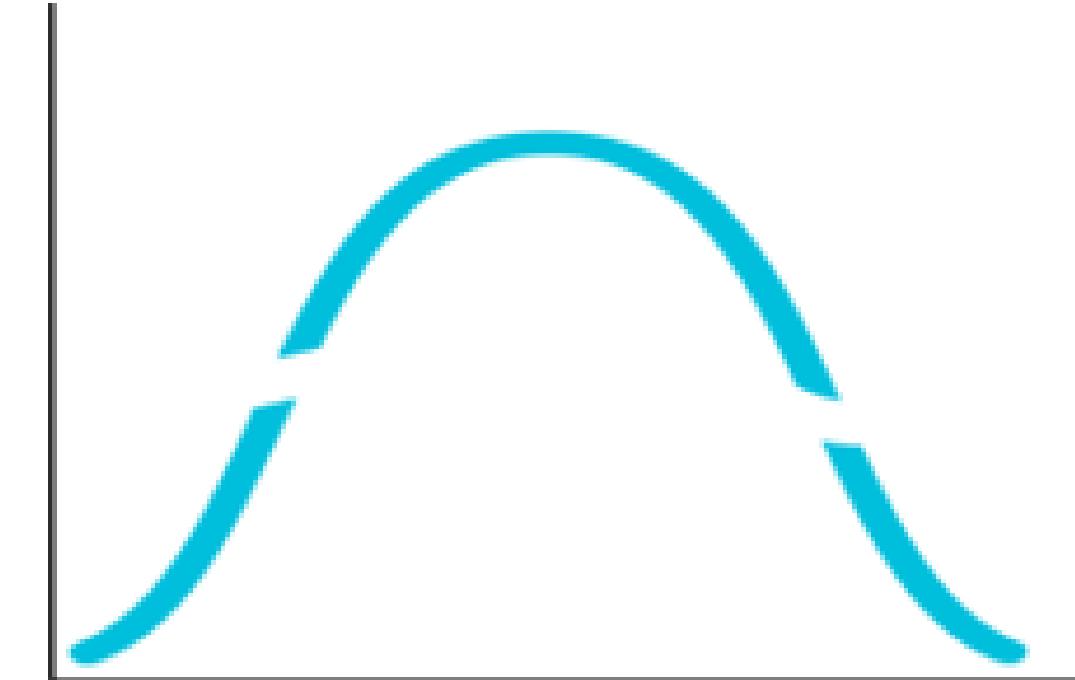


# Modelleme

## Naive Bayes

- Olasılık temelli bir modeldir, kelimelerin etiketle bağımsız olduğunu varsayar.
- Metin sınıflandırma gibi veri boyutunun yüksek olduğu problemler için hızlı ve etkilidir. Hızlı ve etkili bir model olduğu için ve dersten de aşağı olduğumuz bir model olduğu için seçtim.

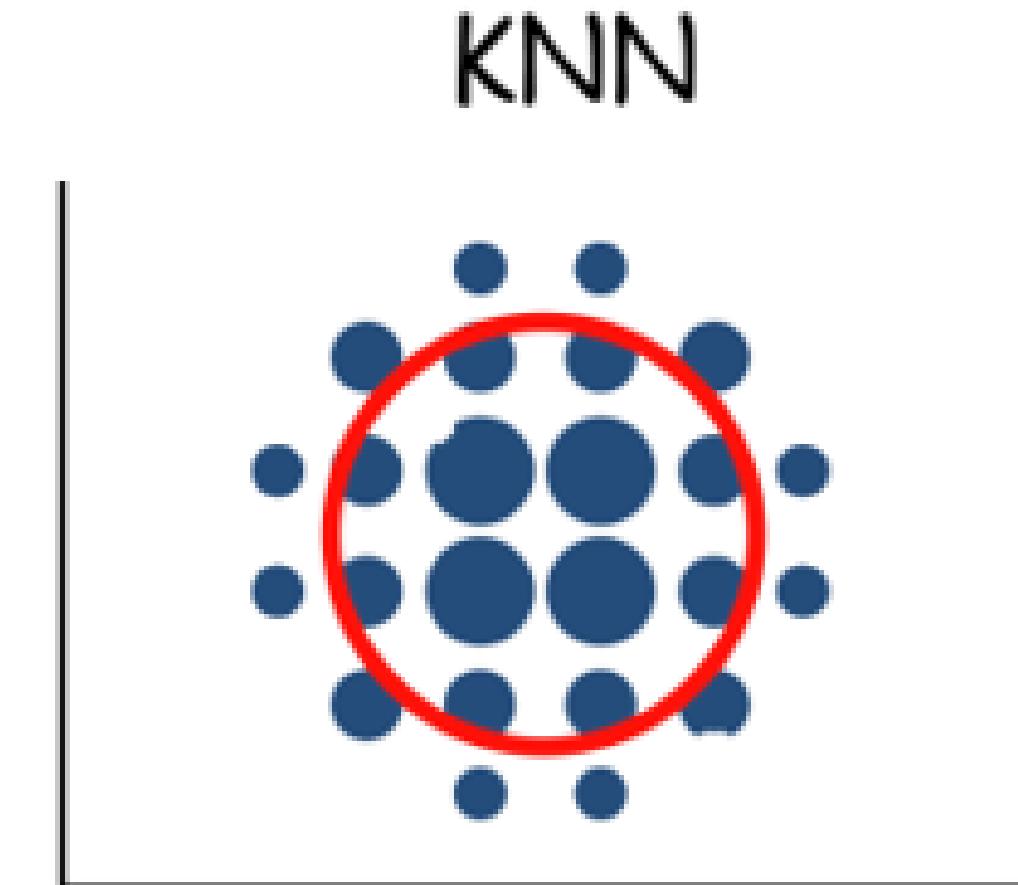
Naive Bayes



# Modelleme

## KNN (K-Nearest Neighbors).

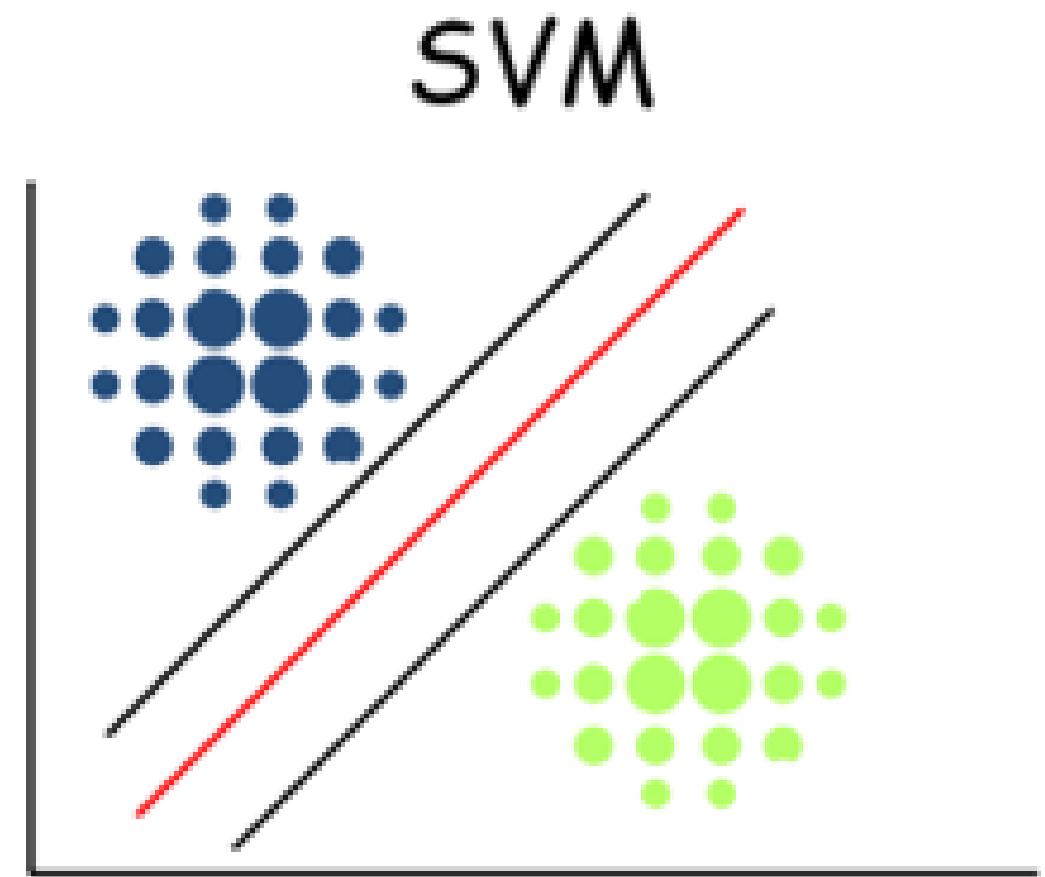
- Yeni bir veriyi sınıflandırmak için en yakın komşularına (örneklerine) bakar.
- Basit ama büyük veri setlerinde yavaş olabilir çünkü tüm veriyi bellekte tutar.



# Modelleme

## SVM (Support Vector Machine)

- Verileri en iyi ayıran düzlemi (vektörü) bulmaya çalışır.
- Yüksek boyutlu uzaylarda etkili çalışır ve genellikle yüksek doğruluk verir.

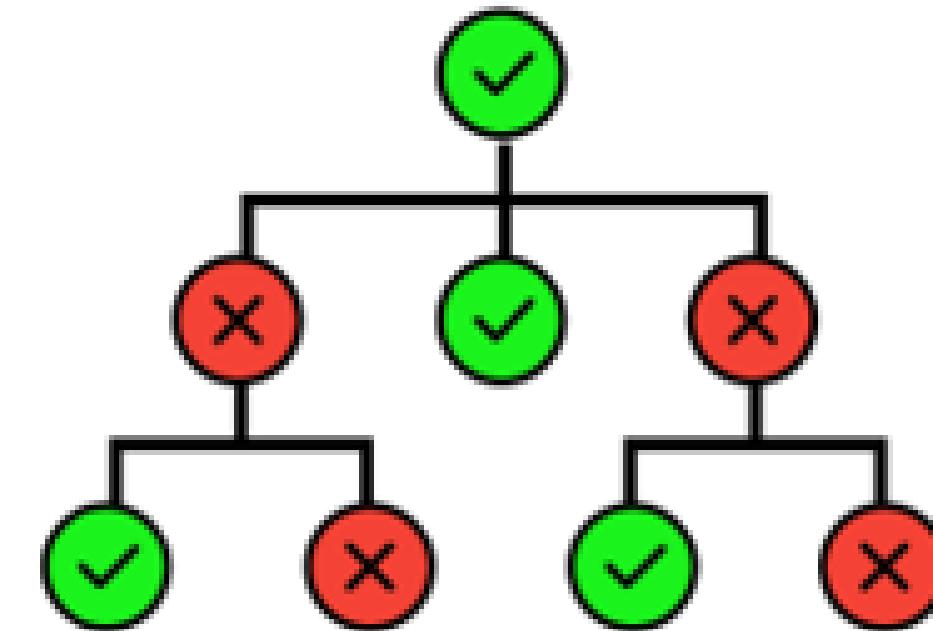


# Modelleme

## Random Forest

- Birden fazla karar ağacının çoğunluk oyu ile karar verdiği topluluk (ensemble) modelidir.
- Aşırı öğrenmeye karşı dayanıklıdır ve genelde dengeli sonuçlar verir.

## Random Forest



# Model Metrikleri

## Neden önemli?

Birden fazla makine öğrenmesi modelinin tahmin aşamasında karşılaştırma da kullanılarak en etkili modeli seçmesinde yardımcı olur.



### Accuracy (Doğruluk)

Modelin doğruluk oranını gösterir

### Precision (Kesinlik)

Modelin doğru dediği tahminler ne kadar doğru?

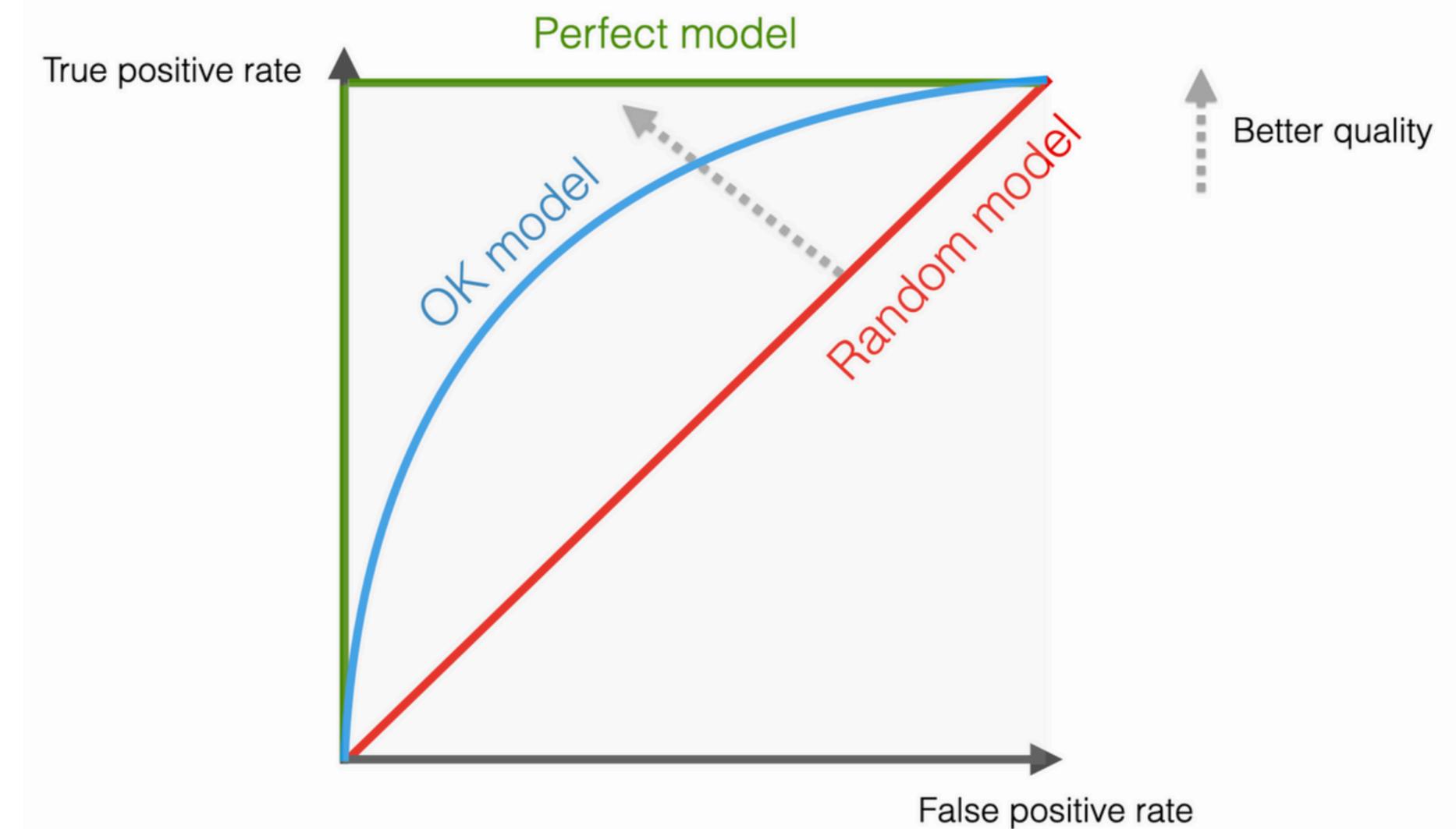
### Recal (Duyarlılık)

Mesela doğruların kaçta kaçını bildik

### F1 Score

Kesinlik ve Doğruluk arasında ki metriktir.

 En iyi model: RFE + SVM  
Doğruluk: %82.71  
F1 Score: %83.25  
Precision: %84.47  
Recall: %82.71

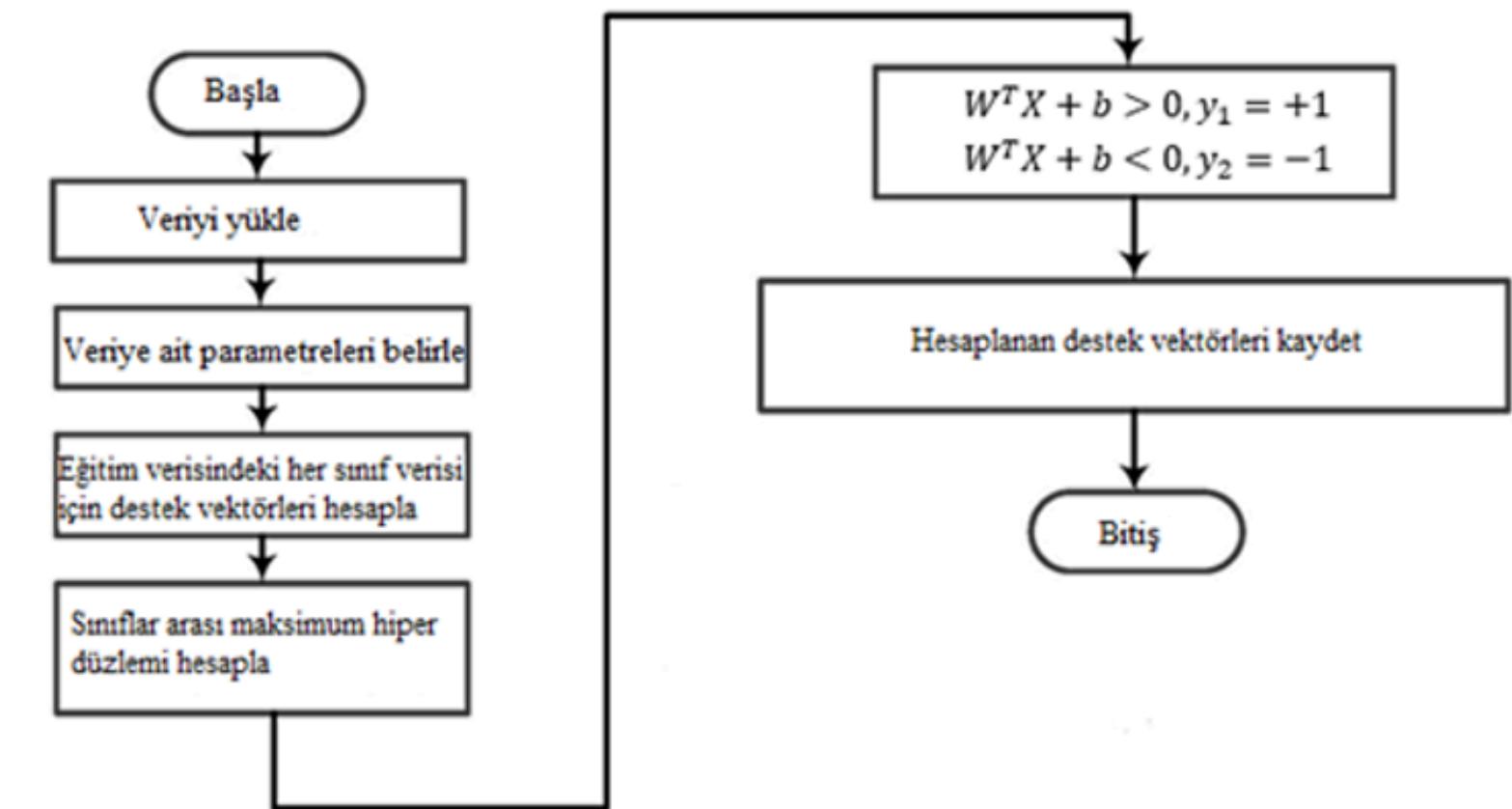


# Neden bu modeli seçtim?

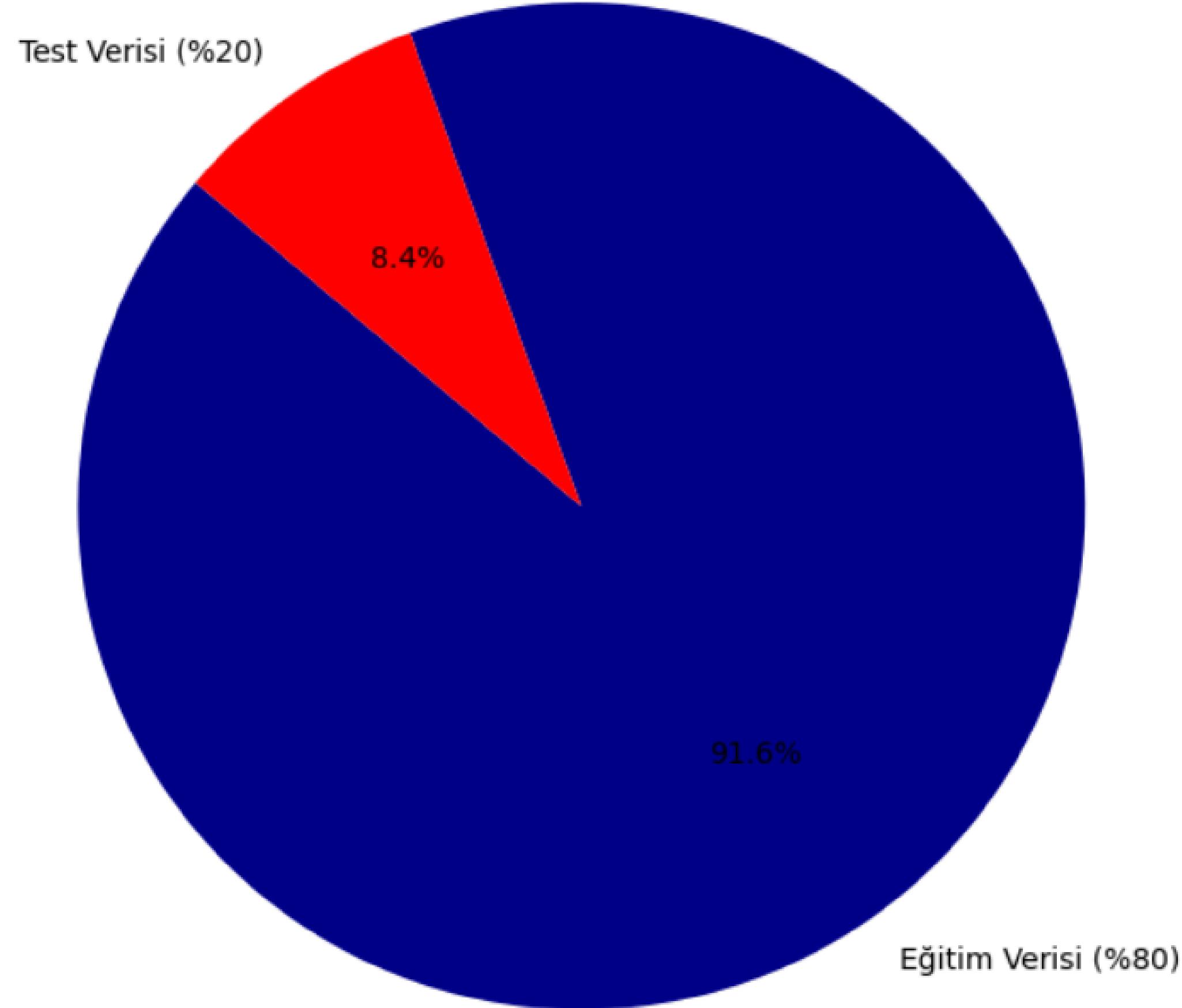
SVM (Support Vector Machine) modelinin bu proje de kullanmamın en büyük etkenleri

1. Yüksek Doğruluk oranına sahip olması.
2. Overfit'e karşı dayanıklı.
3. Gürültüye karşı dayanıklı.
4. Bellek verimliliği.
5. Sınıf dengesizliğine karşı dayanıklı.
6. Hesaplama daha kısa.
7. Modeli daha kolay yorumlarız.
8. Yeni verileri daha kolay güncelleriz.

Bu özellikler, SVM modelinin yemek tarifi sınıflandırma problemi için en uygun model olmasını sağlamıştır. Özellikle yüksek doğruluk oranı ve gürültüye karşı dayanıklılığı, bu projede tercih edilmesinin temel nedenleridir.



### Veri Seti Dağılımı (Toplam %100)





# Yemek Malzemelerine Göre Ülke Tahmini



Kullanılan modeller: Naive Bayes, SVM, Random Forest, KNN

En iyi kombinasyon: RFE + SVM

Model Metrikleri:

- Doğruluk: %82.74
- F1 Score: %83.29
- Precision: %84.52
- Recall: %82.74

Yemeğin malzemelerini gir (örn: garlic, onion, soy sauce)

Tahmin Et

localhost arayüz

 SPANISH mutfağına ait olduğunu tahmin ediyoruz.

 Bu tahmin RFE + SVM kombinasyonuyla yapılmıştır.



## Malzemelerin Geçtiği Mutfaklar

	Mutfak	Eşleşme	%
0	italian	403	56.05
1	southern_us	200	27.82
2	french	50	6.95
3	japanese	50	6.95
4	spanish	16	2.23



## En Çok Eşleşen Örnek

```
▼ {  
    "ID" : 1420  
    "Mutfak" : "italian"  
    ▼ "Malzemeler" : [  
        0 : "italian seasoning"  
        1 : "broiler-fryer chicken"  
        2 : "mayonaise"  
        3 : "zesty italian dressing"  
    ]  
    "Eşlesen Malzeme Sayısı" : 4  
}
```



“  
**GELECEK,  
HAYALLERİNİN  
GÜZELLİĞİNE  
İNANLARINDIR.**

— Eleanor Roosevelt