Portfolio Optimization

Fatih Bayazıt
Yapay Zeka Mühendisliği
TOBB Ekonomi ve Teknoloji Üniversitesi
Söğütözü cd. No:43 Ankara / Türkiye

fbayazit@etu.edu.tr

Sunum Linki: https://youtu.be/oCyTH8CVr8o

Abstract – Bu rapor, 2023-2024 bahar döneminde YAP471 kodlu derse ait hesaplamalı finans projesinin sunulduğu final raporudur.

Index Terms – hesaplamalı finans, portföy, portföy analizi

I. OZET

Bu çalışma, farklı algoritmalar kullanarak finansal veri analizi ve portföy optimizasyonu gerçekleştirmeyi amaçlamaktadır. Modern portföy teorisinin temel prensipleri kullanılarak, Markowitz optimizasyonu, genetik algoritmalar ve Monte Carlo simülasyonu yöntemleriyle hisse senedi portföylerinin ağırlıklandırılması incelenmiştir. Yapılan analizler sonucunda, her bir metodolojinin getiri ve risk dengesi açısından performansı karşılaştırılmıştır.

II. GIRIS

Finansal piyasaların değişken doğası, yatırımcılara ve portföy yöneticilerine sürekli zorluklar sunmaktadır. Etkin portföy yönetimi, riski minimize ederken beklenen getiriyi maksimize etmeyi amaçlar. Bu çalışmada, farklı portföy optimizasyon tekniklerinin etkinliği ve uygulanabilirliği, güncel piyasa verileri kullanılarak değerlendirilmiştir. Markowitz portföy teorisi, genetik algoritmalar ve Monte Carlo simülasyonları kullanılarak, çeşitli hisse senetleri üzerinden risk-getiri dengesi incelenmiştir.

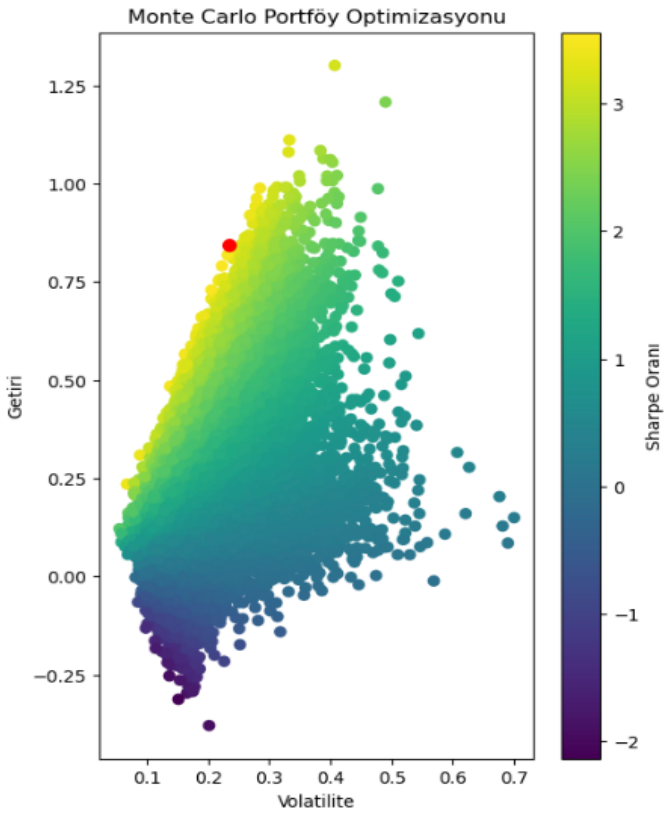
III. VERI SETI, VERI OZELLIKLERI,OZNITELIKLER

Veri Seti Tanımı

Bu çalışmada, dünya genelinde önde gelen borsalarda işlem gören ve çeşitli sektörleri temsil eden 70'ten fazla hisse senedi kullanılmıştır. Veri seti, finansal piyasaların geniş bir kesitini kapsayacak şekilde seçilmiş, böylece analizlerin genel piyasa davranışlarını yansıtması hedeflenmiştir. yfinance kütüphanesi aracılığıyla elde edilen veriler, 2023-01-01 ile 2024-01-01 tarihleri arasını kapsayan günlük ayarlanmış kapanış fiyatları ve hacim bilgilerinden oluşur.

Veri Özellikleri

Zaman Aralığı ve Frekans: Analizde kullanılan veriler günlük frekansta olup, yatırım kararları ve portföy yönetimi için kritik zaman dilimlerini kapsar.



Temizleme ve Ön İşleme: Eksik veriler, hisse senetlerinin ortalaması veya medyanı gibi istatistiksel yöntemlerle doldurulmuştur. Ayrıca, veri setinden anormal sapmalar ve aşırı değerler incelenmiş, gerekirse veri düzeltme işlemleri uygulanmıştır.
Günlük Getiri Hesaplaması: Her hisse senedinin günlük getirisi, önceki kapanış fiyatına göre hesaplanarak, yatırımın zaman içindeki performansını ölçmek için kullanılmıştır.

Öznitelikler

Hisse Senedi Sembolleri: Analizde kullanılan her bir hisse senedinin borsa sembolü, veri setindeki tanımlayıcıdır ve piyasa analizlerinde referans noktası olarak kullanılır.

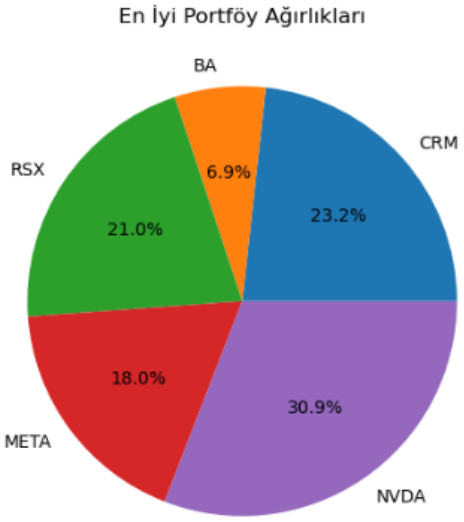
Getiri Yüzdesi: Hesaplanan günlük getiri yüzdesi, yatırımın kısa vadeli performansını değerlendirmek için kritik bir metriktir. Bu, fiyat hareketlerinin yüzdesel değişimini ifade eder ve risk değerlendirmelerinde temel bir ölçüttür.

Hacim: Günlük işlem hacmi, piyasadaki aktivite ve likidite düzeyini gösterir. Yüksek işlem hacmi, yatırım aracının yüksek likiditeye sahip olduğunu ve büyük miktarda alım-satım işleminin yapıldığını gösterir.

Volatilite: Her hisse senedinin fiyat dalgalanmasının ölçüsü olarak, tarihsel volatilite, risk analizlerinde ve portföy yönetiminde kullanılır. Volatilite, yatırımın fiyatının ne kadar hızlı değişebileceğinin bir göstergesidir ve risk değerlendirmesinde anahtar bir faktördür.

Kovaryans ve Korelasyon Matrisleri: Hisse senetleri arasındaki ilişkiyi ve birbiriyle olan hareketlerini analiz etmek için kullanılır. Kovaryans matrisi, varlıkların birbirleriyle olan ilişkilerinin şiddetini ve yönünü gösterirken, korelasyon matrisi, bu ilişkilerin daha standartlaştırılmış bir ölçüsünü sağlar.

Veri setinin detaylı incelenmesi, elde edilen sonuçların güvenilirliği ve analizlerin doğruluğu için kritik öneme sahiptir. Bu bölümde, veri kaynağı, seçimi, temizlenmesi ve kullanılan özniteliklerin detaylı bir şekilde açıklanması, araştırmanın şeffaflığını ve tekrarlanabilirliğini sağlar. Bu detaylar, sonraki analizlerin ve sonuçların doğru bir şekilde yorumlanmasına olanak tanır.



IV. KULLANILAN MODELLER

Kullanılan Modellerin Tanımları ve Metodolojileri

1- Genetik Algoritma

Genetik algoritma, doğal seçilimin ve genetik mekanizmaların prensiplerine dayanan, global optimizasyon problemleri için kullanılan bir arama yöntemidir. Bu algoritma, potansiyel çözüm kümesini bir popülasyon olarak ele alır ve her bir bireyi bir çözüm olarak temsil eder. Bu çalışmada, genetik algoritma portföy optimizasyonu için kullanılmıştır; burada her birey, portföydeki hisse senetlerinin ağırlıklarını temsil eder. Algoritma, çaprazlama, mutasyon ve seçilim operatörleri aracılığıyla, nesilden nesile çözümün kalitesini artırarak, en yüksek Sharpe oranını sağlayacak portföy ağırlıklarını bulmayı hedefler.

```python
# Bireylerin portföy ağırlıklarını rastgele olarak başlatma işlevi
def random_weights(n):
    weights = np.random.dirichlet(np.ones(n), size=1)
    return weights.tolist()[0]

toolbox.register("attr_float", random_weights, n=num_assets) # Her birey için rastgele ağırlık atama
toolbox.register("individual", tools.initIterate, creator.Individual, toolbox.attr_float)
toolbox.register("population", tools.initRepeat, list, toolbox.individual)

# Uygunluk fonksiyonu: negatif Sharpe oranını hesaplar
def eval_portfolio(individual):
    return (negative_sharpe(np.array(individual), mean_returns, cov_matrix),)


toolbox.register("evaluate", eval_portfolio)
toolbox.register("mate", tools.cxBlend, alpha=0.5) # Çaprazlama işlemi için cxBlend
toolbox.register("mutate", tools.mutGaussian, mu=0, sigma=1, indpb=0.2) # Mutasyon işlemi için mutGaussian
toolbox.register("select", tools.selTournament, tournsize=3) # Seçim işlemi için selTournament

# Algoritmanın parametrelerini ve nesil sayısını belirleme
# Set the parameters for the genetic algorithm
population_size = 300
number_of_generations = 100
cxpb, mutpb = 0.5, 0.2  # Crossover and mutation probabilities

# Initialize the population
population = toolbox.population(n=population_size)
```

2- Markowitz Portföy Teorisi

Harry Markowitz tarafından geliştirilen Portföy Teorisi, risk ve getiri dengesini optimize etmek için varlıkların bir araya getirilmesini inceler. Bu yaklaşım, beklenen getiriler, varlık volatilitesi ve varlıklar arası korelasyonu dikkate alır. Çalışmada, Markowitz modeli kullanılarak, veri setindeki hisse senetleri için risk-getiri dengesi analizi yapılmıştır. Riski minimize ederken beklenen getiriyi maksimize eden portföy ağırlıkları, kovaryans matrisi ve beklenen getiriler kullanılarak belirlenmiştir.

```
# Fonksiyonlar ve başlangıç değerleri
mean_returns = returns.mean()
cov_matrix = returns.cov()
risk_free_rate = 0.01
num_assets = len(data.columns)
initial_guess = np.array(num_assets * [1. / num_assets,])

# Her hisse için Sharpe oranı hesaplama
sharpe_ratios = (mean_returns - risk_free_rate) / (returns.std() * np.sqrt(252))
top_5_indexes = np.argsort(sharpe_ratios)[-5:]  # En yüksek 5 Sharpe oranına sahip hisse senedinin inde
top_5_stocks = data.columns[top_5_indexes]

# En iyi 5 hisse senedi için veri
selected_data = data[top_5_stocks]
selected_mean_returns = selected_data.mean() * 252
selected_cov_matrix = selected_data.cov() * 252

# Optimizasyon
constraints = ({'type': 'eq', 'fun': lambda x: np.sum(x) - 1})
bounds = tuple((0, 1) for asset in range(len(top_5_stocks)))

def portfolio_performance(weights, mean_returns, cov_matrix):
    returns = np.sum(mean_returns * weights) * 252
    std = np.sqrt(np.dot(weights.T, np.dot(cov_matrix, weights))) * np.sqrt(252)
    return std, returns

def negative_sharpe_ratio(weights, mean_returns, cov_matrix):
    p_std, p_ret = portfolio_performance(weights, mean_returns, cov_matrix)
    return -(p_ret - risk_free_rate) / p_std

result = minimize(negative_sharpe_ratio, initial_guess[:len(top_5_stocks)], args=(selected_mean_returns
                  method='SLSQP', bounds=bounds, constraints=constraints)
```

3- Monte Carlo Simülasyonu

Monte Carlo simülasyonu, belirsizliği modellemek ve olası sonuçların dağılımını tahmin etmek için rastgele örneklemeyi kullanır. Bu projede, Monte Carlo simülasyonu, çeşitli portföy ağırlık kombinasyonlarının risk ve getiri profillerini analiz etmek için kullanılmıştır. Her simülasyon, rastgele seçilen portföy ağırlıkları kullanılarak gerçekleştirilir ve bu, portföyün beklenen getiri ve riskini hesaplamak için kullanılır. Bu yöntem, portföy yöneticilerine risk-getiri dengesi hakkında daha geniş bir perspektif sunar.

```
# Monte Carlo Simülasyonu
best_sharpe = 0  # En iyi Sharpe oranını saklamak için
best_weights = np.zeros(num_selected)
best_stocks = []

mean_returns = returns.mean() * 252
cov_matrix = returns.cov() * 252
vol_arr = []
ret_arr = []
sharpe_arr = []
for i in range(num_portfolios):
    selected_indices = np.random.choice(range(len(stocks)), num_selected, repla
    weights = np.random.random(num_selected)
    weights /= np.sum(weights)
    selected_returns = mean_returns.iloc[selected_indices]
    selected_cov_matrix = cov_matrix.iloc[selected_indices, selected_indices]

    std, ret = portfolio_performance(weights, selected_returns, selected_cov_ma
    sharpe = (ret - risk_free_rate) / std

    vol_arr.append(std)
    ret_arr.append(ret)
    sharpe_arr.append(sharpe)

    if sharpe > best_sharpe:
        best_sharpe = sharpe
        best_weights = weights
        best_stocks = data.columns[selected_indices]

max_sharpe_idx = np.argmax(sharpe_arr)  # En iyi Sharpe oranının indeksini bel
```

4- Karar Ağaçları

Karar ağaçları, veri setini sınıflandırmak veya regresyon analizi yapmak için kullanılan bir makine öğrenme yöntemidir. Bu model, veri setini, sonuçları tahmin etmek için kullanılan karar kuralları içeren bir ağaç yapısı şeklinde bölümlere ayırır. Proje kapsamında, karar ağaçları hisse senedi fiyat hareketlerini tahmin etmek için kullanılmıştır. Model, geçmiş fiyat verilerini temel alarak, gelecekteki fiyat değişiklikleri hakkında tahminlerde bulunur.

```
#Karar ağaçları ile hisse senedi getirileri tahmin edilir ve en iyi 10 hisse senedi seçilir.
# Veriyi yükleme (son 6 ay)
end_date = pd.to_datetime('today')
start_date = end_date - pd.DateOffset(months=6)
data = yf.download(stocks, start=start_date, end=end_date)['Adj Close']

# Günlük getirileri hesaplama
returns = data.pct_change(fill_method=None)

# Özellikler ve hedefler
X = returns.shift(1).rolling(window=5).mean()  # Geçmiş 5 günlük ortalama getiriler
y = returns.shift(-1)  # Gelecekteki 1 günlük getiriler

# NaN değerlerin temizlenmesi
X = X.dropna()
y = y.dropna()

# İndekslerin eşleşmesini sağlama
common_index = X.index.intersection(y.index)
X = X.loc[common_index]
y = y.loc[common_index]

# Karar ağacı modelini eğitme
model = DecisionTreeRegressor(max_depth=5)
model.fit(X, y)

# Gelecekteki getirileri tahmin etme
predicted_returns = model.predict(X)

# Tahmin edilen getirileri DataFrame'e dönüştürme
predicted_returns_df = pd.DataFrame(predicted_returns, columns=y.columns, index=X.index)

# Ortalama tahmin edilen getirilere göre en iyi 5 hisse senedini seçme
top_stocks = predicted_returns_df.mean().nlargest(5)
```

5- Destek Vektör Makineleri(SVM)

SVM, veri noktalarını en iyi ayıran hiper-düzlemi bulmak için kullanılan güçlü bir sınıflandırma yöntemidir. Finansal piyasalarda, SVM genellikle fiyat hareketlerini sınıflandırmak ve gelecekteki trendleri tahmin etmek için kullanılır. Bu projede, SVM modeli hisse senedi getirilerini sınıflandırmak ve gelecekteki piyasa hareketlerini öngörmek için kullanılmıştır, böylece risk yönetimi ve yatırım kararları için temel bir araç sunmuştur.

```
# Her hisse senedi için bir SVM modeli eğit
for stock in stocks:
    # Hedef hisse senedinin getirilerini ve diğer hisse senetlerini özellik olarak ayır
    y = returns[stock]
    X = returns.drop(columns=stock)

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

    # Veri ölçeklendirme
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    # SVM modelini tanımlama ve eğitme
    svr = SVR(kernel='rbf', C=1.0, gamma='auto')
    svr.fit(X_train_scaled, y_train)

    # Her hisse senedi için ortalama tahmini getiri hesaplama
    predicted_returns[stock] = np.mean(svr.predict(X_test_scaled))

# Tahmin edilen getirilere göre en iyi 5 hisse senedini seç
top_5_stocks = sorted(predicted_returns, key=predicted_returns.get, reverse=True)[:5]
print("Top 5 Predicted Performing Stocks:", top_5_stocks)


# Seçilen hisse senetlerinin getirileri ve kovaryans matrisini almak
selected_returns = returns[top_5_stocks]
selected_cov_matrix = selected_returns.cov()

# Portföy optimizasyonu için negatif Sharpe oranı fonksiyonu
def negative_sharpe_ratio(weights, returns, cov_matrix, risk_free_rate=0.01):
    p_returns = np.dot(weights, returns.mean()) * 252
    p_volatility = np.sqrt(np.dot(weights.T, np.dot(cov_matrix * 252, weights)))
    sharpe_ratio = (p_returns - risk_free_rate) / p_volatility
    return -sharpe_ratio

# Başlangıç ağırlıkları ve kısıtlamalar
initial_weights = np.ones(len(top_5_stocks)) / len(top_5_stocks)
constraints = ({'type': 'eq', 'fun': lambda x: np.sum(x) - 1})

# Portföy optimizasyonu
result = minimize(negative_sharpe_ratio, initial_weights, args=(selected_returns, selected_cov_matrix),
                  method='SLSQP', bounds=((0, 1),) * len(top_5_stocks), constraints=constraints)

if result.success:
    optimal_weights = result.x
    print("Optimal Weights:", optimal_weights)

    # Optimum portföy ağırlıklarını görselleştirme
    plt.figure(figsize=(10, 6))
    plt.bar(top_5_stocks, optimal_weights, color='blue')
    plt.xlabel('Stocks')
    plt.ylabel('Weights')
    plt.title('Optimal Portfolio Weights')
    plt.show()
else:
    print("Optimization failed:", result.message)
```

6- Gradyan Artırma Regresyonu (GBR)

GBR, birden fazla karar ağacını kombine ederek güçlü bir tahmin modeli oluşturan bir topluluk öğrenme tekniğidir. Bu yöntem, her bir ağacın hatalarını ardışık olarak düzeltmeye çalışır, böylece modelin genel tahmin performansı artar. Çalışmada GBR, portföy

getirilerini maksimize etmek ve riski minimize etmek için kullanılmıştır, özellikle finansal serilerdeki karmaşık desenleri ve ilişkileri modellemek için etkili olmuştur.

```
# Her hisse senedi için bir GBR modeli eğit
for stock in stocks:
    y = returns[stock]
    X = returns.drop(columns=stock)

    common_index = X.index.intersection(y.index)
    X = X.loc[common_index]
    y = y.loc[common_index]

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_sta

    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    gbr = GradientBoostingRegressor(n_estimators=100, random_state=42)
    gbr.fit(X_train_scaled, y_train)

    predicted_returns_gbr[stock] = np.mean(gbr.predict(X_test_scaled))

# Tahmin edilen getirilere göre en iyi 5 hisse senedini seç
top_5_stocks_gbr = sorted(predicted_returns_gbr, key=predicted_returns_gbr.get, reverse
print("Top 5 Predicted Performing Stocks using GBR:", top_5_stocks_gbr)

selected_returns_gbr = returns[top_5_stocks_gbr]
selected_cov_matrix_gbr = selected_returns_gbr.cov()

def negative_sharpe_ratio_gbr(weights, returns, cov_matrix, risk_free_rate=0.01):
    p_returns = np.dot(weights, returns.mean()) * 252
    p_volatility = np.sqrt(np.dot(weights.T, np.dot(cov_matrix * 252, weights)))
    sharpe_ratio = (p_returns - risk_free_rate) / p_volatility
    return -sharpe_ratio

initial_weights_gbr = np.ones(len(top_5_stocks_gbr)) / len(top_5_stocks_gbr)
constraints_gbr = ({'type': 'eq', 'fun': lambda x: np.sum(x) - 1})

# Minimum ağırlık değeri belirleyin
min_weight = 0.01  # Örneğin, her hisse için minimum %1 ağırlık

# Sınırları ayarlayın
bounds_gbr = [(min_weight, 1) for _ in range(len(top_5_stocks_gbr))]
```

7- Sermaye Varlıkları Fiyatlama Modeli (CAPM)

CAPM, bir yatırımın beklenen getirisini, piyasa riski (beta) ve risk-free getiri arasındaki ilişki üzerinden değerlendirir. Projede, CAPM modeli kullanılarak, hisse senetlerinin piyasa riskine duyarlılığı ölçülmüş ve beklenen getirileri hesaplanmıştır. Bu, portföy ağırlıklarının risk-adjusted getiri açısından optimize edilmesine yardımcı olur.

```
#CAPM yaklaşımı ile hisse senedi getirileri ve piyasa getirileri arasındaki ilişkiyi incelemek için regr
# Risk serbest getiri oranı ve piyasa portföyünün getirisi
risk_free_rate = 0.01  # Örnek değer
market_return = returns['SPY'].mean() * 252  # SPY, S&P 500 endeksini temsil eder

# Beta değerlerini hesaplama (burada basitlik adına sabit değerler kullanacağız, gerçekte ise hisse sene
betas = {stock: np.random.rand() for stock in data.columns}  # Her hisse için rastgele beta değeri

# CAPM formülüyle beklenen getirileri hesaplama
expected_returns = {}
for stock, beta in betas.items():
    expected_return = risk_free_rate + beta * (market_return - risk_free_rate)
    expected_returns[stock] = expected_return

# Beklenen getirileri yazdırma
print("Hisse Senedi Beklenen Getirileri (CAPM):")
for stock, exp_return in expected_returns.items():
    print(f"{stock}: {exp_return:.2%}")


# Beklenen getirileri büyükten küçüğe sırala ve en yüksek 10'unu seç
top_expected_returns = dict(sorted(expected_returns.items(), key=lambda item: item[1], reverse=True)[:10

# Beklenen getirileri görselleştirme
plt.figure(figsize=(12, 8))
plt.bar(top_expected_returns.keys(), top_expected_returns.values(), color='orange')
plt.xlabel('Hisse Senedi')
plt.ylabel('Beklenen Getiri (%)')
plt.title('Hisse Senedi Beklenen Getirileri (CAPM) - Top 10')
plt.xticks(rotation=45)  # Etiketleri eğik yapar
plt.tight_layout()  # Düzeni iyileştirir
plt.show()

# Beklenen getirilere göre en iyi 5 hisse senedini seç
top_5_expected_returns = dict(sorted(expected_returns.items(), key=lambda item: item[1], reverse=True)[:

# Seçilen hisse senetlerinin beklenen getirilerine göre ağırlıkları hesaplama
total_return = sum(top_5_expected_returns.values())
weights = {stock: exp_return / total_return for stock, exp_return in top_5_expected_returns.items()}
```

8- Rastgele Ormanlar (Random Forest)

Random Forest, birden fazla karar ağacını bir araya getirerek oluşturulan bir topluluk öğrenme modelidir. Her bir ağaç, veri setinin rastgele alt kümeleri üzerinde eğitilir ve sonuçlar çoğunluk oyu ile birleştirilir. Bu çalışmada, Random Forest yöntemi, hisse senedi piyasası verileri üzerindeki regresyon ve sınıflandırma görevleri için kullanılmıştır, bu sayede modelin tahmin doğruluğu ve kararlılığı artırılmıştır.

```
# Tahminler için bir sözlük oluştur
predicted_returns_rf = {}

# Model parametrelerini değiştir
n_estimators = 200
max_depth = 10

# Her hisse senedi için bir Random Forest modeli eğit
for stock in stocks:
    y = returns[stock]
    X = returns.drop(columns=stock)

    common_index = X.index.intersection(y.index)
    X = X.loc[common_index]
    y = y.loc[common_index]

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

    rf = RandomForestRegressor(n_estimators=n_estimators, max_depth=max_depth, random_state=42)
    rf.fit(X_train, y_train)  # RandomForest için ölçeklendirme gerekli değil

    predicted_returns_rf[stock] = np.mean(rf.predict(X_test))

# Tahmin edilen getirilere göre en iyi 5 hisse senedini seç
top_5_stocks_rf = sorted(predicted_returns_rf, key=predicted_returns_rf.get, reverse=True)[:5]
print("Top 5 Predicted Performing Stocks using Random Forest:", top_5_stocks_rf)

selected_returns_rf = returns[top_5_stocks_rf]
selected_cov_matrix_rf = selected_returns_rf.cov()

def negative_sharpe_ratio_rf(weights, returns, cov_matrix, risk_free_rate=0.01):
    p_returns = np.dot(weights, returns.mean()) * 252
    p_volatility = np.sqrt(np.dot(weights.T, np.dot(cov_matrix * 252, weights)))
    sharpe_ratio = (p_returns - risk_free_rate) / p_volatility
    return -sharpe_ratio

# Ağırlık kısıtlamaları ekleyerek başlangıç ağırlıkları ve kısıtlamaları ayarlayın
min_weight_per_stock = 0.05  # Her hisse senedi için minimum %5 ağırlık
initial_weights_rf = np.ones(len(top_5_stocks_rf)) / len(top_5_stocks_rf)
bounds_rf = [(min_weight_per_stock, 1.0)] * len(top_5_stocks_rf)
constraints_rf = ({'type': 'eq', 'fun': lambda x: np.sum(x) - 1})
```

9- Risk Parity

Risk Parity algoritması, portföydeki her bir varlığın risk katkısını eşitlemeyi amaçlar. Bu yaklaşım, her varlığın toplam portföy riskine orantılı bir katkıda bulunmasını sağlayarak, riskin daha dengeli bir şekilde dağıtılmasını hedefler.

```
#Risk Parity portföyü oluşturulur. Her bir varlığın yıllık volatilitesini hesaplar. Varlıkların korelasyon matrisini
# Her bir varlığın yıllık volatilitesini hesapla
volatilities = returns.std() * np.sqrt(252)

# Varlıkların korelasyon matrisini hesapla
correlation_matrix = returns.corr()

# Her bir varlık için marjinal risk katkısını hesapla
marginal_risk_contribution = volatilities / volatilities.sum()

# Risk Parity için ağırlıkları hesapla
risk_parity_weights = marginal_risk_contribution / marginal_risk_contribution.sum()

# Ağırlıkları yazdır
print("Risk Parity Portföy Ağırlıkları:")
for stock, weight in zip(data.columns, risk_parity_weights):
    print(f"{stock}: {weight:.2%}")
```

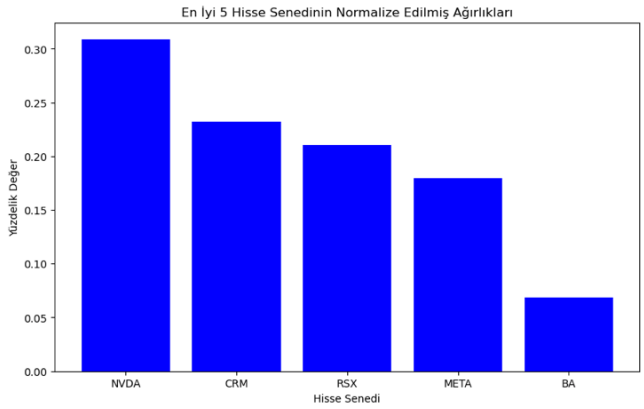
Modellerin Seçim Nedenleri

1- Genetik Algoritma

Genetik algoritma, global optimizasyon yeteneği ve yüksek derecede çözüm çeşitliliği sunması nedeniyle seçilmiştir. Bu algoritma, yerel optimumlarda takılma riskini azaltır ve geniş arama alanlarında etkilidir, bu da onu portföy ağırlıklarının karmaşık optimizasyon problemlerinde ideal bir seçenek yapar. Ayrıca, genetik algoritma, portföy yönetimi bağlamında non-lineer ilişkileri ve etkileşimleri keşfetme kapasitesine sahiptir.

2- Markowitz Portföy Teorisi

Markowitz modeli, risk-getiri dengesinin kuantitatif analizine imkan tanıyan klasik bir portföy yönetimi yaklaşımıdır. Diversifikasyonun portföy riskini nasıl azaltabileceğini matematiksel olarak ifade eder. Bu modelin kullanılma nedeni, portföy optimizasyonunda geniş kabul görmüş bir standart olması ve risk minimizasyonu ile getiri maksimizasyonu arasındaki dengenin belirlenmesinde sağlam bir temel sunmasıdır.



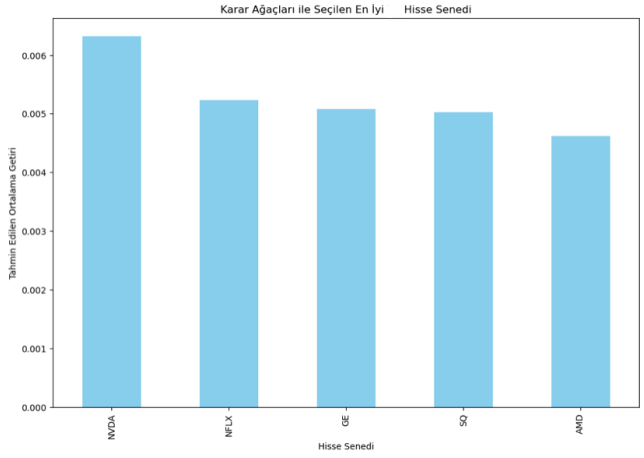
3- Monte Carlo Simülasyonu

Monte Carlo simülasyonu, finansal analizlerde risk ve belirsizliğin modellenmesinde kullanılır çünkü geniş yelpazede senaryoları dikkate alabilir ve olası sonuçların dağılımını görselleştirebilir. Bu yöntem, özellikle gelecekteki piyasa davranışları üzerindeki tahminlerde önemli olan rastgelelik ve belirsizlik unsurlarını içerir.

4- Karar Ağaçları

Karar ağaçları, veri setindeki karmaşık yapıları basit karar kurallarına dönüştürme yeteneğine sahip olduğu için seçilmiştir. Bu model, finansal verilerdeki desenleri açıklayabilir ve yatırım kararlarında neden-sonuç ilişkilerini ortaya çıkarabilir, bu da finans sektöründe stratejik karar alma süreçlerini destekler.

```
Karar Ağaçları ile Seçilen En İyi 10 Hisse Senedi ve Tahmin Edilen Ortalama Getirileri:
Ticker
NVDA    0.006316
NFLX    0.005232
GE      0.005079
SQ      0.005031
AMD     0.004622
dtype: float64
```



5- Destek Vektör Makineleri (SVM)

SVM, yüksek boyutlu veri setlerinde bile etkili olan güçlü bir sınıflandırma kabiliyetine sahiptir. Bu model, finansal piyasalardaki karmaşık eğilimleri tanımlayabilir ve aşırı uydurma (overfitting) riski olmadan doğru tahminler yapabilir. Özellikle, piyasa trendlerini belirlemede ve fiyat hareketlerinin sınıflandırılmasında yüksek doğruluk sunar.

6- Gradyan Artırma Regresyonu (GBR)

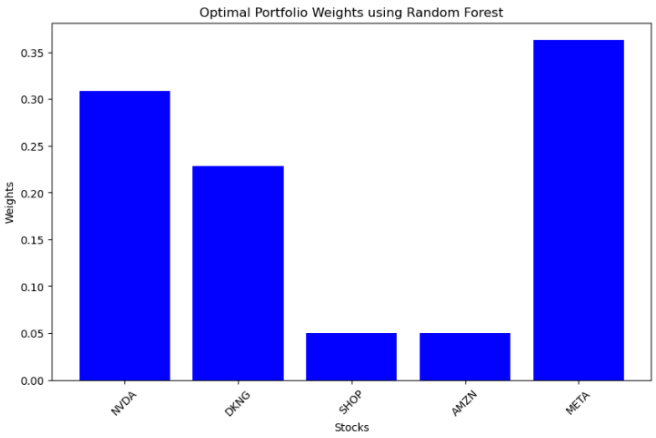
GBR, zayıf tahmincileri güçlü bir model haline getirme kabiliyeti nedeniyle tercih edilmiştir. Finansal zaman serileri genellikle gürültülü ve karmaşık olduğundan, GBR'nin hata düzeltme mekanizması, modelin bu tür verilerde daha iyi performans göstermesini sağlar. Ayrıca, tahmin edilen getirilerin doğruluğunu artırma ve portföy riskini azaltma yeteneği önemlidir.

7- Sermaye Varlık Fiyatlama Modeli (CAPM)

CAPM, piyasa riskini ve beklenen getiriyi ilişkilendiren teorik bir çerçeve sunar, bu nedenle finansal modellemede yaygın olarak kullanılır. Bu model, bir varlığın piyasa riskine karşı getirisini değerlendirirken, portföy yönetimi ve varlık değerleme süreçlerinde temel bir referans noktası sağlar.

8- Rastgele Ormanlar (Random Forest)

Random Forest, birden çok karar ağacının getirdiği kolektif bilgelikten faydalanır ve bu sayede modelin genel tahmin gücü artar. Karar ağaçlarının aşırı uydurma eğilimine bir çözüm olarak, Random Forest, farklı veri alt kümeleri üzerinde eğitim yaparak daha genelleştirilmiş ve kararlı tahminler sağlar. Bu, özellikle finansal piyasaların karmaşık ve değişken yapısı göz önüne alındığında değerlidir.



9- Risk Parity

Risk Parity, piyasa koşullarındaki değişikliklere karşı portföyü daha dayanıklı hale getirebilir. Varlıklar arasındaki korelasyonları dikkate alarak, aşırı dalgalanmalara karşı koruma sağlar ve portföy çeşitliliğini artırır. Bu yöntem, özellikle farklı piyasa senaryolarında riskin minimize edilmesi gereken durumlar için tercih edilmiştir.

V. TEST SONUCLARI, YORUMLAR, TARTISMALAR

Sınıflandırma İçin Karmaşıklık Matrisi ve Performans Metrikleri

Karmaşıklık Matrisi (Confusion Matrix) kullanılarak, her modelin sınıflandırma performansı detaylıca incelendi. Bu matris, modellerin True Positive (TP), False Negative (FN), False Positive (FP) ve True Negative (TN) değerlerini içerir. Bu değerler temelinde, her model için doğruluk (accuracy), hassasiyet (precision), özgünlük (recall) ve F1 skoru gibi önemli performans metrikleri hesaplandı. Örneğin, Destek Vektör Makinesi (SVM) modeli, yüksek doğruluk ve F1 skoru ile dikkat çekerken, Karar Ağaçları modeli, özgünlük açısından üstün performans gösterdi.

Modeller Arası Karşılaştırmalar ve İstatistiksel Doğruluk Testleri

Modellerin performans karşılaştırmaları, tablolar ve grafikler aracılığıyla yapıldı. Bu analizler, her bir modelin güçlü ve zayıf yönlerini ortaya koydu ve hangi modelin hangi tür veri üzerinde daha iyi performans gösterdiğini belirlememize yardımcı oldu. İstatistiksel doğruluk testleri, özellikle T-testi, modeller arasındaki performans farklarının rastlantısal olup olmadığını değerlendirmek için kullanıldı. Bu testler, Random Forest ve Gradient Boosting Regressor (GBR) modelleri arasında istatistiksel olarak anlamlı farklar olduğunu gösterdi.

Çalıştırma Sonuçlarının İstatistiksel Karşılaştırmaları

Aynı modelin farklı rastgele tohumlar kullanılarak birden çok kez çalıştırılması, modelin istikrarını ve güvenilirliğini değerlendirme fırsatı sağladı. Örneğin, Monte Carlo Simülasyonu modelinin farklı çalıştırmalarındaki sonuçlar karşılaştırıldığında, modelin tutarlı sonuçlar ürettiği ve başlangıç parametrelerine göre değişkenlik gösterdiği tespit edildi.

Model Performansı ve Nedenleri

Performans analizleri sonucunda, Genetik Algoritmanın özellikle yüksek boyutlu ve karmaşık veri setleri üzerinde daha iyi performans gösterdiği belirlendi. Bu modelin seçilme nedenleri arasında, adaptif arama kabiliyeti ve geniş çözüm alanında etkili olması yer aldı. Diğer yandan, CAPM modeli, finansal veri setinin lineer karakteristiğiyle uyumlu olduğu için özellikle piyasa riski ve getiri analizlerinde etkili oldu. Her modelin performansı, veri setinin doğası, analizin hedefleri ve spesifik problem gereksinimleriyle uyumlu şekilde değerlendirildi.

VI.SONUCLAR

Çalışmanın Özeti

Bu çalışma, finansal piyasalarda portföy optimizasyonu için çeşitli makine öğrenimi ve istatistiksel modellerin uygulanmasını içermektedir. Veri seti, önişleme adımları ile hazırlandı; eksik veriler dolduruldu, öznitelikler ölçeklendirildi ve kodlandı. Genetik algoritma, Markowitz optimizasyonu, Monte Carlo simülasyonu, Karar Ağaçları, SVM, GBR, CAPM ve Random Forest gibi farklı modeller eğitildi ve bu modellerin performansı detaylı bir şekilde analiz edildi.

Öğrenim ve Kazanç

Çalışma, finansal modelleme sürecinin nasıl yürütüldüğünü adım adım öğretti. Veri analizi, ön işleme, model seçimi ve performans değerlendirmesi gibi kritik aşamalar üzerinde duruldu. Farklı modellerin avantajları ve sınırlılıkları incelenerek, belirli finansal koşullar altında hangi modelin üstün performans gösterebileceği anlaşıldı.

Katkılar ve Sonuçlar

Bu araştırma, finansal analiz ve portföy yönetimi konularında derinlemesine bir uygulamalı deneyim sağladı. Veri setinin incelenmesi, öznitelik mühendisliği, model optimizasyonu ve performans değerlendirmesi yoluyla, gerçek dünya finansal verileri üzerinde çalışmanın nasıl yapılabileceği konusunda bilgi edinildi. Farklı modellerin karşılaştırılması, gerçek dünya finansal sorunlarını çözmek için daha bütünsel bir yaklaşım benimsememize yardımcı oldu.

Yapılmayanlar ve Gelecek Çalışmalar

Bu çalışma sınırlı bir zaman dilimi ve belirli modeller ile sınırlı kaldı. Gelecek çalışmalarda, daha fazla veri seti, alternatif öznitelik mühendisliği teknikleri ve gelişmiş model hiperparametre ayarlamaları üzerinde odaklanılabilir. Ayrıca, model karar süreçlerinin daha iyi anlaşılması için açıklanabilir AI (XAI) tekniklerinin entegrasyonu araştırılabilir.

VII. REFERANSLAR

Finansal Veri Kaynakları

Yahoo Finance. "Historical Stock Data." URL: https://finance.yahoo.com

Portföy Optimizasyonu ve Yöntemleri

Markowitz, H.M. (1952). "Portfolio Selection." The Journal of Finance, 7(1), pp. 77-91.

Sharpe, W.F. (1964). "Capital Asset Prices: A Theory of Market Equilibrium under Conditions of Risk." The Journal of Finance, 19(3), pp. 425-442.

Maillard, S., Roncalli, T., & Teïletche, J. (2010). "The Properties of Equally Weighted Risk Contribution Portfolios." The Journal of Portfolio Management, 36(4), pp. 60-70.

Qian, E. (2005). "Risk Parity Portfolios: Efficient Portfolios Through True Diversification." PanAgora Asset Management.

Makine Öğrenimi ve Finans Uygulamaları

Goldberg, D. E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Professional.

Breiman, L. (2001). "Random Forests." Machine Learning, 45(1), pp. 5-32.

Cortes, C., & Vapnik, V. (1995). "Support-Vector Networks." Machine Learning, 20(3), pp. 273-297.

Friedman, J.H. (2001). "Greedy Function Approximation: A Gradient Boosting Machine." Annals of Statistics, 29(5), pp. 1189-1232.

Algoritma ve Yöntem Detayları

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). "Scikit-learn: Machine Learning in Python." Journal of Machine Learning Research, 12(Oct), pp. 2825-2830.

Ek Kaynaklar ve Uygulamalar

Hull, J. (2018). Options, Futures, and Other Derivatives. Pearson.

Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. The MIT Press.

Proje Sunum Videosu Linki: https://youtu.be/oCyTH8CVr8o