**Makine Öğrenmesi Teknikleriyle Satrançta Son Oyun Durumu Analizi: KRKP-A7 Veri Seti Üzerine Çalışma**

**ÖZET**

Satrançta KRKP-A7 (Beyazlarda Şah-Piyon (piyon a7 karesinde) karşısında Siyahlarda Şah-Kale) konumunda iki taraf için de en iyi hamleler oynandığı takdirde beyazların, kendi piyonunu terfi edip oyun sonunda kazanıp kazanamayacağının Makine öğrenmesi algoritmalarıyla tahmin edilmesi amaçlanmaktadır. Elimizde bulunan veri setinin bir bölümü makine öğrenmesine, bir bölümü ise test etmesi için ayrılmıştır. Tahmini yapmak için; Bayes Yöntemi, Karar Ağacı Yöntemi, Rastgele Orman Yöntemi, Lojistik Regresyon Yöntemi, kNN Yönteminin kullanımına karar verilmiştir. Test verisi ve Eğitim verisinin oranları değiştirilip karşılaştırıldığında Bayes Yöntemi dışındaki diğer yöntemlerin doğruluk oranı hepsi için yüksek oranda çıktığı tespit edilmiştir. Ayrıca random\_state değeri 100 gibi bir sabit değer olarak tutularak olası eğitim verisi farklılıklarının önüne geçilmiştir. Bu yöntemler arasında en tutarlı olanının Karar Ağacı Yöntemi olduğu tespit edilmiştir. Bu duruma göre Karar Ağacı yönteminin bu veri seti için seçilmesinin en uygun olduğu kararına varılmıştır.

**Anahtar Kelimeler:** Makine Öğrenmesi, Satranç, Algoritma, Veri Bilimi, Stockfish

1. **Giriş**

Satranç gibi deterministik oyunlarda kuralların net olması, makine öğrenmesi uygulamaları için oldukça uygun bir alan sunar. Bu yüzden satranç, yapay zeka araştırmalarında uzun süredir kullanılmaktadır. Alen David Shapiro, bu alanda erken dönem çalışmalarıyla öne çıkan bir araştırmacıdır. 1983 yılında Edinburgh Üniversitesi’nde tamamladığı doktora tezinde, yapılandırılmış tümevarım (structured induction) adını verdiği bir yöntem geliştirerek satranç oyun sonları üzerinden sistemler tasarlamıştır.

Shapiro’nun bu çalışmaları, daha sonra 1987 yılında yayımladığı Structured Induction in Expert Systems adlı kitapta toplanmıştır. Kitapta ele alınan KRKPA7 veri seti (King-Rook vs. King-Pawn on a7), makine öğrenme algoritmalarının sınırlı ama anlamlı bir problem üzerinde nasıl karar verdiğini test etmek için hazırlanmıştır. Bu veri setinde, beyaz tarafın (şah ve piyon) siyah tarafa (şah ve kale) karşı kazanıp kazanamayacağı sınıflandırılmaya çalışılır.

Bu çalışmada da farklı makine öğrenme yöntemleri kullanılarak beyazın kazanıp kazanamayacağı tahmin edilmeye çalışılmıştır. KRKPA7 gibi veri setleri, hem algoritmaların başarısını değerlendirmek hem de karar sürecinde hangi özelliklerin etkili olduğunu anlamak açısından önemlidir. Shapiro’nun bu çalışmaları, yapay zeka sistemlerinin nasıl bilgi işleyip karar verdiklerine dair önemli katkılar sağlamıştır.

Shapiro’nun kitabından örnek vermek gerekirse:

Aşağıdaki pozisyonda beyaz “nowin” pozisyonundadır. Beyaz piyonun ilerlemesi halinde kale tarafından yenmektedir. Kaleyi alması durumda ise beraberlik söz konusudur.

metin, kalıp, desen, düzen, ekran görüntüsü, dikdörtgen içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.metin, ekran görüntüsü, kalıp, desen, düzen, dikdörtgen içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

1. **Materyal ve Yöntem**

**2.1. Veri Seti**

Çalışmada Kaliforniya Üniversitesi’nin sağladığı “Chess (King-Rook vs. King-Pawn)” başlıklı,

Siyah Şah ve Kale ile Beyaz Şah ve Piyon oyun sonu durumunu inceleyen veri seti kullanılmıştır. Veri seti 1’i sonuç değer olmak üzere 36 adet özellik içerir. Bu özellikler aşağıda verilmekte olup özetlemek gerekirse “Beyaz Şah tehdit altında mı, İkinci ve sekizinci sıralar boş mu” gibi kesin pozisyon belirtmeyen lakin oyun sonu tahmini yapabileceğimiz özelliklerdir. Verinin %20’si ve %30’u test için bölümlendirilmiştir. Burada amaç eğitim verisinin miktarının doğruluğa olan etkisini gözlemlemektir. Ayrıca sonuçları stabilize etmek için random\_state değeri 100 olarak belirlenmiştir.

Yöntem olarak, Karar Orman, Bayes Teoremi, Lineer Regresyon, Logical Regresyon, K-NN ve Karar Ağacı yöntemleri kullanılmıştır. Bu yöntemler aşağıda detaylıca açıklanmıştır.

Modelin doğruluğu, çeşitli metotlar ile ölçülmüş olup, her birinin doğruluk (accuracy), duyarlılık (recall), kesinlik (precision) ve F1-Score 3. bölümdeki tabloda belirtilmiştir.

**Özellikler listesi:**

**bkblk** – Black king blocked?

Siyah şahın hareket edemeyecek şekilde bloklanmış olup olmadığını belirtir.

**bknwy** – Black king not with white?

Siyah şah ile beyaz taşlar arasında doğrudan temas yok mu?

**bkon8** – Black king on 8th rank?

Siyah şah 8. yatayda mı?

**bkona** – Black king on a-file?

Siyah şah "a" sütununda mı?

**bkspr** – Black king support rank?

Siyah şah destekleyici bir konumda mı (muhtemelen piyonu koruyor mu)?

**bkxbq** – Black king x-blocked by queen?

Siyah şah, beyaz vezir tarafından çaprazdan mı engelleniyor? (ya da benzeri bir anlam)

**bkxcr** – Black king x-blocked by rook?

Siyah şah, beyaz kale tarafından çapraz ya da çizgisel olarak mı engelleniyor?

**bkxwp** – Black king x-blocked by white pawn?

Siyah şah, beyaz piyon tarafından engellenmiş mi?

**blxwp** – Black left x white pawn?

Siyah taşın solunda beyaz piyon var mı?

**bxqsq** – Black on queening square?

Siyah piyon vezir çıkma karesinde mi?

**cntxt** – Context

Genel pozisyonun bağlamsal durumu (belki hamle sayısı, ya da taşların dizilimi).

**dsopp** – Distance to opponent?

Rakip şah ile mesafe?

**dwipd** – Double white pawn defense?

Beyazın piyonları birbirini savunuyor mu?

**hdchk** – Horizontal check?

Yataydan şah çekme tehdidi var mı?

**katri** – King at triangle?

Şah belirli bir "üçgen" formasyonda mı

**mulch** – Multiple check possibilities?

Çoklu şah tehditi mi var?

**qxmsq** – Queen can move to square?

Vezir belli kareye oynayabiliyor mu? (ya da kale bu kareye gidebiliyor mu?)

**r2ar8** – Rook from 2nd to 8th rank?

Kale 2. yataydan 8'e mi gidebilir?

**reskd** – Restricted king defense?

Şahın savunması sınırlı mı?

**reskr** – Restricted rook?

Kale sınırlanmış mı (hareket alanı dar)?

**rimmx** – Rook immediate max threat?

Kale anlık maksimum tehdit yaratıyor mu?

**rkxwp** – Rook captures white pawn?

Kale beyaz piyonu alabilir mi?

**rxmsq** – Rook move square?

Kalenin hamle yapabileceği kare var mı?

**simpl** – Simplified position?

Basitleştirilmiş bir pozisyon mu (az taşlı)?

**skach** – Skewer check?

Skewer (şişleme) ile şah tehdidi var mı?

**skewr** – Skewer tactic possible?

Şişleme taktiği uygulanabilir mi?

**skrxp** – Skewering rook captures pawn?

Şişleme pozisyonundaki kale piyon alabilir mi?

**spcop** – Support by cooperation?

Taşlar arasında karşılıklı destek var mı?

**stlmt** – Stalemate threat?

Pat olma ihtimali var mı?

**thrsk** – Threatened by risk?

Riskli bir tehdit mi mevcut?

**wkcti** – White king central/influence?

Beyaz şah merkeze yakın mı ya da etkili bir pozisyonda mı?

**wkna8** – White king not on a8?

Beyaz şah a8 karesinde değil mi?

**wknck** – White king check?

Beyaz şah şahta mı?

**wkovl** – White king overlaps path?

Beyaz şahın pozisyonu başka taşların yolu üzerinde mi?

**wkpos** – White king position

Beyaz şahın konumu

**wtoeg** – White to endgame?

Beyaz taraf oyunu bitirebilir mi?

**2.2. Kullanılan Makine Öğrenmesi Algoritmaları**

**Rastgele orman (Random Forest) yöntemi**, makine öğrenmesinde kullanılan kuvvetli ve yaygın bir yöntemdir. Rastgele orman algoritması birden fazla karar ağacı oluşturur ve bu ağaçların sonuçlarını birleştirerek daha doğru ve dengeli bir tahmin yapar. Her karar ağacı veri kümesinden rastgele seçilen bir alt küme ile eğitilir ve ağaçlar birbirinden bağımsız çalışır. En son bu ağaçların ortak kararı alınır.

**K-NN (K-Nearest Neighbors) yöntemi**, makine öğrenmesinde oldukça sık kullanılan basit ve etkili bir yöntemdir. Yöntem temel olarak, bir veriyi sınıflandırırken ona en yakın olan K tane komşusuna bakar. Veri noktaları arasındaki benzerlik genellikle mesafe (Öklid, Minkowski, Manhattan vs.) hesaplanarak bulunur. Yeni bir veri geldiğinde, eğitim verisindeki mesafe bakımından en yakın K sayıdaki komşuya bakılır ve çoğunluk hangi sınıftaysa yeni veri o sınıfa atanır.

**Bayes teoremi**, olasılık hesaplamalarında kullanılan ve bir olayın olasılığını yeni bilgiler ışığında güncelleme imkanı veren temel bir yaklaşımdır. Yani, elimizde bir bilgi varken bu bilgiye yeni bir veri eklendiğinde, olayın gerçekleşme ihtimalinin nasıl değiştiğini gösterir. Bu yöntem, geçmiş deneyimlerle elde edilen bilgileri, sonradan gelen verilerle birleştirerek daha iyi tahminler yapmamızı sağlar. Bayes teoremi aslında istatistik bilimindeki koşullu olasılık mantığını temel alır ve özellikle belirsizlik içeren durumlarda karar vermek için oldukça işe yarar.

**Lojistik regresyon**, bir olayın olma olasılığını tahmin etmek için kullanılan bir yöntemdir. Sonuçlar 0 ile 1 arasında bir değer alır ve genellikle evet-hayır gibi iki sınıflı durumlar için kullanılır. Lineer regresyondan farklı olarak, sonuçları sınırlı bir aralığa (0-1) sıkıştırmak için özel bir lojistik eğrisi kullanır. Model, verilen özelliklere göre bir olayın gerçekleşme ihtimalini hesaplar ve belirli bir eşik değerine göre sınıflandırma yapar.

**Karar ağacı**, veri setini daha küçük gruplara ayırarak kararlar veren bir makine öğrenmesi yöntemidir. Her bir dal, bir özellik üzerinde yapılan bir testi temsil eder ve her bir yaprak ise sınıf etiketini veya tahmin edilen değeri gösterir. Ağacın her seviyesinde, veri setindeki entropiyi yani belirsizliği en aza çekecek olan veriyi kullanarak ayırma işlemi yapar. Yani veriyi mümkün olan en iyi şekilde bölmek için bilgi kazancını kullanır.

**2.3. Literatür Taraması**

Makine öğrenmesi algoritmalarının satranç üzerindeki oyun ve hamle tahmini için kullanılması, makine öğrenmesinin oldukça popüler bir uygulama alanıdır. Lakin ilginç bir olay vardır ki bu da ilk Satranç robotunun makine öğrenme algoritmalarının ve hatta bilgisayarların icadından önce olmasıdır. “Türk” adı verilen bu satranç robotu aslında bir kabin içerisinde gizlenen bir satranç ustasından ibarettir. Her ne kadar gerçek bir robot olmasa da bu olay makine öğrenmesinin satranç ile ilişkisinin derinliği hakkında fikir vermektedir.

Gerçek manada makine öğrenmesinin satranç üzerindeki ilk büyük çalışması, 1980'li yılların sonlarına doğru geliştirilen Deep Thought adlı bilgisayar sistemiyle başlamıştır. Carnegie Mellon Üniversitesi bünyesinde geliştirilen Deep Thought, o dönemin en güçlü bilgisayar satranç robotu olarak kabul edilmiştir. Sistem, temel olarak heuristik algoritmalar, statik değerlendirme fonksiyonları ve arama derinliğine dayalı hamle tahmin mekanizmaları ile çalışmaktaydı. Bununla birlikte, sınırlı da olsa bazı öğrenme kabiliyetlerine sahipti ve bu sayede insan ustalarla karşılaştırılabilir düzeyde performans göstermişti. Deep Thought’un geliştirilmesi, satranç gibi yüksek kombinasyonel karmaşıklığa sahip bir problem üzerinde bilgisayarlı karar verme süreçlerinin uygulanabilirliğini göstermesi açısından önemli bir kilometre taşıdır.

Bu başarıyı takip eden süreçte IBM, Deep Thought projesini satın alarak onu daha gelişmiş bir versiyon olan Deep Blue sistemine dönüştürmüştür. Deep Blue, 1997 yılında satranç dünya şampiyonu Garry Kasparov'u mağlup ederek tarihe geçmiştir. Her ne kadar bu olay makine tabanlı zekanın insan karşısında kazandığı ilk büyük zafer olarak kabul edilse de, Deep Blue’nun mimarisi günümüz makine öğrenmesi tanımına tam olarak uymamaktadır. Çünkü sistemde öğrenme becerisi bulunmamaktadır. Sistem, insan uzmanlar tarafından oluşturulan açılışlar, insan eli ile tasarlanmış taktikler ve derinlemesine arama algoritmaları ile donatılmıştır. Bu nedenle, oyun içerisinde ya da oyunlar arası aktif öğrenme ve kendini geliştirme yetisi bulunmamaktadır. Başka bir deyişle, Deep Blue’nun başarısı, öğrenen bir modelden ziyade, insan bilgisinin bilgisayarın çok güçlü hesap yetenekleriyle birleştirilmiş versiyonudur.

Günümüzde ise satrançta makine öğrenmesinin gerçek anlamda etkili bir şekilde kullanıldığı sistemler gelişmiştir. Özellikle açık kaynaklı bir satranç motoru olan Stockfish, klasik algoritmik yapıların ulaştığı zirveyi temsil etmektedir. Lakin Stockfish, tıpkı az önce bahsettiğimiz Satranç motorları gibi kendi kendine öğrenmeden ziyade çok yüksek hesaplama gücünün insan rehberliği ile birleştirilmesinden oluşturulmuştur.

Öte yandan, 2017 yılında Google DeepMind tarafından geliştirilen AlphaZero, satrançta makine öğrenmesinin devrim niteliğindeki örneğidir. AlphaZero, herhangi bir açılış veritabanı veya insan bilgisinden faydalanmaksızın yalnızca kendi kendine oynadığı milyonlarca oyun sayesinde satrancı öğrenmiş ve kısa sürede Stockfish’i bile mağlup edebilmiştir. Bu sistem, derin pekiştirmeli öğrenme (deep reinforcement learning) ve sinir ağları kullanarak pozisyonları değerlendirmekte ve hamle üretmektedir. AlphaZero’nun başarısı, klasik programlama yaklaşımlarının ötesine geçerek, öğrenme temelli modellerin satranç gibi karmaşık problemler üzerinde üstün performans sergileyebileceğini göstermektedir. Bu gelişme, makine öğrenmesinin satranç uygulamalarında ulaştığı noktayı temsil etmekte ve yapay zeka araştırmaları için önemli bir örnek sunmaktadır.

1. **Bulgular ve Tartışma**

Tablo 1 ve 2: Elde edilen sonuçlar (random\_state = 100 alınmıştır, virgülden sonra 6 hane kesme kullanılmıştır)

|  |  |  |
| --- | --- | --- |
| Algoritma | %20 test verisi ile doğruluk oranı | %30 test verisi ile doğruluk oranı |
| K-NN | 0.942187 | 0.936392 |
| **KARAR AĞACI** | **0.99375** | **0.987486** |
| BAYES TEOREMİ | 0.628125 | 0.631908 |
| LOJİSTİK REGRESYON | 0.9375 | 0.937434 |
| RASTGELE ORMAN | 0.9875 | 0.983315 |

Yukarıda kullanılan yöntemlerin %20 test, %80 eğitim ve %30 test, %70 eğitim işlemleri sonucundaki doğruluk oranları

görülmektedir.

çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma, diyagram, ekran görüntüsü içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

|  |  |  |  |
| --- | --- | --- | --- |
| Algoritma | Kesinlik  (Precision) | Duyarlılık  (Recall) | F1-Score |
| K-NN | 0.950292 | 0.942028 | 0.946142 |
| KARAR AĞACI | 0.99375 | 0.99375 | 0.99375 |
| BAYES TEOREMİ | 0.765165 | 0.597101 | 0.536942 |
| LOJİSTİK REGRESYON | 0.939481 | 0.944927 | 0.942196 |
| RASTGELE ORMAN | 0.987675 | 0.987177 | 0.987416 |

Yukarıda kullanılan yöntemlerin %20 test, %80 eğitim durumlarında kesinlik duyarlılık ve f1-score görülmektedir.

metin, ekran görüntüsü, diyagram, renklilik içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**Karar ağacı:**

diyagram, plan, piksel, tasarım içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

Max\_depth=3, random\_state=100

Reskr= Kalenin etkinliği

Wkna8=Beyaz şah a8’de mi

Blxwp=Siyah şah beyaz’ın ilerlemesini durdurabilir mi

Wkcti=Beyaz şah tehdit altında mı

Qxmsq=Kale ile tehdit edilen kare var mı

Wknck=Beyaz, Şah çekiyor mu

**Karar ağacı karışıklık matrisi:**

metin, ekran görüntüsü, diyagram, yazı tipi içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**KNN sütun grafiği:**

metin, ekran görüntüsü, diyagram, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**KNN karışıklık matrisi:**

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**Rastgele Orman Önem dağılımı Grafiği:**

metin, ekran görüntüsü, çizgi, öykü gelişim çizgisi; kumpas; grafiğini çıkarma içeren bir resim

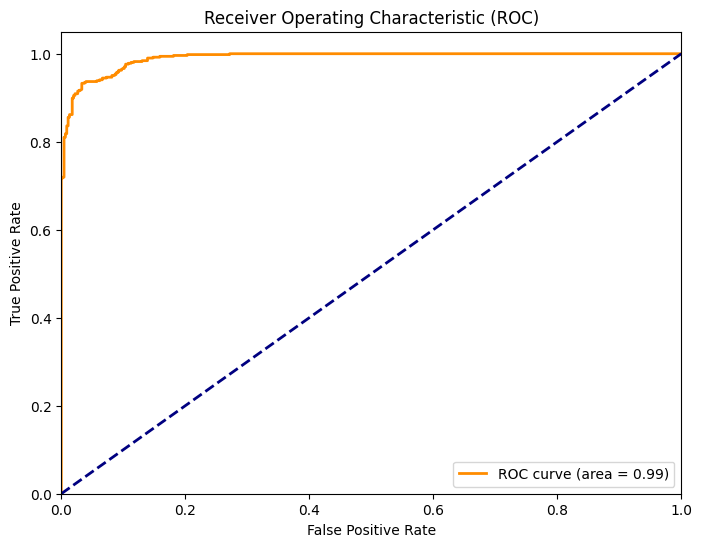
Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**Rastgele Orman karışıklık matrisi:**

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**Lojistik Regresyon ROC grafiği:**



**Lojistik Regresyon karışıklık matrisi:**

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

**Bayes Teoremi Karışıklık matrisi:**

metin, ekran görüntüsü, diyagram, dikdörtgen içeren bir resim

Yapay zeka tarafından oluşturulan içerik yanlış olabilir.

1. **Sonuç ve Öneriler**

Yapılan çalışmada, satranç oyununda KRKPA-7 konumunda beyazların kazanıp kazanamayacağın olasılığının makine öğrenmesi yöntemleriyle tahmin edilmesi amaçlanmıştır. Bu yöntemler; Bayes, Karar Ağacı, k-NN, Rastgele Orman ve Lojistik Regresyondur. Yapılan çalışma sonucunda, kullanılan algoritmalar arasında en başarılı olan algoritmanın Karar Ağacı algoritması, en başarısız olan algoritmanın ise Bayes Teoremi olduğu saptanmıştır. Karar Ağacı algoritmasının bu denli başarılı olmasının temel sebebi, verideki sınıflandırma kriterlerini hiyerarşik bir şekilde ayırabilmesi, bu sayede karmaşık örüntüleri kolaylıkla öğrenebilmesidir. Özellikle satranç gibi deterministik, belirli kurallara dayalı veri setlerinde karar ağaçları, veriyi etkili biçimde dallara ayırarak sınıflandırma işlemini yüksek doğruluk oranıyla gerçekleştirebilmektedir. Bayes Teoreminin düşük doğruluk göstermesinin nedeni ise, algoritmanın veri setindeki değişkenler arasındaki ilişkiyi anlamak için koşullu olasılıklara dayanmasıdır. Satranç oyununda ise özellikler birbirine bağlıdır. Bu nedenle bağımsızlık varsayımı çoğu zaman geçerli olmamaktadır. Çünkü satrançta bir hamle diğerini etkilemektedir. Bu da Bayes yönteminin sınıflandırma gücünü zayıflatmıştır. Çalışmayı yaparken farklı oranlardan eğitim verisi kullanılmıştır. Veri setinin %90’ı kullanıldığında test verisinin çok düşük kalmaması sebebiyle, doğrulama güvenilirliği azaldığı gözlemlenmiştir. Bu yüzden veri setinin %80’i ve %70’i olmak üzere iki farklı eğitim verisi ayrılmıştır. Bu verilerin rastgeleliğini önlemek için random\_state değeri her iki deneyde de sabit tutulmuştur.

Bu çalışma, satranç oyun sonu pozisyonlarından biri olan KRKP-A7 konumunun analizine odaklanmış olup, gelecekte daha geniş veri kümeleriyle ve pozisyon çeşitliliğiyle zenginleştirildiğinde, yalnızca sonucun tahminiyle sınırlı kalmayıp, hamle önerisi yapabilen veya pozisyonları değerlendiren daha gelişmiş yapay zeka sistemlerine evrilebilir. Pozisyon derinliği arttırılarak, çok hamleli kombinasyonları değerlendirme kabiliyetleri kazandırılabilir ve bu da çalışmayı, Stockfish benzeri satranç motorlarının öğrenen bir versiyonuna dönüştürme potansiyeli taşıyabilir.

**KAYNAKÇA**

* Standage, T. (2002). The Turk: The Life and Times of the Famous Eighteenth-Century Chess-Playing Machine. Walker Publishing Company.​
* <https://en.wikipedia.org/wiki/Deep_Thought_(chess_computer)>
* <https://en.wikipedia.org/wiki/Deep_Blue_versus_Garry_Kasparov>
* <https://en.wikipedia.org/wiki/Stockfish_(chess)>
* <https://deepmind.google/discover/blog/alphazero-shedding-new-light-on-chess-shogi-and-go/>
* <https://tr.wikipedia.org/wiki/Bayes_teoremi>
* Shapiro, A. D. (1987). Structured Induction in Expert Systems. Addison-Wesley.
* Shapiro, A. D. (1983). The Role of Structured Induction in Expert Systems (Doctoral dissertation, University of Edinburgh).
* UCI Machine Learning Repository: [KRKPA7 Data Set](https://archive.ics.uci.edu/ml/datasets/King-Rook+vs.+King-Pawn)