T.C.
NECMETTİN ERBAKAN ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YAPAY ÖĞRENME İLE GEN DİZİLİMİNDE EKSİK VERİNİN TAMAMLANMASI

Mithat Raşit ÖZÇIKRIKCI

YÜKSEK LİSANS TEZİ

Endüstri Mühendisliği Ana Bilim Dalı

Temmuz - 2018
KONYA

TEZ KABUL VE ONAYI

Mithat Raşit Özçıkrıkcı tarafından hazırlanan "Yapay Öğrenme ile Gen Diziliminde Eksik Verinin Tamamlanması" adlı tez çalışması 9/11/2018 tarihinde aşağıdaki jüri tarafından oy birliği ile Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü Endüstri Mühendisliği Ana Bilim Dalı'nda YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Jüri Üyeleri	**İmza**
Başkan Prof. Dr. Sabri KOÇER	……..
Danışman Dr. Öğr. Üyesi Ali Osman ÇIBIKDİKEN	……..
Üye Dr. Öğr. Üyesi Muhammed Abdullah BÜLBÜL	……..

Yukarıdaki sonucu onaylarım.

Prof. Dr. Ahmet AVCI
FBE Müdürü

TEZ BİLDİRİMİ

Bu tezdeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz olarak atıf yapıldığını bildiririm.

DECLARATION PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

İmza

Mithat Raşit Özçıkrıkcı

ÖZET

YÜKSEK LİSANS TEZİ

YAPAY ÖĞRENME İLE GEN DİZİLİMİNDE EKSİK VERİNİN TAMAMLANMASI

Mithat Raşit ÖZÇIKRIKCI

Necmettin Erbakan Üniversitesi Fen Bilimleri Enstitüsü
Endüstri Mühendisliği Ana Bilim Dalı

Danışman: Dr. Öğr. Üyesi Ali Osman ÇIBIKDİKEN

2018, 55 Sayfa

Jüri
Dr. Öğr. Üyesi Ali Osman ÇIBIKDİKEN
Prof. Dr. Sabri KOÇER
Dr. Öğr. Üyesi Muhammed Abdullah BÜLBÜL

Canlıların DNA dizilimleri büyük oranda aynıdır. DNA dizilimi içerisinde polimorfik özellik gösteren farklı alt dizilim bölgeleri ise aynı türe sahip bireylerin birbirinden farklı olan yönlerini kodlar. Bireye göre polimorfik özellik gösteren alt dizilimlerin toplamı ilgili bireyin genotipini oluşturur. Hücre örneklerinden genetik materyalin elde edilmesi ve elde edilen genetik materyalden de genotip verisinin elde edilme sürecinde cihaz hassasiyeti kaynaklı kayıp veri problemi ile karşılaşılmaktadır. Eldeki genotip verisi üzerinde kayıp veri bölgelerinin bulunması ise genetik araştırmaları kötü yönde etkilemektedir. Kayıp veri bölgelerine sahip genotip verileri üzerinde yüksek başarımlı analizler gerçekleştirilememektedir. Bundan kaynaklanan istatistiksel eksiklikler genetik haritalamayı imkansız hale getirdiği için fenotipik özelliklerin veya herhangi bir hastalığın genetik arka planı bulunamamaktadır. Tüm bu sebeplerden dolayı kayıp verilerin tamamlanması, biyoenformatik analizlerin en temel gerekliliği haline gelmektedir.

Genotip verisi üzerinde kayıp veri tahmini işlemini gerçekleştirebilmek için farklı istatistiksel modeller kullanılmakla birlikte en çok kullanılan istatistiksel model, Markov zincirleri olmuştur. Markov zincirleriyle yüksek başarımlı kayıp veri tahmini yapılabilmektedir. Çalışmada Markov zincirleriyle yapılan çalışmalar incelenmiş olup, patlıcan türündeki bireylere ait kayıp veri bölgeleri içeren genotip verisi üzerinde uygulanmış olup aynı veri üzerinde kayıp veri tahmini işleminin yapay öğrenme algoritmalarıyla gerçekleştirilmesi araştırılıp aynı patlıcan genotip veri seti üzerinde uygulanmıştır.

Anahtar Kelimeler: Derin öğrenme, kayıp veri tahmini, yapay öğrenme, gen dizilimi

ABSTRACT

MS THESIS

IMPUTATION OF GENOME SEQUENCES USING MACHINE LEARNING

Mithat Raşit ÖZÇIKRIKCI

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCE OF NECMETTIN ERBAKAN UNIVERSITY
THE DEGREE OF MASTER OF SCIENCE
IN INDUSTRIAL ENGINEERING

Advisor: Assistant Prof. Dr. Ali Osman ÇIBIKDİKEN

2017, 55 Pages

Jury
Assistant Prof. Dr. Ali Osman ÇIBIKDİKEN
Prof. Dr. Sabri KOÇER
Assistant Prof. Dr. Muhammed Abdullah BÜLBÜL

DNA sequences of living organisms are substantially equivalent. Polymorphic subsequence regions on DNA sequence define different from another attribute of same kind of individuals. Genotype consist of this polymorphic subsequences. Obtaining genetic material from cell samples and obtaining genotype data from obtained genetic material is encountered with lost data problem. The discovery of missing data regions on the existing genotype data is affecting the genetic researches in the worst way. Analysis is impossible on genotype data that contains missing subsequence region. Since genetic mapping can not be performed, the disease genetic relationship can not be established today.

The most commonly used statistical model is the Markov chains, although different statistical models are used to perform the loss estimation on the genotype data. Markov chains can predict high performance loss data. In the study, Markov chain and machine learning based imputation methods are researched and estimation of lost data was performed by using deep learning methods on genotype data of eggplant tuna.

Keywords: Deep learning, missing data imputation, machine learning, genetic sequencing

ÖNSÖZ

Veri seti içerisinde kayıp verilerin tahmini oldukça önemli bir yer tutmaktadır. Mevcut veri yapısına bakarak kayıp veri hakkında öngörüde bulunmak için istatistiki ve yapay öğrenme metotları bulunmaktadır.

Tez çalışmasında genetik dizilim verisi üzerinde **kayıp veri tahmini** (*imputation*) yapan algoritmalar üzerinde çalışılmıştır. Genotip verisi üzerinde kayıp veri tahmini yapan algoritmalar incelenip aynı işlemin yapay öğrenme algoritmalarıyla nasıl yapılabileceği araştırılmıştır. Bu alanda geliştirilen algoritmalar bilgisayar destekli hesaplama sistemleri yardımıyla analiz edilip farklı algoritmaların sonuçlarına yönelik kıyaslama yapılmıştır.

Öğrencilik hayatım, meslek hayatım ve tez çalışmam boyunca desteğini benden esirgemeyen kıymetli hocam Dr. Öğr. Üyesi Ali Osman ÇIBIKDİKEN'e ve değerli kardeşim Öğretim Görevlisi Mücahit BÜYÜKYILMAZ'a bu çalışma vesilesiyle şükranlarımı sunarım.

Bir bilgisayar mühendisi olarak genetik bilimi içerisinde karşılaştığım zorlukları aşmada geniş bilgi birikimiyle tez çalışmam boyunca yanımda olan değerli hocam Dr. Öğr. Üyesi Ali Tevfik UNCU'ya da şükranlarımı samimiyetle arz ederim.

Mithat Raşit ÖZÇIKRIKCI
KONYA-2018

İÇİNDEKİLER

KISALTMALAR

Kısaltmalar

- **MLP** **:** Multi Layer Neural Networks
- **GPU** **:** Graphics Processing Unit
- **CPU** **:** Central Processing Unit
- **ANN** **:** Artificial Neural Networks
- **LMS** **:** Least Mean Square
- **LD** **:** Linkage Disequilibrium
- **HMM** **:** Hidden Markov Model

1. GİRİŞ

Kayıp veri tahmini (*Imputation*), eldeki veri yığını içerisindeki kayıp verileri belli analizler sonucu dolduran istatistiksel işlemlerdir. Kayıp verileri de içeren eldeki veri yığını bir takım "kayıp veri tahmini" analizlerinden geçer. Analiz sonucu kayıp veri bölgeleri için elde edilen değerler veri yığınında yerine konularak kayıp veri tahmini işlemi tamamlanır.

1.1. Çalışmanın Amacı

Çalışmada genetik diziliş veri yığınları içerisindeki kayıp verilerin tahmini için, geliştirilmiş istatistiksel yöntem ve algoritmalar araştırılmıştır. Sonuçlar kıyaslanarak, derin öğrenme (*deep learning*) algoritmalarıyla kayıp veri tahmini işleminin nasıl yapılabileceğine yönelik araştırmalar yapılmış ve örnek veri seti üzerinde uygulanmıştır.

Bilgisayar hesaplama gücünün artışı ile büyük veri yığınlarının analizi ve işlenmesi kolaylaşmıştır. Aynı zamanda yüksek hesaplama gücü işlem sürelerini de kısaltmıştır. Bütün bu gelişmeler ışığında, yapay öğrenme yaklaşımı ile gen dizilimlerinin öğrenebilen sistemlerle bir platform şeklinde geliştirilmesi, genetik araştırmacılarının hizmetine sunulabilmesi mümkün hale gelecektir.

1.2. Çalışmanın Önemi

Genetik bilimindeki ilerlemeler bilim dünyasını hastalıkların ve doğuştan gelen bir takım bozuklukların genetik boyutlarını araştırmaya yönlendirmiştir. Canlılara ait özelliklerin ilgili canlının genomunun hangi bölgesiyle ilişkili olduğunu belirlemeye yönelik bir dizi genom bazlı ilişki araştırması (Genome Wide Association Study) çalışmaları sürmektedir. Bu sayede canlılara ait hastalık ve bozuklukların genom bazında engellenmesi hedeflenmektedir.

Canlılara ait genetik bilgi elde edilirken kayıplar yaşanabilmektedir. Canlıya ait hücreden genetik veri okuma işlemi yapılırken (*extraction*), işlemi yapan cihaza bağlı olarak bazı genetik bölgeler okunamamaktadır. Bu da elde edilen genetik veri yığını içerisinde kayıp bölgeler oluşturmaktadır. Eldeki genetik veri yığını içerisinde kayıp bölgelerin bulunması **genom bazlı ilişki araştırması** çalışmalarını olumsuz yönde

etkilemektedir. Eksik alt dizilim bölgelerini içeren bir genotip dizilim verisi üzerinde analiz imkansız hale geldiği için ilgili verinin yorumlanması da imkansız hale gelmektedir. Bu noktada **kayıp veri tahmini** çalışmaları genom bazlı ilişki araştırmalarının başarısını etkileyen önemli unsurlardan biri olmaktadır.

1.3. Tezin Yapısı

Tez beş bölümden oluşmaktadır.

Birinci bölümde kayıp veri tahmini problemi ve genetik verileri üzerindeki önemi ele alınmıştır. Ayrıca çalışmanın amacı ve önemi de bu bölüm altında belirtilmiştir.

İkinci bölümde temel genetik, kayıp veri tahminleme ve yapay öğrenme konularında yapılan literatür taraması anlatılmıştır. Literatürdeki çalışmalarda bu bölümde incelenmiştir.

Üçüncü bölümde çalışma boyunca kullanılan materyal ve metotlar verilmiştir.

Dördüncü bölümde çalışmada kullanılan algoritma ve sonuçların veri seti üzerinde gerçeklenmesinin sonuçlarından bahsedilmiştir.

Beşinci bölümde ise çalışmanın sonuçları değerlendirilmiş ve buna bağlı olarak öneriler verilmiştir.

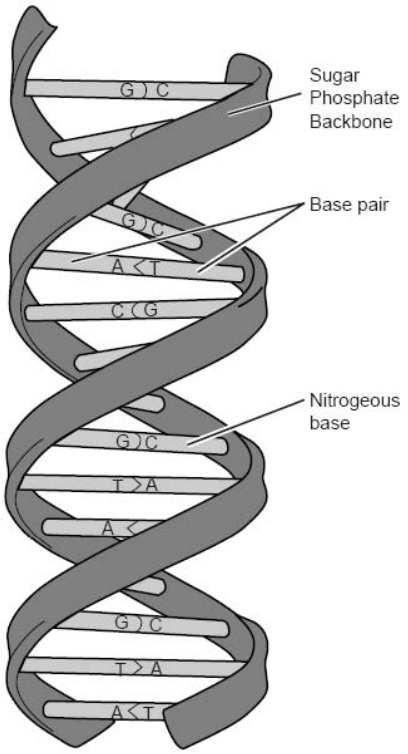
2. KAYNAK ARAŞTIRMASI

2.1. Temel Genetik Kavramlar

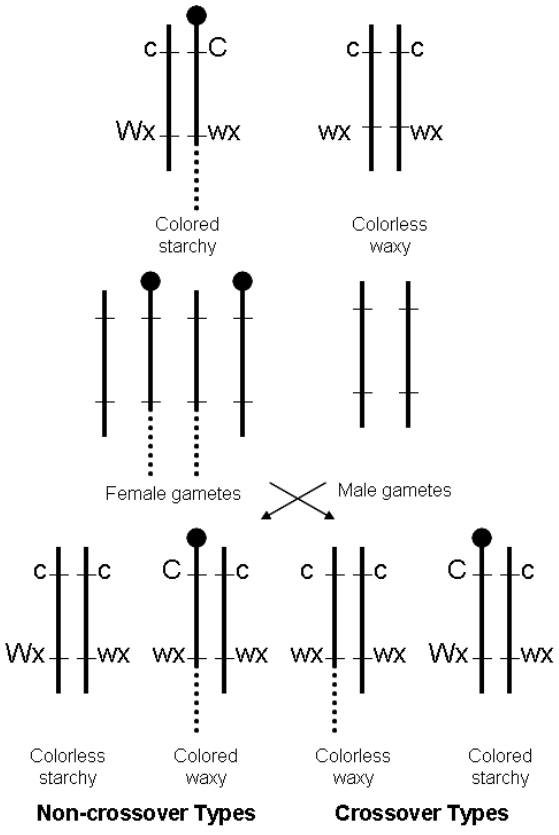
Temel genetik kavramlar aşağıda verilmiştir (Kandemir, 2010):

- **Nükleotid (***nucleotide***):** DNA sarmallarını oluşturan baz çiftlerinden herbirine bir nükleotit denir. Her nükleotit Adenin, Guanin, Timin ve Sitozin olmak üzere 4 baz'dan birini taşır. DNA çift sarmal dan oluşan bir yapıda olduğu için her bir nükleotit karşı iplikteki başka bir nükleotid ile eşleşir. Nükleotitlerin eşleşmesi aşağıdaki kurallara göre gerçekleşir:
    - **Adenin** bazlı nükleotid **timin** bazlı nükleotid ile eşleşebilir.
    - **Guanin** bazlı nükleotid **sitozin** bazlı nükleotid ile eşleşebilir.
- **DNA** (*Deoxyribonucleic acid*)**:** Canlılarda nesilden nesile aktarılan ve türün kalıtsal özelliklerini belirleyen kalıtım materyalidir. DNA çift sarmal dan oluşan iki iplikli bir yapı olarak organize olmuş bir moleküldür. Her bir sarmal bir nükleotid dizisidir. Bir DNA sarmalı örneği Şekil 2.1. de gösterilmiştir.
- **SNP** (*Single Nucleotide Polymorphism*): İki DNA molekülü arasındaki tekil nükleotid farklardır. Aynı türdeki canlıların genetik dizilimleri büyük ölçüde aynıdır. Farklı olan bölgeler SNP lerden oluşur. SNP ler için canlının genetik profilini oluşturan nükleotidler denebilir. Çünkü bireyden bireye polimorfik bir özellik gösterir.
- **Alel** (*Allele*)**:** SNP lerden meydana gelen belirli bir özelliği taşıyan genlere alel genler denir. Aynı türe ait bireylerin polimorfik bir özelliğini kodlar. Örneğin, göz rengini belirleyen genler ele alındığında siyah göz rengini veren gende, kahverengi göz rengini veren gende birer alel gendir. Bu özellikler bireyden bireye farklılık gösterdiği için her bir farklılığı kodlayan SNP dizilimi de farklı olmak zorundadır. Buradaki her bir farklı SNP dizilimi kodladığı özelliğin aleli olarak adlandırılır.
- **Lokus** (*Locus*)**:** Bir genin bir kromozom üzerindeki lokasyonudur. Herhangi bir gene ait DNA dizilimini kapsayan kromozomal bölgedir.

- **Rekombinasyon** (*Recombination*)**:** Mayoz bölünme sırasında genetik materyalin iki farklı kromozom arasında veya aynı kromozom içerisinde yer değiştirmesidir (Şekil 2.2). Buradaki genetik materyal haplotipler dir. Sonuç olarak genetik materyal nesilden nesile aktarılırken haplotipler halinde aktarılmakla beraber haplotipler asla parçalanamamaktadır.
- **Haplotip** (*Haplotype*)**:** Bir kromozom içerisinde birlikte hareket eden SNP dizilerinden oluşan DNA varyasyonları dır. Tek bir lokusu içerebileceği gibi komple bir kromozomu da içerebilir. İçerisinde alel kombinasyonlarını barındırır. Bir haplotip içerisinde yer alan genler birbirleriyle **bağımlıdır** (*linked*). Bu yüzden rekombinasyon boyunca birlikte hareket ederler. İnsan genomunun haplotip haritasını oluşturmak üzere bir takım akademik merkezler, araştırma grupları ve Kanada, Çin, Japonya, Nijerya, İngiltere ve ABD deki şirketlerden oluşan bir konsorsiyum tarafından Uluslararası HapMap Projesi başlatılmıştır. 10 milyon SNP ten oluşan veri setiyle ortak özellik taşıyan genetik alt dizilerin tanımlanması hedeflenmiştir. Bu sayede diyabet, kanser, kalp krizi, depresyon, astım gibi genetik arka planı bulunabilen hastalıkların genetik sebeplerinin belirlenerek genom bazlı ilişki araştırmalarına katkı sağlayabileceği düşünülmektedir.
- **Genotip** (*Genotype*)**:** Bir organizmadaki alel genlerin tamamı organizmanın genotipini oluşturur. Yani bir organizmanın aynı türün ortak özellikleri dışındaki özelliklerini kodlayan genetik dizilimi ilgili organizmanın genotipidir. Genom bazlı ilişki araştırmalarında genetik arka planı bulunan hastalıkları ele veren unsur SNP lerdir.



Şekil 2.1. Örnek bir DNA sarmalı (Kapiel, 2006)



Şekil 2.2. Mayoz bölünmede rekombinasyon (Clancy, 2008)

2.2. Bağlantı Dengesi ve Bağlantı Dengesizliği

Genetik bilimi üzerine yapılan çalışmalar, aynı kromozom üzerinde birbirine yakın lokasyonlarda bulunan genlerin birbirinden bağımsız olamayacaklarını göstermiştir. Örneğin, bir kromozom üzerinde (X) aleli (y) aleline bağlı ise (x) aleli de (Y) aleline bağlı olmak zorundadır. Bu durumda kromozom üzerinde bir alel bağlı olduğu alel ile birlikte bulunmak zorundadır. Aynı kromozom üzerinde iki farklı lokusta bulunan aleller arasında bulunan rastlantısal olmayan (non-random) bir bağımlılık **bağlantı dengesizliği** (*linkage disequilibrium*) meydana getirir. Çizelge 2.1. deki gibi aynı kromozom üzerinde bulunan iki farklı alelde bulunan SNP ler ele alındığında **bağlantı dengesi** (*linkage equilibrium*) ve **bağlantı dengesizliği** (*Linkage disequilibrium*) aşağıda belirlenen adımlarla elde edilir.

Çizelge 2.1. Aynı kromozom üzerinde bulunan iki alelde bulunan SNP'ler

Alel	**SNP1**	**SNP2**
Alel 1	G	A
Alel 2	C	T

Öncelikle alel frekansları ve haplotip frekansları Çizelge 2.2, Çizelge 2.3 ve Çizelge 2.4. deki gibi belirlenmelidir.

Çizelge 2.2. Alel frekansları

SNP1		**SNP2**	
Alel	**Frekans**	**Alel**	**Frekans**
G	p1	A	q1
C	p2	T	q2

Alellerin birbirine olan bağımlılıkları Alel-SNP frekansına bağımlı olduğu için frekansları belirlenen SNP lere ait aleller Çizelge 2.3 deki gibi belirlenmelidir.

Çizelge 2.3. Eldeki SNP'lerden oluşan haplotipler

		SNP2	
	Alel	A	T
SNP1	G	GA	GT
	C	CA	CT

Çizelge 2.4. Haplotip frekansları

Haplotip	**Frekans**	**Haplotip**	**Frekans**
GA	p11	GT	p12
CA	p21	CT	p22

Haplotip frekansı haplotipi meydana getiren alellerin frekansları çarpımına eşit olduğu durumlarda **bağlantı dengesi** meydana gelir. Yani bu aleller birbirinden bağımsızdır (Çizelge 2.5).

Çizelge 2.5. Haplotip frekansları ve alel frekans çarpımları

Haplotip Frekansı (Gözlenen)	**Alel frekans çarpımları (Beklenen)**
p11	p1×q1
p12	p1×q2
p21	p2×q1
p22	p2×q2

Gözlenen ve beklenen haplotip frekansları arasında fark oluşması durumunda **bağlantı dengesizliği** ortaya çıkar. Bu durumda ilgili aleller birbirine bağımlıdır. Aleller ve haplotipler arasındaki **bağlantı dengesizliği (LD)** Çizelge 2.6. daki gibi olur.

Çizelge 2.6. Alel'ler ve haplotipler arası bağlantı dengesizliği

LD11	p11 – (p1 × q1)
LD12	p12 – (p1 × q2)
LD21	p21 – (p2 × q1)
LD22	p22 – (p2 × q2)

Aleller arası bağlantı dengesizliği (LD);

$$LD = p_{11} \times p_{22} - p_{12} \times p_{21} \quad (1)$$

şeklinde formülize edilir (Lewontin R.C. 1988, Devlin B., Risch N. 1995).

2.3. Kayıp Veri Tahminlemede Kullanılan Temel Yaklaşımlar

Kayıp veri tahmini yöntemleri, **tekil** (*single imputation*) ve **çoklu** (*multiple imputation*) veri tahmini olmak üzere ikiye ayrılır. Tekil Kayıp Veri Tahmini (*single imputation*) ise Hot-deck, Cold-deck, Mean Substitution ve Regression olmak üzere dörde ayrılır. Kayıp veri tahmini işlemleri DNA dizilim ve genotip verileri için özelleştirildiğinde halihazırda literatürde tercih edilen yöntemler aşağıda başlıklar halinde verilmiştir.

2.3.1. En Yakın K Komşuluk *(K Nearest Neighbors)*

Birçok kayıp veri tahmini algoritması yüksek kalitede genotip verisine olduğu gibi ilgili türe ait referans genom ve referans genotip panele bilgisine de ihtiyaç duyar. Kayıp veri'nin tahmininde genotip verisi ve haplotip bilgisi yoğun olarak kullanılır.

En Yakın K Komşuluk algoritması, genotip verisi için dizayn edilmemiş olsa da kayıp genotip verilerinin tahmininde kullanılabilmektedir. Ancak kayıp genotip verisine en yakın **K** sayıda komşu değerlerin bulunabilmesi açısından genotip verisine ihtiyaç duymaktadır. Burada genotip verisinden kastedilen, kayıp genotip verisi içeren canlıyla aynı türde farklı bireylere ait genotip verisidir. Yani algoritmanın

çalıştırılabilmesi için aynı türde farklı bireylere ait genotip verilerinden oluşan bir veri setine ihtiyaç duyulmaktadır.

En Yakın K Komşuluk algoritmasının ilk aşamasını, en yakın k sayıda komşu değerlerin bulunabilmesi için kayıp genotip verisine sahip birey ile veri setindeki diğer bireyler arasındaki mesafelerin bulunması oluşturmaktadır. İki birey arasındaki fark;

$$d(s_1, s_2) = \frac{1}{n} \sum_{p \in P} |g(s_1, p) - g(s_2, p)| \quad (2)$$

ile bulunur. Burada g(s, p); s bireyinin p noktasındaki genotipini ifade eder. Aynı genotipin bireylerdeki konumlarının farkı alınarak genotipik fark elde edilir. Farkı alınmak istenen iki genotipin de kayıp olduğu durumlarda bu iki genotip işleme dahil edilmez. 1/n ifadesiyle de denklemin normalizasyonu sağlanır. Buradaki n değeri karşılaştırılan SNP sayısını ifade eder.

Eldeki veri seti içerisinden en yakın k sayıda komşu birey elde edilir. Son olarak k sayıdaki komşu birey içerisinden en yakın olan seçilip, eldeki kayıp genotip verisi seçilen bireye göre doldurulur. $g_i(s_i, p_j)$, tahmini yapılacak olan genotip verisi, N ise seçilen k sayıdaki komşu birey olmak üzere kayıp genotip verisi tahmini olmak üzere;

$$g_i(s_i, p_j) = argmax_{a \in \{0,1,2\}} \sum_{s \in N} \frac{1}{d_n(s_i, s)} I(g(s, p_j) = a) \quad (3)$$

şeklinde gerçekleştirilir. Burada $I(g(s, p_j)=a)$; bir işaret fonksiyonu olmak üzere $g(s, p_j)=a$ olduğu durumlarda 1, aksi halde 0 değerini alır.

Standart En Yakın K Komşuluk algoritması, tüm genom boyunca en çok benzeyen bireylerin, kayı genotip tahmini için en uygun küme olduğu varsayımına dayanır. Ancak bu bireylerin tüm genom boyunca yüksek benzerlik göstermeme olasılıkları da vardır.

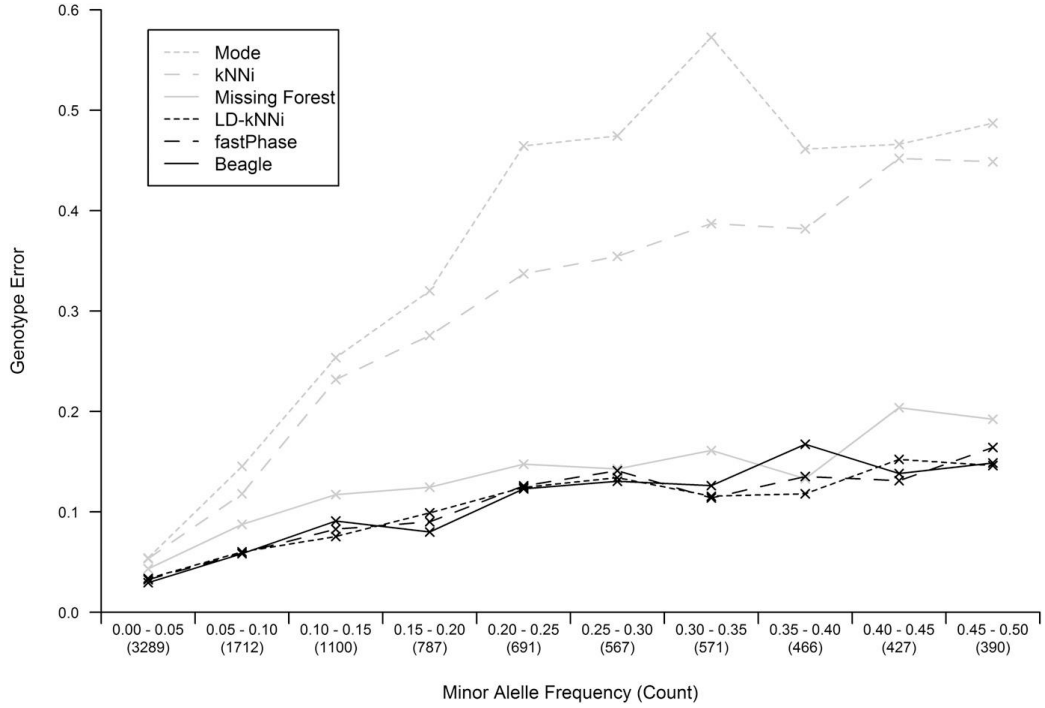
Money ve ark. (2015) tarafından standart En Yakın K Komşuluk algoritmasında SNP örnek uzayı olarak en yüksek **bağlantı dengesizliği (LD)** değerine sahip L tane SNP belirlenmiştir. Bu durumda (1) ifadesi;

$$d_l(s_1, s_2) = c + \frac{1}{n} \sum_{p \in L(p_i)} \left| g(s_1, p) - g(s_2, p) \right| \quad (4)$$

haline gelir (Money ve ark., 2015).

Money ve ark. (2015) tarafından üç farklı veri seti (elma, mısır, üzüm) üzerinde **KNN** tabanlı LinkImpute, **KNNi**, Hidden Markov Model tabanlı **Beagle**, **fastPhase**

isimli uygulamalar ve **Missing Forest** algoritması ile yapılan kayıp veri tahmini işlemlerine ait sonuçlar Şekil 2.3. de verilmiştir.



Şekil 2.3. KNNi ve diğer algoritmaların kayıp genotip veri tahmin performansı

2.3.2. Hidden Markov Model

Hidden Markov Model, ardışık yapıdaki veri kümeleri için geliştirilmiş olasılıksal bir modeldir. Görüntü işleme, ses işleme çalışmalarında olduğu gibi biyoinformatik alanında da protein modelleme, dizi hizalama, filogenetik analiz, kayıp genotip veri tahmini gibi çalışmalarda da sıkça kullanılmaktadır.

Hidden Markov Model, gözlemlenebilen bir dizilim ve gizli bir dizi üzerindeki olasılık dağılımıdır. Gözlemlenebilen dizilim üzerinde Markov süreçlerinin işletilerek gizli dizilimin tanımlanması hedeflenir (Şekil 2.4). Markov modelleri hafızasız sistemler olduğu için gelecek olan durum geçmiş durumlardan bağımsız olmak üzere sadece içinde bulunulan durumdan etkilenmektedir. Gözlemlenebilen durumların kümesi;

$$O = \{o_1, o_2, ..., o_M\} \quad (5)$$

ve gizli durumların kümesi;

$$H = \{h_1, h_2, ..., h_N\} \quad (6)$$

olmak üzere; gizli durumların geçiş matrisi;

$$A = \{ah_i, h_j\}_{1<=i,j<=N} \quad (7)$$

olur ve h_i durumundan h_j durumuna geçiş olasılığı;

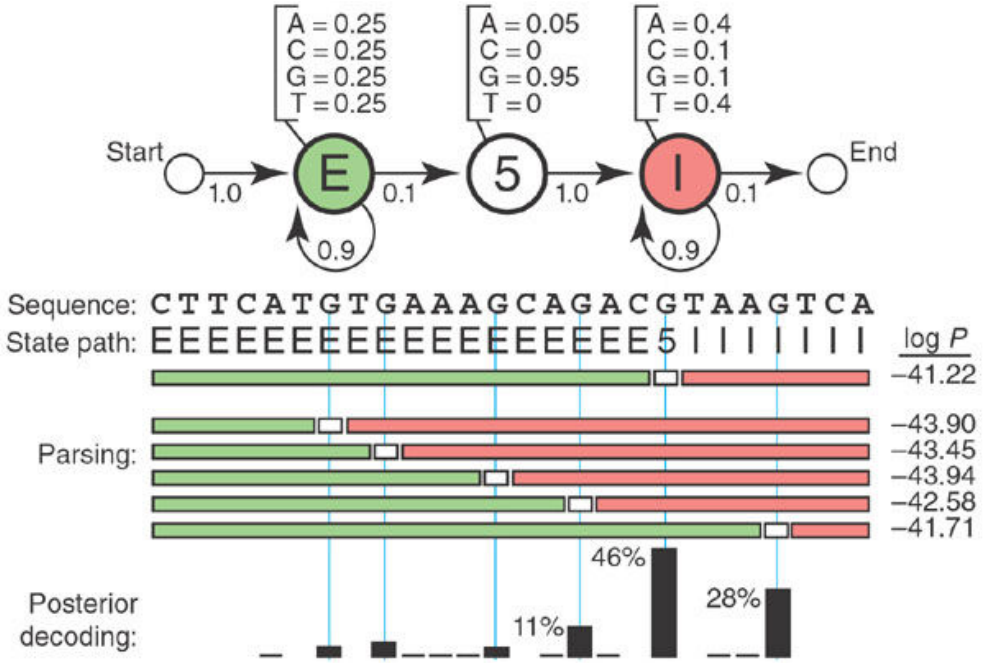
$$ah_i,\ h_j = P(X_t = h_j \mid X_{t-1} = h_i) \quad (8)$$

dir.

Gözlemlenebilen dizilimin o_j elemanından, h_i gizli dizilim elemanına geçiş matrisi;

$$B = \{b_{hi,\,oj}\}_{1<=i<=N}^{1<=j<=M} \quad (9)$$

şeklinde ifade edilir (Tataru ve ark., 2013).



Şekil 2.4. Hidden Markov Model (Eddy, 2004)

Hidden Markov Model olasılık değerlerinin hesaplanması ve kayıp veri bölgelerinin tahmini için kendi içerisinde bir takım algoritmalar kullanılır.

Berry ve arkadaşı (2011) tarafından density panellerin doğruluk derecelerini anlamak amacıyla bir çalışma yapılmıştır. Yapılan çalışmada low density panel verisinden high density panel verisinin elde edilmesi hedeflendi. Hidden Markov Model tabanlı Beagle isimli kayıp veri aracı kullanılarak gerçekleştirilen çalışmada 764 tane 2006 öncesi doğmuş Holstein Frieisan türü inekten oluşan popülasyon tahmini yapılan

verilerin doğruluğunu ölçmek amacıyla test grubu olarak atanmıştır. Bu populasyondaki hayvanların hepsi high density panel için kullanılabilecek genotipler içermektedir. Low density panel verileri ise, üzerinde kayıp veri tahmini yapılması için bırakıldı. Referans popülasyon olarakta 2006 öncesi doğmuş 4732 hayvandan oluşan bir küme seçilmiştir. Bu hayvanların genotipleri de high density panelde bulunmaktadır. İşlem sonucunda test grubu içerisindeki mevcut verilerlle tahmini yapılmış genotip verisi arasındaki uyumluluğun kromozomdan kromozoma farklı olmakla birlikte ortalama uyumun %95 olduğu görüldü. Tüm SNP ler içerisinde ortalama alel uyumluluğunun ise %97 olduğu görüldü. Gerçek ve tahmin edilmiş genotipler kullanılarak 15 tane fenotipik özellik için yapılan genomik tahmin işlemleri arasında da çok fazla fark görülmedi.

Pausch ve arkadaşları (2013) tarafından yapılan çalışmada ise 797 adet erkek sığırdan elde edilen 639214 SNP içerisinden **BovineSNP50 Bead chip** SNP'leri çıkarılarak elde edilen veri seti üzerinde kayıp veri tahmini yapılmak suretiyle çıkarılan verilerin elde edilmesi hedeflenmiştir. Kayıp veri tahmini için Hidden Markov Model tabanlı Beagle, findhap.f90, Mach ve Minimac araçları kullanılmıştır. Bu araçlar içinde sırasıyla 50, 100, 200 ve 400 hayvandan oluşan referans popülasyonlar değerlendirilmiştir. Referans popülasyonlardaki hayvanlar toplam popülasyonlarının sırasıyla %78.03, %89.21, %97.47 ve %99'nu oluşturmaktadır. Çalışma sonucunda kayıp genotip veri tahmin işleminin doğruluğunun hayvan sayısı ve referans popülasyon arttıkça arttığı gözlemlenmiştir. Popülasyon tabanlı algoritmalarla en güvenli sonuç 50 ve 100 referans hayvanın kullanıldığı senaryolar ile gerçekleştirilmiştir.

Akraba bireyler arasında paylaşılan haplotip miktarı fazla olduğu için popülasyon bazlı kayıp genotip veri tahmini daha yüksek doğruluk oranıyla gerçekleşebilecektir. Sargolzaei ve ark. (2014) göre klasik **Hidden Markov Model** tabanlı kayıp genotip tahmin işlemlerine populasyondaki canlıların köken bilgisi (*pedigree*) eklenerek yüksek başarımlı kayıp genotip tahminlemesi gerçekleştirilebilmektedir.

Moghaddar ve ark. (2015) tarafından safkan ve melez merinos koyunlar üzerinde yapılan çalışmada melez ve safkan koyun genotiplerinden rastgele oluşturulan bir referans genotip panel ve **Hidden Markov Model** tabanlı **Beagle** uygulaması kullanılmıştır. **Hidden Markov Model** oluşturulan sentetik referans panel kullanılarak

melez ve safkan koyunlardan oluşan gruplara ayrı ayrı uygulanmıştır. Safkan koyunlardan oluşan test gruplarından ortalama 90% - 95% arası başarım elde edilirken melez koyunlardan oluşan gruplarda bu oran 78% - 87% dolaylarında kalmıştır.

Hidden Markov Model olasılık hesapları ve hesaplanan olasılık değerlerine göre gizli durumların belirlenmesi için **The Forward** ve **Viterbi** algoritmalarını kullanır.

2.3.2.1. The Forward Algoritması

The Forward algoritması gözlemlenen dizilim durumlarının olasılıklarının tüm gözlemlenen ve gizli dizilim dizilim durumlarının ortak olasılık toplamlarından elde edilmesi mantığına dayanır. Bu durumda verilen

$$P(y_{1:T},\ x_{1:T}) = \pi_{x1} b_{x1,y1} \prod_{t=2}^{T} a_{xt-1,xt} b_{xt,yt} \tag{10}$$

$$P(y_{1:T}) = \sum_{x_{1:T}} P(y_{1:T}, x_{1:T}) \tag{11}$$

(10) denkleminden elde edilen olasılık değerlerinin (11) denklemindeki gibi toplamıyla da gizli dizilim durumundan gözlemlenen dizi durumuna geçiş olasılığı hesaplanır (Tataru ve ark., 2013).

2.3.2.2. Viterbi Algoritması

Viterbi algoritması Hidden Markov Model içerisinde gizli dizilimlerin bulunması için kullanılır. Kayıp dizilim The Forward algoritmasıyla bulunan geçiş olasılık değerlerinden en yüksek değere sahip dizilim alınır (Tataru ve ark., 2013). Yöntem

$$\omega_t(h_i) = b_{h_i,y_t} \,.\, max_j \left\{ \omega_{t-1} \left(h_j \right) a_{h_j,h_i} \right\} \tag{12}$$

denklemini kullanmaktadır.

2.3.2.3. Mach Uygulaması

Mach, Michigan Üniversitesinde geliştirilen, Markov zinciri tabanlı bir haplotip ve kayıp genotip veri tahmin aracıdır. Mach, kayıp veri analizi için ilgili bireylerin aile, soy bilgisine (Pedigree Dosyası) ve referans panele ihtiyaç duyar. Kayıp veri analizi yapılan bireylerle aileden gelen genotip bilgisi (pedigree dosyası) ve referans paneldeki genotipler haplotipler şeklinde hizalanır.

Hizalama işleminden sonra kayıp veri bölgesinin doldurulması için sezgisel bir istatistik ve tahmin algoritmasına ihtiyaç duyulur. Markov zinciri tabanlı istatistiksel algoritmalar haplotiplerin hizalanması sonrası kayıp veri bölgelerinin doldurulması noktasında ihtiyaç duyulan sezgisel yöntemleri sunar (Ellinghaus ve ark., 2009).

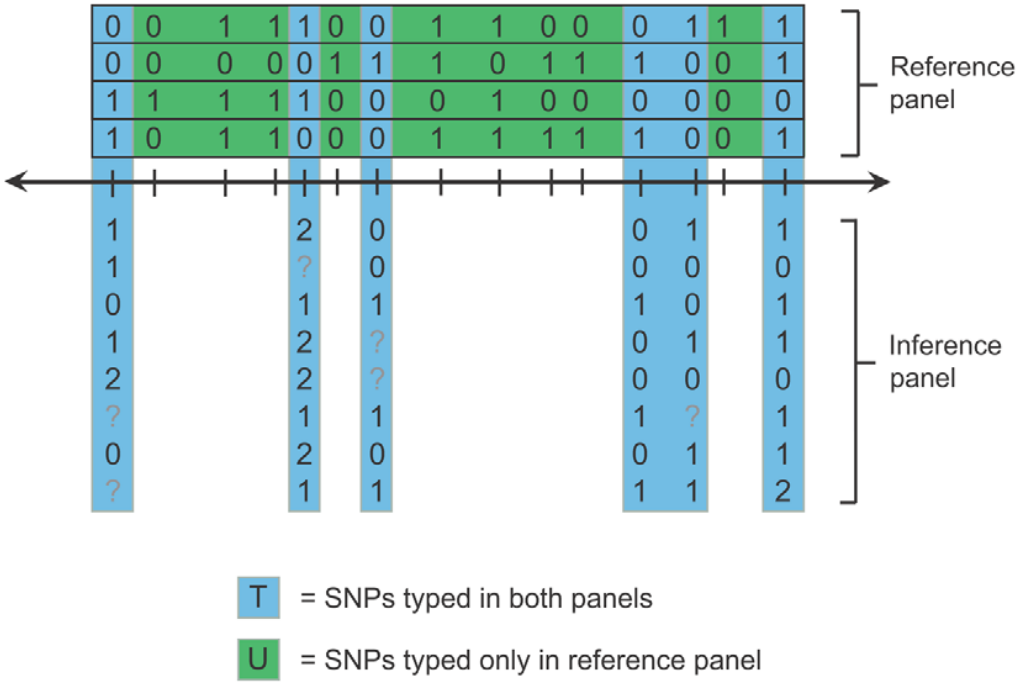
2.3.2.4. Impute Uygulaması

Impute, Oxford Üniversitesi bünyesinde geliştirilen, Markov zinciri tabanlı bir kayıp genotip veri tahmin uygulamasıdır. Çalışma prensibi olarak Mach ile benzerdir. Kayıp veri tahmini için referans haplotip panele ihtiyaç duyar. Referans paneldeki haplotip dizileri eldeki bireylere ait kayıp veri bölgelerini de içeren haplotip dizileriyle Şekil 2.5.'de olduğu gibi hizalanır. Sonrasında Markov Zinciri'nin doğası gereği kayıp veri bölgelerinin olasılıkları hesaplanır. Her iterasyonda kayıp veri bölgeleri, hesaplanan olasılık değerleri göz önünde bulundurularak ve referans haplotip dizisiyle tutarlı olmak koşuluyla doldurulur. Markov modelleri hafızasız sistemler olduğu için bir sonraki adım geçmiş adımlardan bağımsız olmak koşuluyla sadece içinde bulunulan durumdan etkilenmektedir. Bu durumda veri bölgeleri doldurulduğça diğer kayıp veri bölgelerinin olasılıklarında da değişimler olmaktadır. Böylelikle her iterasyonda veri setindeki tüm haplotipler iki adımda güncellenmektedir (Howie ve ark., 2009);

1. $H^T_{S,i}$ i bireyine ait bir haplotip dizisi, $G^T_{s,i}$ i bireyine ait multi lokus genotip, $H^T_{s,(-i)}$ i bireyi haricindeki bireylerin ilgili bölgedeki genotipi, H^T_R ilgili

bölgedeki referans genotip verisi, p ilgili böldegedeki rekombinasyon haritası olmak üzere Pr($H^T_{S,i}$ | $G^T_{s,i}$, $H^T_{s,(-i)}$, H^T_R, p) şartlı dağılımı belirlenir.

2. Pr şartlı dağılımına göre bireylerin kayıp veri bölgeleri doldurulur.



Şekil 2.5. Kayıp genotip veri tahmin senaryosu

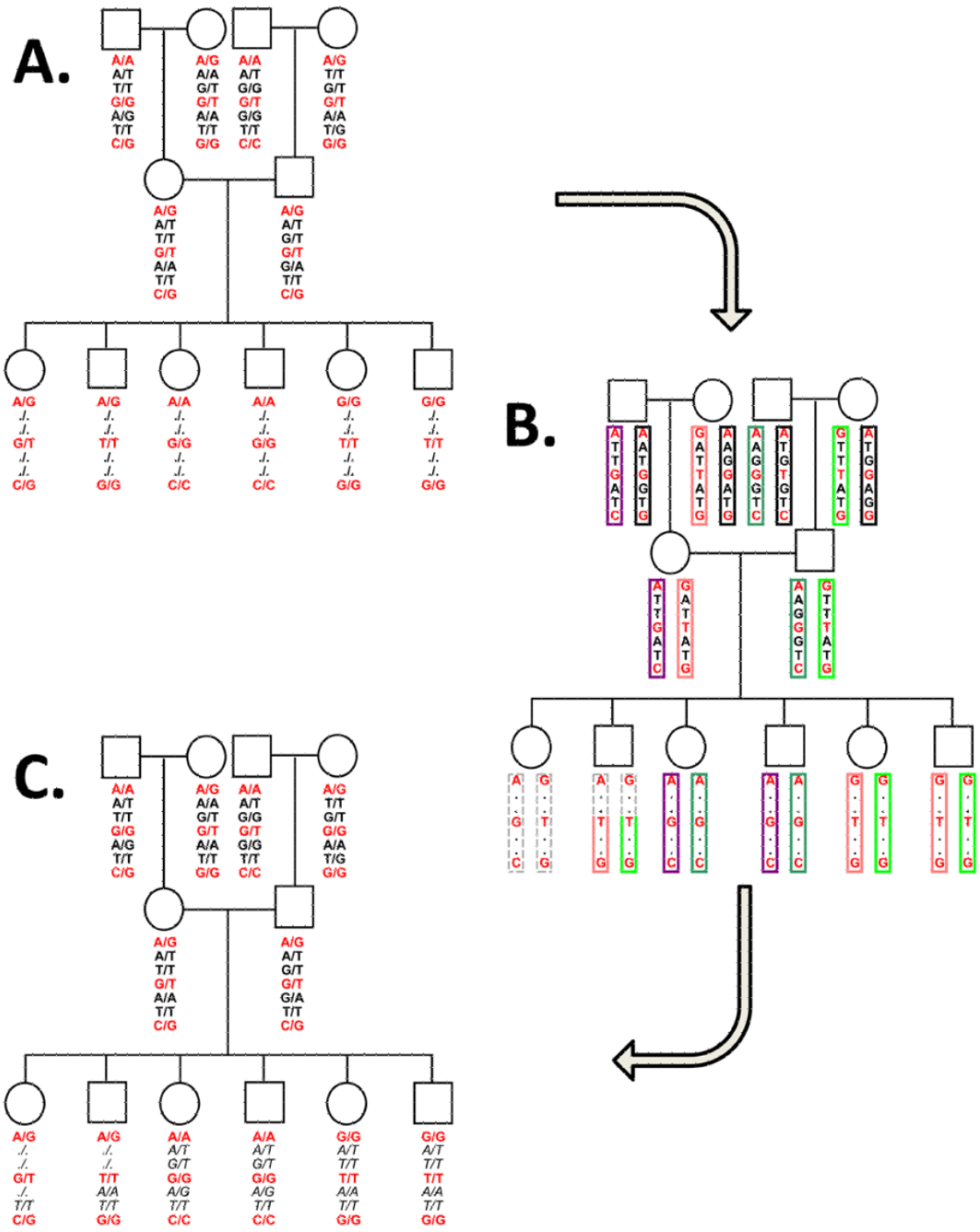
2.4. Akraba Bireyler Arası Kayıp Genotip Verisi Tahmini

Li ve ark. (2009), yaptıkları çalışmada akraba bireyler'in genotiplerindeki benzerlik ve aralarında paylaşılan uzun haplotipleri kullanarak, Kayıp Veri Tahmini için sezgisel bir seçenek sunmuşlardır. Bu haplotipler genotip materyallerin arasındaki bağlantının tipik bir kanıtı olarak kabul edilir. Aslında **genetik bağlantı** (*linkage*) ile, aynı kromozom bölgesini paylaşan akraba bireyler arasındaki benzerliğin aynı kromozom bölgesini paylaşmayan ve akraba olmayan bireylerden daha fazla olduğu kastedilir.

Akraba bireyler arası kayıp veri tahmini için örnek olarak Şekil 2.6. deki veri seti incelendiğinde ortak atadan türemiş nesillere ait genotip verileri görülür. Şekil 2.6. nın A bölümünde görüldüğü gibi ilk iki nesle ait genotip verisi çıkarılmış olmakla beraber, son jenerasyona ait genotip verilerde bazı kayıpların yaşandığı görülür. Burada veri kaybı yaşanan bireylerin birbirleriyle akraba oldukları bilindiği için bu bireyler haplotip olarakta büyük benzerlikler göstereceği sonucuna varılır. Tüm jenerasyonlar

dahil olmak üzere eldeki genotipler haplotipler şeklinde Şekil 2.6.'nın B bölümünde olduğu gibi ayrıştırılır. Veri kaybı yaşanan jenerasyona ait, içerisinde eksik veri barındıran haplotipler ata haplotiplerle karşılaştırılarak yine Şekil 2.6.'nin C bölümündeki genotiplere ulaşılır.

Buradaki akraba bireyler yaklaşımı sezgisel bir yaklaşımdır. Temelde matematiksel bir modele dayanmaz. Bunun yerine haplotip karşılaştırmaları yapılarak bir karar mekanizması ifade edilir. Akraba bireyler yaklaşımı, uzun haplotipleri paylaşan akraba bireylere dayandığı için neredeyse tüm **bağlantı analiz** (*linkage analysis*) metodlarını da destekler. Bundan dolayı ilk kayıp veri tahmini yaklaşımları, aynı atadan gelen bireylerin genotip dağılımı üzerinde durmuşlardır.



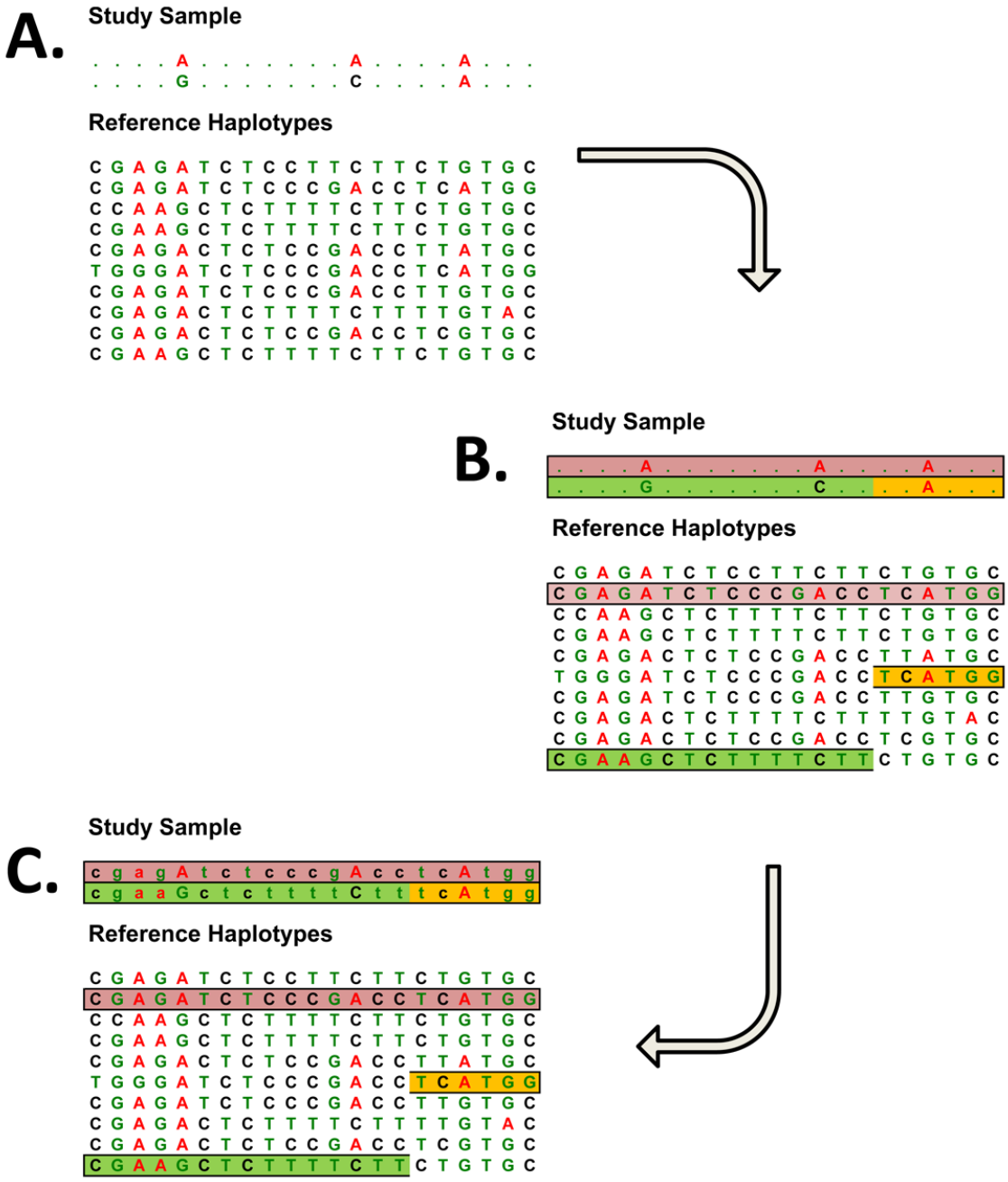
Şekil 2.6. Ortak atadan gelen bireylere ait genotip veri örneği (Li ve ark., 2009)

2.5. Akraba Olmayan Bireyler Arası Kayıp Genotip Verisi Tahmini

Akraba bireyler arasında yapılan Kayıp Genotip Verisi analizi sezgisel bir yaklaşıma dayanır. Bir canlının bir kromozomunun incelenmesi o canlının genotipi hakkında bilgi vermesinin yanı sıra aynı atadan gelen başka canlıların genotipleri hakkında da bilgiler verir. Bunun sebebi de akraba bireylerin genotiplerindeki haplotip benzerliğidir. Akraba bireyler arası Kayıp Genotip Veri Tahmininde temel dayanak noktası haplotip bazında benzerlik yaklaşımıdır.

Akraba olmayan bireyler arası Kayıp Genotip Veri Tahmini içinde akraba bireylerdeki sezgisel yaklaşım kullanılabilir. Fakat akraba olmayan bireyler için haplotipler karşılaştırılırken bireylerin atalarının genetik olarak birbirinden uzak olması sebebiyle daha kısa haplotip uzunlukları kullanılır. Bu yüzden haplotiplerin tanımlanması ve analiz edilmesi zorlaşır.

Akraba olmayan bireylerde Kayıp Genotip Veri Tahmini için kayıp bölgeler referans panelle karşılaştırılır. Karşılaştırma işlemi haplotip bazında olmakla beraber daha kısa haplotip uzunlukları kullanılır. Kayıp veri bölgelerini içeren haplotipler referans paneldeki genotip verisiyle karşılaştırılıp en uygun haplotip dizisi kayıp veri bölgelerine kopyalanır (Şekil 2.7). Li ve ark. (2009) tarafından yapılan, Avrupa populasyonun dan gelen bireyler için Kayıp Genotip Veri Tahmini çalışmasında (100-200 kb) uzunluğundaki haplotip dizileri, HapMap CEU panel'de paylaşılan haplotiplerle karşılaştırılmıştır (Li ve ark., 2009).



Şekil 2.7. Akraba olmayan bireyler arasında yapılan Kayıp Genotip Veri Tahmini (Li ve ark., 2009)

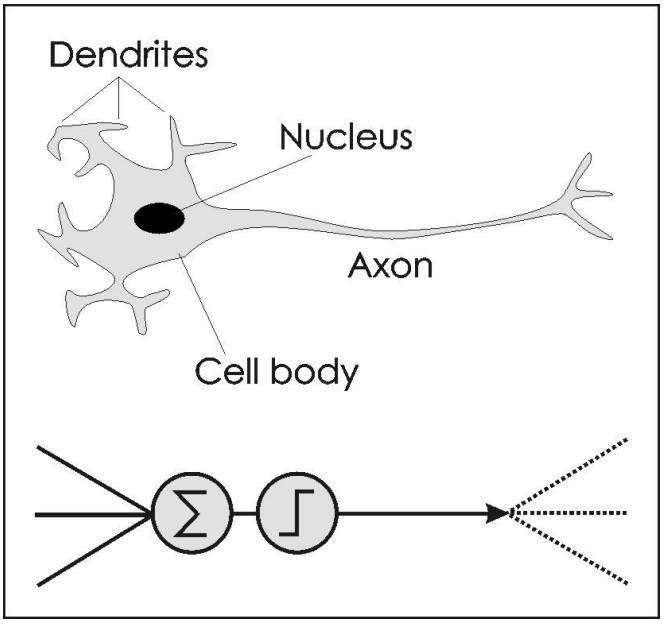
Referans haplotip dizi olmaksızın birey genotipleri kendi aralarında hizalanarak ta kayıp genotip veri tahmini yapılabilmektedir. Fakat bu tür bir yöntemde kullanılan bireylerin kayıp genotip veri oranı yüksek olmamalıdır. Aksi halde tahmin başarısında düşüşler yaşanabilmektedir (Browning, 2008).

2.6. Yapay Sinir Ağı

Zihinsel etkinliklerin, beyin hücrelerinde bulunan sinir ağlarının temelini oluşturan nöron (neuron) adı verilen birimlerin birbirleriyle olan elektro kimyasal etkileşimler ile gerçekleştiği düşüncesi temel alınarak McCulloch ve Pitts (1943) tarafından bir nöronun matematiksel modeli ortaya konmuştur. Biyolojik olarak bir nöronun aktiviteleri temelde aşağıdaki bileşenden oluşmaktadır;

- **Dendrit** (*Dendrite*): Sinir hücrelerinin uç kısımlarında yer alan dendritler farklı sinir hücrelerinden gelen sinyalin hücre çekirdeğine taşınmasından sorumludur.
- **Hücre Gövdesi (***Cell body***)**: Sinir hücresinin merkezi kabul olarak kabul edilen hücre gövdesi, dendritler den gelen sinyallerin toplandığı bir merkezdir.
- **Akson** (*Axon*): Hücre gövdesinden sinapslara kadar uzanmakla birlikte hücre gövdesinden gelen bilgilerin farklı bir sinir hücresine iletirmesinden sorumludur.
- **Sinaps** (*Synapses*): Sinir hücrelerinin, dendritlerle zıt yöndeki uç kısımlarında yer alan sinapslar, aksonlardan gelen bilgilerin, bir eşik değerle kıyaslanarak normalize edilmesi şeklinde gerçekleşen bir işlem sonrası farklı sinir hücrelerine iletilmesinden sorumludur.

Biyolojik bir nöronun çalışma prensibi ve bileşenleri taklit edilerek yapay sinir kavramı ortaya konmuştur (Şekil 2.8) (Rosenblatt, 1962).



Şekil 2.8. Biyolojik nöron ile yapay sinir arasındaki ilişki

Yapay sinir ağları (YSA), insan beyninden esinlenilerek geliştirilen, biyolojik sinir ağlarını taklit eden, öğrenme, ezberleme ve bilgiler arası ilişki kurma yeteneğine sahip sistemlerdir. YSA lar bağlantı ağırlıkları ile birbirlerine bağlanan işlem elemanlarından oluşur. Bu işlem elemanlarının her biri kendi belleğine sahip olmakla beraber diğer işlem elemanlarına aralarındaki bağlantı nispetince bağlanabilmektedir. Bir yapay sinir aşağıdaki bileşenlerden oluşur (Elmas, 2011);

- **Giriş Değerleri** (*Input*): Giriş değerleri (x_1, x_2,....., x_n) çevreden alınıp sinire iletilen bilgilerdir.
- **Ağırlıklar** (*Weight*): Ağırlıklar (w_1, w_2, ……., w_n), bir yapay sinire gelen bilgilerin yapay sinir üzerindeki etkisini ifade eden katsayılar dır.
- **Toplam Fonksiyonu** (*Addition Function*): Sinire gelen her bir girişin ilgili ağırlıkla çapımlarının toplamının sapma (*bias*) değeriyle toplanması işlevidir (Şekil 2.9.).
- **Aktivasyon Fonksiyonu** (*Activation Function*): Aktivasyon fonksiyonunun amacı, toplam fonksiyonundan gelen bilginin aktivasyon fonksiyonunun eşik değerine göre normalize edilmesidir. Yani bir yapay sinirin, aktivasyon fonksiyonunun eşik değeri altında çıkış üretmesinin önüne geçmek istenmektedir.
- **Çıkışlar** (*Output*): Aktivasyon fonksiyonu sonucundan gelen, dış dünyaya veya sonraki yapay sinire iletilen bilgilerdir.

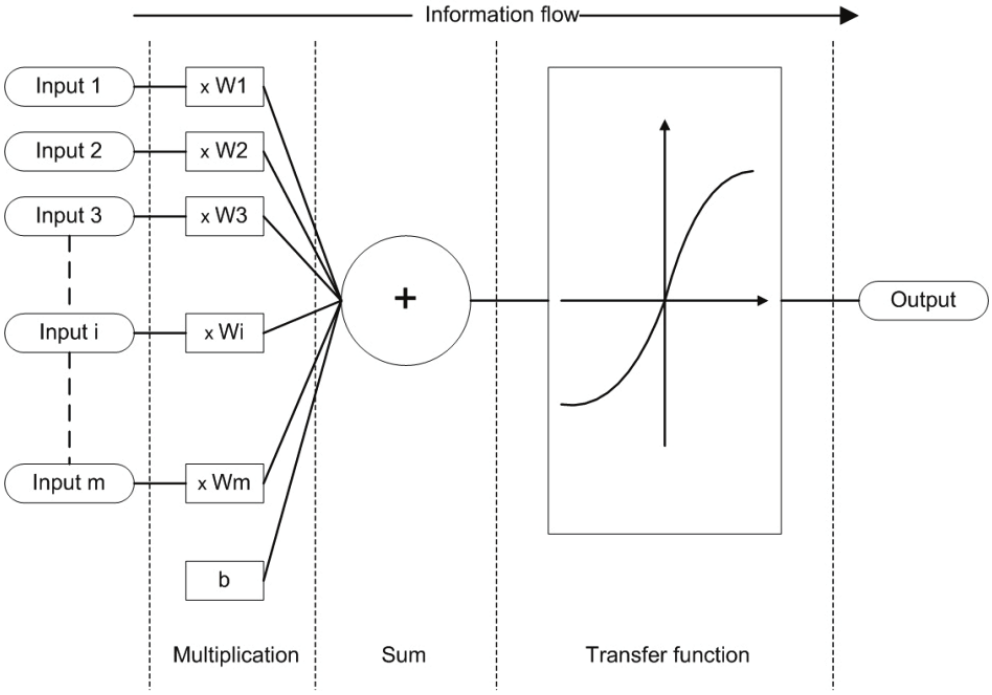
Bir yapay sinirin çalışma prensibinin matematiksel modeli aşağıdaki gibidir;

- n: Giriş değerlerinin sayısı
- x: Giriş değerleri
- w: Ağırlıklar
- b: Sapma
- f: Aktivasyon fonksiyonu
- y: Çıkış değerleri

olmak üzere,

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b \right) \tag{13}$$

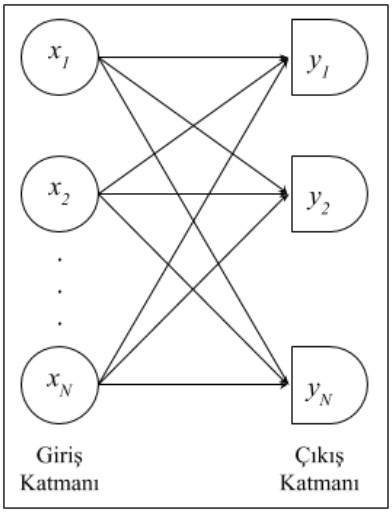
şeklindedir. Modelin çalışma şeması Şekil 2.9'da verilmiştir.



Şekil 2.9. Yapay sinir ağı çalışma modeli

2.6.1. Tek Katmanlı Yapay Sinir Ağları

Tek katmanlı yapay sinir ağları, her bir giriş nöronunun belli bağlantı ağırlıklarıyla tüm çıkış nöronlarına bağlanabileceği şekilde tasarlanmış YSA modelidir. Giriş nöronları çıkış nöronlarına tek bir katman içerisinde bağlanır. YSA'nın doğası gereği her bir çıkış nöronu için, girişlerin bağlantı ağırlıklarıyla çarpımları toplanır. Elde edilen sonucun bir aktivasyon fonksiyonuna tabi tutulmasıyla da ilgili nöronun çıkışı elde edilir. Tek katmanlı bir YSA Şekil 2.10'daki gibidir.

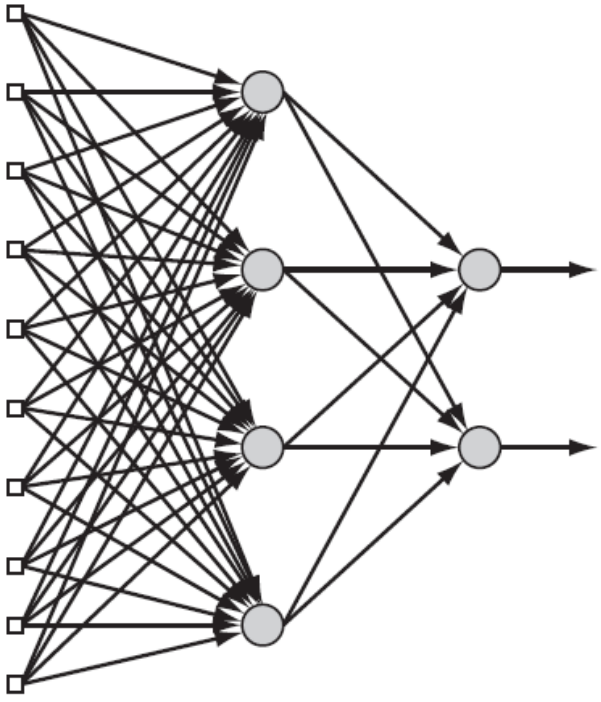


Şekil 2.10. Tek katmanlı yapay sinir ağı

2.6.2. Çok Katmanlı Yapay Sinir Ağları

Çok katmanlı YSA lar bir veya daha çok gizli katman içeren YSA modelidir. Gizli katmanlar tamamen hesaplama amaçlı olmakla birlikte giriş ve çıkışlar içinde giriş ve çıkış katmanları bulunmaktadır. Giriş katmanları dış Dünyadan gelen bilgilerin YSA ya girdiği katman, çıkış katmanı ise gizli katmanlardan elde edilen çıkış bilgilerinin dış Dünyaya aktarıldığı katmandır. Bir YSA da bir veya daha çok gizli katman kullanılması, YSA'nın hesaplama gücünü artırmakla birlikte daha yüksek seviyeli istatistiksel çıkarımlar yapılabilmesini sağlar. Her katmanın kendi giriş ve çıkışları bulunmaktadır. YSA boyunca her bir katmana ait çıkışlar bir sonraki katmanın girişi olarak kullanılır. Çok katmanlı YSA lar katmanlar arası bağlantı şekline göre;

- **Tam Bağlı Yapay Sinir Ağı** (*Fully Connected*): Her bir nörona ait çıkışın bir sonraki katmandaki tüm nöronlara giriş olarak uygulandığı YSA modelidir (Şekil 2.11).
- **Kısmi Bağlı Yapay Sinir Ağı** (*Partially Connected*): Bazı nöronların kendisinden sonraki katmanda bulunan nöronların sadece bir kısmına bağlı olacak şekilde tasarlanan YSA modelidir.



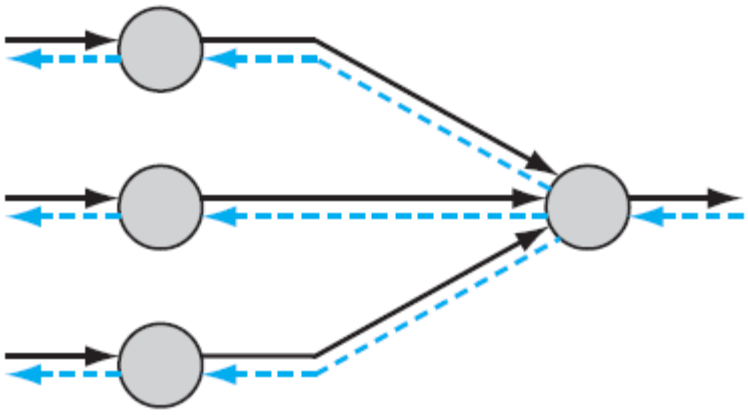
Şekil 2.11 Çok katmanlı tam bağlı bir yapay sinir ağı

Çok katmanlı YSA ların eğitimi için popüler iki yöntem mevcuttur. Bunlar;

- **İleri Doğru İşlem** (*feed-forward*): Sinaptik ağırlıklar sabit olmak üzere input sinyalleri YSA boyunca çıkış katmanına ulaşıncaya kadar katman katman yayılır. Yayılım boyunca standart YSA işlemleri gerçekleşir.
- **Geriye Yayılımlı İşlem** (*backpropagation*): Geriye yayılımlı YSA larda çıkış katmanı tarafından üretilen çıkış değerleri olması gereken değerlerle karşılaştırılır. Çıkış verisi istenenden farklı olması durumunda bir hata sinyali üretilir. Üretilen hata sinyali YSA boyunca ters yönde yayılır. Sonrasında ağırlıklar üzerinde düzenlemeler gerçekleştirilir ve yeni bir iterasyon başlatılır. Bu eğitim süreci çıkış değeri istenilen değere ulaşıncaya kadar devam eder.

Çok katmanlı bir YSA da tanımlı iki tür sinyal vardır. Bunlar (Haykin, 2009);

- **Fonksiyon Sinyali** (*function signals*): YSA nın giriş katmanına bir giriş sinyali olarak gelip, YSA boyunca yayılır ve çıkış katmanında da bir çıkış sinyali olarak ortaya çıkar.
- **Hata Sinyali** (*error signals*): YSA boyunca geriye doğru yayılan bir sinyal olmakla birlikte iterasyon sonucunda istenilen değerin elde edilemediği durumlarda ortaya çıkar (Şekil 2.12).



Şekil 2.12. Fonksiyon sinyali ve hata sinyali

Geriye yayılımlı işlemde, YSA çıkış verileri eldeki verilerle değerlendirilip giriş değerlerinin ağırlıkları iterasyonlar boyunca yeniden hesaplanarak yeni giriş değerlerinin oluşturulması YSA öğrenme performansını artırmaktadır. İterasyonlar da

çıkış değeriyle beklenen değer arasındaki hata oranının minimize edilmesi amaçlanmaktadır. Hata minimizasyonu için de en küçük ortalama kareler (*LMS-least mean squares*) için "eğim iniş (*gradient descent*)" metodu kullanılır. Ağırlık vektörü *w,* hedef değer *t* ve hesaplanan değer *z* olmak üzere hedef değer ile hesaplanan değer arasındaki hata;

$$J(w) = \frac{1}{2}(t - z)^2 \tag{14}$$

ile elde edilir. Ağırlıkların başlangıç değerleri rastgele seçilir ve öğrenme oranı (*learning rate*) η olmak üzere hata aşağıdaki matematiksel modelde olduğu gibi azaltılarak ilerlenir.

$$\Delta w = -\eta \frac{\partial j}{\partial w} \tag{15}$$

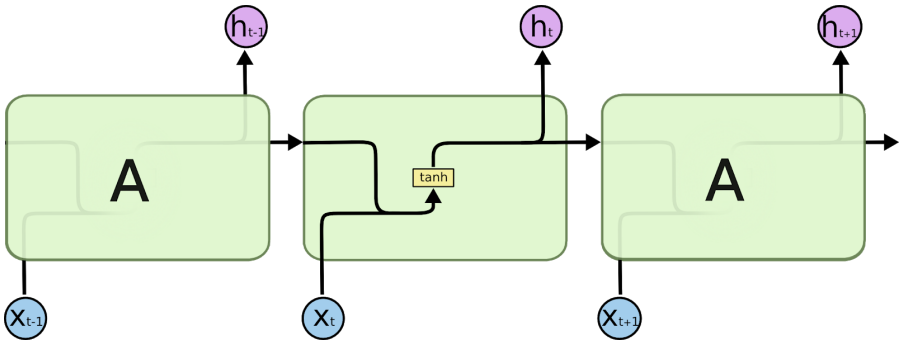
Elde edilen değer iterasyon yöntemiyle bir sonraki ağa aşağıdaki gibi aktarılır (Duda ve ark., 2012):

$$w(m + 1) = w(m) + \Delta w(m) \tag{16}$$

2.6.3. Uzun-Kısa Süreli Hafıza

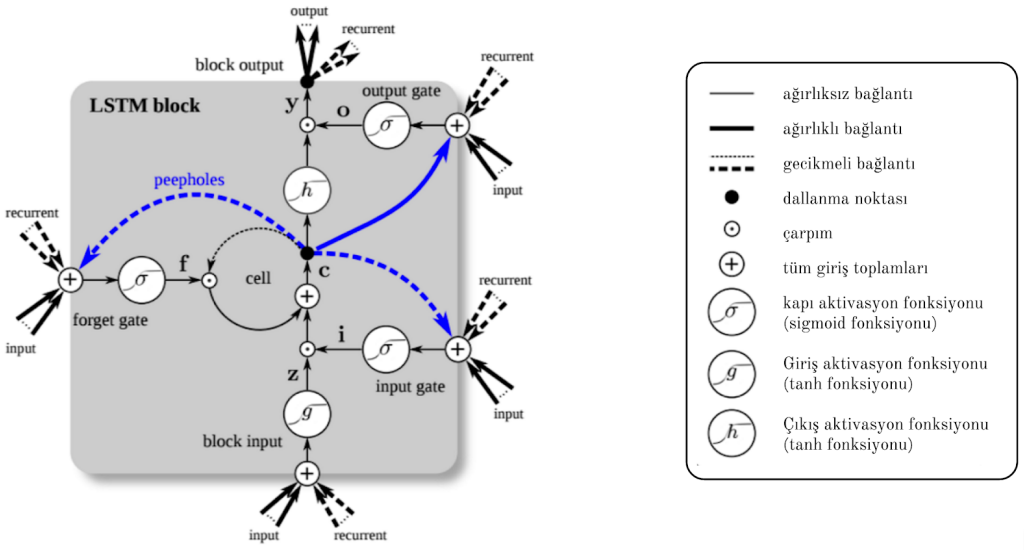
Uzun - Kısa Süreli Hafıza (*Long Short Term Memory - LSTM*) adından da anlaşılacağı üzere uzun zamanlı öğrenme yeteneğine sahip bir Tekrarlamalı (*Recurrent*) Yapay Sinir Ağıdır. İlk defa Hochreiter ve Schmidhuber (1997) tarafından geliştirilmiştir. Uzun zaman dilimi içerisinde bilginin hatırlanmasına dayanır. Bu, doğal bir yaklaşım olmasına rağmen, öğrenme işlemi için gereken bir davranış değildir.

Uzun - Kısa Süreli Hafıza yaklaşımının temel amacı uzun periyotlu hafıza probleminin çözülmesi dir. Yapı itibariyle Tekrarlamalı Sinir Ağı, bilgiyi tekrarlayan bir modüller dizisine sahiptir. Tekrarlamalı bir Yapay Sinir Ağı tekrar eden modüller için tek bir aktivasyon fonksiyonu (**tanh**) kullanan tek bir katmandan oluşan basit yapıda bir mimariye sahiptir (Şekil 2.13.) (Olah, 2015).



Şekil 2.13. Tekrarlamalı Sinir Ağı içerisinde tek bir aktivasyon fonksiyonu ve tek katmandan meydana gelen bir birim

Uzun - Kısa Süreli Hafıza ise mimari yapı olarak farklılık arzeden tekrarlı bloklardan oluşan bir diziye benzeyen bir ağ şeklindedir (Şekil 2.14.) (Greff ve ark., 2015).



Şekil 2.14. Uzun - Kısa Süreli Hafıza blok çalışma şeması

2.6.4. ReLU

Yapay Sinir Ağı uygulamalarında nöronların çıktı değerleri için aktivasyon fonksiyonu olarak genellikle tanh ve sigmoid fonksiyonları kullanılmaktadır. Fakat bu iki fonksiyon da yavaş çalışmaktadır. Bu sebeple Yapay Sinir Ağı uygulamalarında bu tür yavaşlıklardan kaçınmak için **Düzeltilmiş Doğrusal Birim** (*ReLU- Rectified Linear Unit*) kullanılmaktadır. ReLU fonksiyonu;

$$f(x) = max(0, x) \tag{17}$$

olarak ifade edilir (Hinton ve ark., 2010).

2.6.5. Dropout

Gizli katmanlarda bulunan nöronların çıktı değerlerini öncesinde belirlenmiş olan bir olasılık değeri ile sıfıra yakınsayarak **uyumsuzluğu** (*over fitting*) önlemede kullanılan bir yaklaşımdır (Krizhevsky ve ark. 2012, Szegedy ve ark., 2015).

2.6.6. Softmax

Yapay Sinir Ağı uygulamalarında çoğunlukla sinir ağının en son katmanı olarak kullanılan bir katmandır Softmax katmanında, boyutu K olan z vektörünün ve değeri 0 ile 1 arasında değişkenlik gösteren farklı bir vektöre dönüştürür.

2.7. Kullanılan Teknolojiler

2.7.1. Python

Python nesneye dayalı, platform bağımsız, açık kaynak kodlu olarak geliştirilip dağıtılan üst seviye bir programlama dilidir. Dinamik bir programlama dili olarak Python derlemeli değil, yorumlamalı bir programlama dilidir. Bu sebeple bir defa yazılan kod üzerinde herhangi bir değişiklik yapılmaksızın farklı işletim sistemlerinde çalışabilmektedir.

Esnek bir programlama dili olduğu için web, masaüstü, mobil uygulamaların yanı sıra bilimsel hesaplama işlemlerinde de sıkça tercih edilmektedir. Bu esneklik Python a farklı disiplinlerden gelen geniş bir geliştirici kitlesi kazandırmıştır.

Geniş modül desteği, Python programlama dilinin bir diğer avantajıdır. Geniş geliştirici kitlesi tarafından farklı amaçlar için geliştirilmiş birçok Python modül'ü pip adı verilen paket yöneticisi tarafından işletim sistemi içerisindeki Python dizinine yüklenerek Python projeleri içerisinde kullanılabilmektedir. Bu çalışmanın Makine Öğrenmesi bölümünde gerçekleştirilen uygulamalar Python programlama dili ve aşağıda belirtilen Python modül ve paketleri kullanılarak gerçekleştirilmiştir.

2.7.2. Numpy

NumPy, çok boyutlu matrislerde matematiksel işlemler gerçekleştirmek amacıyla geliştirilmiş bir Python modülüdür. Matris oluşturma, matris toplamı, matris çarpımı gibi matematiksel işlemler optimize bir şekilde gerçekleştirilerek projelerdeki kod fazlalığını ortadan kaldırmaktadır. Ayrıca bünyesindeki hesaplama fonksiyonları C, C++ gibi derlemeli daha alt seviye diller ile geliştirildiği için NumPy kullanılarak yapılan hesaplamalar, standart Python ile yapılan hesaplamalara göre daha performanslı hesaplama yapmaktadır (Van Der Walt ve ark., 2011).

2.7.3. Theano

Matematiksel ifadelerin tanımlanıp, optimize edilerek çalıştırılmasına yarayan bir Python modülüdür. NumPy ile entegre çalışabilme özelliğine sahip olan Theano ile NumPy dizileri direkt olarak kullanılabilmektedir. Theano ile tanımlanan matematiksel ifadeler hem CPU, hemde GPU üzerinde direkt olarak çalıştırılabilmektedir. Derin öğrenme gibi Yapay öğrenme algoritmalarının bilgisayarda tasarlanıp çalıştırılmasına olanak sağlayan arayüzlere sahiptir (Bergstra ve ark., 2010). Bu çalışmada yine bir Python modülü olan Kerasın bir bağımlılığı olarak kullanılan Python modüllerinden bir taneside Theano dur.

2.7.4. Keras

Keras, Python programlama dili ile geliştirilen Yapay Sinir Ağı kütüphanesidir. Theano ve Tensorflow, Kerasın çalıştırılabilmesi için gereken bağımlılıklardır. Theano sayesinde CPU üzerinde çalışabildiği gibi GPU üzerinde de çalışabilmektedir. Keras esnek bir araçtır. Öyleki, kendi bünyesinde sunduğu aktivasyon, öğrenme gibi algoritmaların yanı sıra geliştiriciye kendi aktivasyon fonksiyonunu ve öğrenme algoritmasını geliştirme avantajı sunmaktadır (Chollet, 2015).

3. MATERYAL VE YÖNTEM

3.1. Kayıp Genotip Veri Tahmini İçin Yapay Öğrenme ve İstatistiksel Algoritmalar

Kayıp veri tahmini için en çok kullanılan istatistiksel model olan **Hidden Markov Model** ve yine kayıp veri tahmini işlemlerinde sıkça kullanılan bir yapay öğrenme algoritması olan **KNN** (*K Nearest Neighbor*) algoritması belirlenen veri setleri üzerinde uygulanarak alınan sonuçlar karşılaştırılmıştır.

3.2. Veri Seti

Tez çalışması kapsamında kullanılan biyolojik veri patlıcan (*S. melongena*) (RIL) rekombinant saf hat populasyonunda (GBS) dizileyerek genotipleme yöntemi ile yeni nesil yüksek veri çıkışlı DNA sekanslama platformu olan HiSeq2000 sequencer (Illumina, Inc., San Diego, CA, United States) cihazında üretilmiş ve NCBI veritabanında "Eggplant SRA" sekmesinde SRX461583 veri giriş etiketinden indirilmiştir. GBS çalışması için genomik kütüphaneler ApeKI restriksiyon enzimi kullanılarak hazırlanmış ve çalışma kapsamında geliştirilen ham SNP markör verisi; popülasyon içerisindeki minimum SNP alel frekansı %5 ve maksimum heterozigot alel miktarı %15 olacak şekilde filtrelenmiştir. Veri seti içerisinde farklı bireylere ait genotipler kromozom üzerindeki lokasyonu ve referans genotip bilgisi de dahil olmak üzere her kromozom için ayrı ayrı kaydedilmiştir (Çizelge 3.1). Burada "N" ile ifade edilen bölgeler, SNP bazında kayıp genotip bölgelerini ifade eder. Veri setinin kayıp veri tahminleme öncesi 1. kromozomundaki SNP'leri Şekil 3.1. deki gibidir.

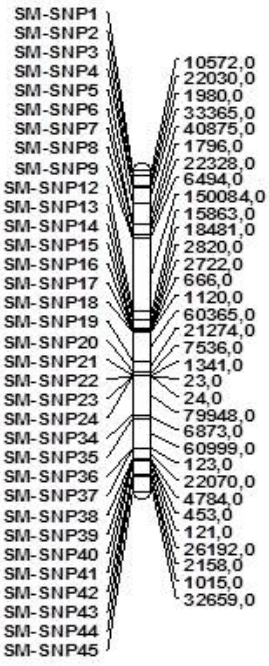
Çizelge 3.1. Patlıcan 1. kromozom veri setinden bir bölüm

Lokasyon	Referans	Birey 1	Birey 2	Birey 3	Birey 4	Birey 5
412739	T	N	N	N	N	Y
412749	G	N	N	N	N	C
412757	A	N	N	N	N	N
578559	T	C	N	N	N	N
590972	T	N	N	N	N	N
896003	A	A	T	T	N	N
954386	T	N	A	N	A	N
955094	G	G	T	N	T	T
960629	G	T	T	N	T	N
993974	T	N	N	N	N	C
1130285	A	T	N	T	T	N
1130286	A	T	N	T	T	N
1720056	T	A	A	N	N	N

Eldeki patlıcan bitkisine ait genotip verisindeki kayıp SNP miktarları Çizelge 3.2. deki gibidir.

Çizelge 3.2. Patlıcan veri seti kayıp SNP miktarları

Kromozom	**Kayıp SNP Miktarı**	**Toplam SNP Miktarı**	**Kayıp %**
Kromozom 1	21290	40766	52%
Kromozom 2	15122	28985	52%
Kromozom 3	16665	31042	53%
Kromozom 4	14786	27489	53%
Kromozom 5	14139	24123	58%
Kromozom 6	9315	16082	57%
Kromozom 7	16135	30294	53%
Kromozom 8	15843	29172	54%
Kromozom 9	12158	22627	53%
Kromozom 10	8992	15147	59%
Kromozom 11	16202	28424	57%
Kromozom 12	11212	21879	51%
Kromozom 13	11035	19261	57%
Kromozom 14	16620	27863	59%
Kromozom 15	12972	22253	58%
Kromozom 16	13382	22253	60%
Kromozom 17	11584	18700	61.000000



Şekil 3.1. Kayıp veri tahminleme öncesi veri setinin 1. kromozomundaki SNP'ler

Yapay Sinir Ağı yaklaşımında sinir ağını eğitmek için kullanılacak bir eğitim veri setine ihtiyaç duyulmaktadır. Eğitim amaçlı kullanılacağı için veri setinin eksiksiz olması gerekmektedir. Eksiksiz genotip verisi üreten hassasiyete sahip cihazların maliyetleri fazla olduğu için eksiksiz genotiplenmiş birey verileri bulunmamaktadır. Bu çalışmada eksiksiz genotip verisine sahip birden fazla birey eğitim amaçlı kullanılacağı için yine bu çalışmada uygulanan Hidden Markov Model yaklaşımından elde edilen çıktı verisi içerisinden 10 adet birey eğitim amaçlı ayrılmış, kalan her bir bireyin genotipinden 40 adet SNP silinerek üzerinde tahminleme yapılacak veris seti oluşturulmuştur.

3.3. Tassel Uygulaması

Tassel, genetik araştırmalar için geliştirilmiş çok yönlü bir araçtır. İstatistiksel yaklaşımlardan, kayıp veri tahminine, bağlantı dengesizliği ve genom bazlı ilişki araştırmalarına kadar birçok genetik analize olanak sağlayan etkili bir araçtır. Ayrıca Nature Genetik deki bazı makalelerin uygulamalarını sunmaktadır (Bradbury ve ark., 2007).

3.4. Beagle Uygulaması

Beagle, genotip fazlama ve kayıp genotip veri tahmini üzerinde özelleştirilmiş, Washington Üniversitesi tarafından geliştirilmiş, **Hidden Markov Model** tabanlı bir araçtır. Java programlama dili kullanılarak geliştirilmiştir. Açık kaynak lisansıyla (*GNU*) dağıtılmaktadır (Browning ve ark., 2018).

3.5. JoinMap Uygulaması

Genotip dizileri içerisinde yapılan kayıp veri tahmini işlemleri çoğunlukla alellerin birbirlerine olan bağımlılıklarını (*Linkage*) kullanır. Aynı şekilde genotip dizilerinde yapılan kayıp veri tahmin işlemlerinin doğruluğu da genetik bağlantıyla (*Linkage*) ölçülür.

JoinMap, diploid kromozom yapısına sahip popülasyonlar için genetik bağlantı (*Linkage*) haritalarını hesaplayan bir araçtır. Veri seti üzerindeki alelleri referans genom üzerinde test ederek kayıp veri tahmini işleminin başarımı konusunda da raporlar sunar (Stam, 1993).

3.6. KNN (K Nearest Neighbor) İle Kayıp Genotip Veri Tahmini

KNN algoritmasının doğası gereği öncelikle her bir kayıp SNP için K sayıda komşu SNP değeri bulundu. SNP lerin genom üzerindeki lokasyonları baz alınarak bir mesafe matrisi oluşturuldu. Son olarakta mesafe matrisi içerisinden en yakın mesafedeki nükleotit bazı kayıp SNP bölgesine yerleştirildi (Çizelge 3.3). **KNN** algoritması eldeki veri setine tassel isimli uygulama aracılığıyla farklı K değerleri için uygulandı (Şekil 3.2).

Çizelge 3.3. Kayıp Genotip Verileri için KNN algoritması

Algoritma 1: (KNN)

Girdi : Kayıp genotip bölgeleri ve referans genotip bilgisi içeren veri seti

Çıktı : Algoritma sonucuna göre kayıp bölgeleri doldurulmuş veri seti

---

Adım 1 : Başla

Adım 2 : Döngü (N adet kayıp SNP bölgesi)

- Herbir kayıp SNP için K sayıda komşu SNP bul

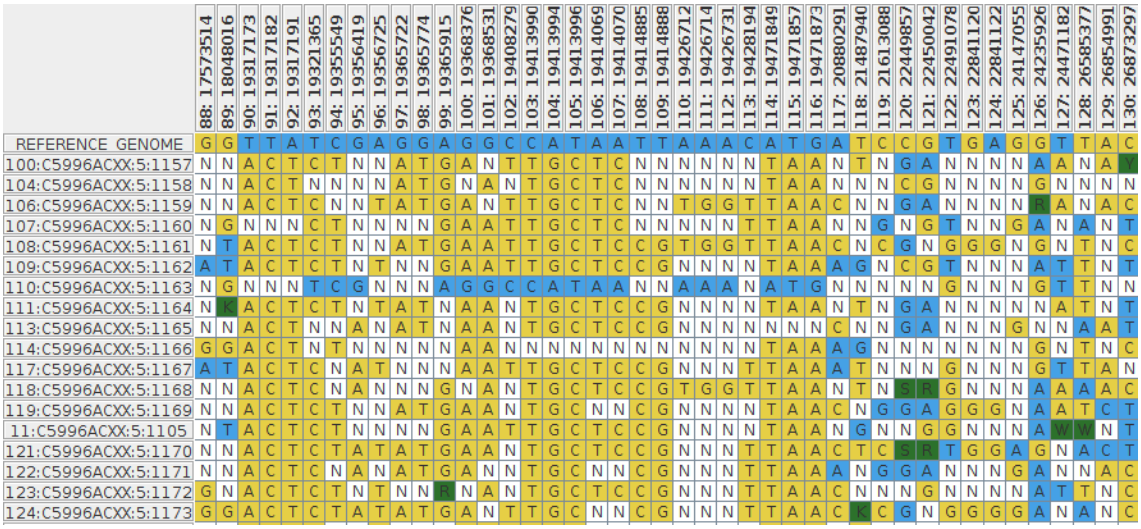
Adım 3 : Döngü (N adet kayıp SNP bölgesi)

- Herbir kayıp SNP için K sayıda komşu SNP le mesafe matrisi oluştur.

Adım 4 : Döngü (N adet kayıp SNP bölgesi)

- Herbir komşuluk matrisi içerisindeki en yakın SNP i bul.
- Bulunan SNP le kayıp bölgeyi doldur.

Adım 5 : Bitir.



Şekil 3.2. Tassel uygulaması

3.7. Hidden Markov Model İle Kayıp Genotip Veri Tahmini

Hidden Markov Modeli eldeki veri setine uygulayabilmek için öncelikle veri setini vcf formatına dönüştürmek gerekmektedir. Bu sebeple Tassel uygulaması aracılığıyla hapmap formatındaki genotip veri seti vcf formatına dönüştürüldü.

Vcf formatına dönüştürülen veri seti Beagle uygulamasına parametre olarak girilip Markov süreci başlatıldı (Şekil 3.3, Şekil 3.4).

```
java -jar beagle.08Jun17.d8b.jar gt=chr1.vcf out=out.gf
```

Şekil 3.3. Beagle ile Hidden Markov Model başlatılması.

Hidden Markov Model, olasılık dağılımına dayanan bir istatistiksel model olduğu için markov sürecinde olması gereken **The Forward** ve **Viterbi** algoritmaları veri seti üzerinde sırasıyla işletildi (Çizelge 3.4, Çizelge 3.5).

Çizelge 3.4. The Forward Algoritması

Algoritma 2: (The Forward)
Girdi : Kayıp genotip bölgeleri ve referans genotip bilgisi içeren veri seti **Çıktı :** Dizilim durumlarının geçiş olasılıkları
Adım 1 : Başla **Adım 2 :** Gözlemlenen dizilim durumlarını belirle. **Adım 3 :** Gizli dizilim durumlarını belirle. **Adım 4 :** Gözlemlenen durumlardan gizli durumlara geçiş olasılıklarını hesapla **Adım 5 :** Bitir

Çizelge 3.5. Viterbi Algoritması

Algoritma 3 : (Viterbi)

Girdi : Dizilim durumlarının geçiş olasılık matrisi

Çıktı : Algoritma sonucuna göre kayıp bölgeleri doldurulmuş veri seti

Adım 1 : Başla

Adım 2 : Döngü (N x M durum geçiş olasılıkları)

- Gözlemlenen durumlardan geçiş için maksimum olasılığa sahip gizli durumu bul.

Adım 3 : Bitir

```
beagle.08Jun17.d8b.jar (version 4.1)
Copyright (C) 2014-2015 Brian L. Browning
Enter "java -jar beagle.08Jun17.d8b.jar" for a summary of
command line arguments.
Start time: 12:59 AM EET on 24 Dec 2017

Command line: java -Xmx1749m -jar beagle.jar
  gt=genotype_data/hapmap/chr1.vcf
  out=genotype_data/outputs/chr1_beagle_out.txt

No genetic map is specified: using 1 cM = 1 Mb

reference samples:       0
target samples:        188

Window 1 [ 1:412739-50068972 ]
target markers:        218

Starting burn-in iterations

Window=1 Iteration=1
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 5      max edges/level: 11
```

```
mean edges/node:  1.880  mean count/edge: 75

Window=1 Iteration=2
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 5      max edges/level: 14
mean edges/node:  1.981  mean count/edge: 75

Window=1 Iteration=3
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 5      max edges/level: 12
mean edges/node:  1.846  mean count/edge: 75

Window=1 Iteration=4
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 6      max edges/level: 14
mean edges/node:  1.731  mean count/edge: 63

Window=1 Iteration=5
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 5      max edges/level: 13
mean edges/node:  1.783  mean count/edge: 75

Window=1 Iteration=6
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 6      max edges/level: 13
mean edges/node:  1.618  mean count/edge: 63

Window=1 Iteration=7
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 6      max edges/level: 12
mean edges/node:  1.639  mean count/edge: 63

Window=1 Iteration=8
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
```

```
mean edges/level: 7      max edges/level: 15
mean edges/node:  1.598  mean count/edge: 54

Window=1 Iteration=9
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 6      max edges/level: 13
mean edges/node:  1.616  mean count/edge: 63

Window=1 Iteration=10
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 7      max edges/level: 15
mean edges/node:  1.589  mean count/edge: 54

Starting phasing iterations

Window=1 Iteration=11
Time for building model:         0 seconds
Time for sampling (singles):     1 second
DAG statistics
mean edges/level: 11     max edges/level: 18
mean edges/node:  1.438  mean count/edge: 34

Window=1 Iteration=12
Time for building model:         0 seconds
Time for sampling (singles):     1 second
DAG statistics
mean edges/level: 14     max edges/level: 24
mean edges/node:  1.409  mean count/edge: 27

Window=1 Iteration=13
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 16     max edges/level: 25
mean edges/node:  1.382  mean count/edge: 24

Window=1 Iteration=14
Time for building model:         0 seconds
Time for sampling (singles):     0 seconds
DAG statistics
mean edges/level: 18     max edges/level: 29
mean edges/node:  1.363  mean count/edge: 21

Window=1 Iteration=15
```

```
Time for building model:          0 seconds
Time for sampling (singles):      0 seconds
DAG statistics
mean edges/level: 22     max edges/level: 34
mean edges/node:  1.290  mean count/edge: 17

Number of markers:                  218
Total time for building model: 1 second
Total time for sampling:       4 seconds
Total run time:                6 seconds

End time: 12:59 AM EET on 24 Dec 2017
beagle.08Jun17.d8b.jar (version 4.1) finished
```

Şekil 3.4. Beagle ekran çıktısı

3.8. Yapay Öğrenme İle Kayıp Genotip Veri Tahmini

Eğitim (*training*), test (*test*) ve doğrulama işlemleri için algoritmaları belirtilerek Python programlama dilinde uygulamalar geliştirilmiştir. Ayrıca bu çalışmanın makine öğrenmesi bölümünde kullanılan eğitim ve test sentetik verilerini esas veri üzerinden oluşturan uygulama da Python programlama dili kullanılarak geliştirilmiştir.

Öncelikle her bir kayıp genotip bölgesinin sınıflandırılması için sınıflar sisteme Çizelge 3.6 daki gibi tanımlanmıştır.

Çizelge 3.6. Kayıp genotip tahminleme için makine öğrenmesi sınıfları

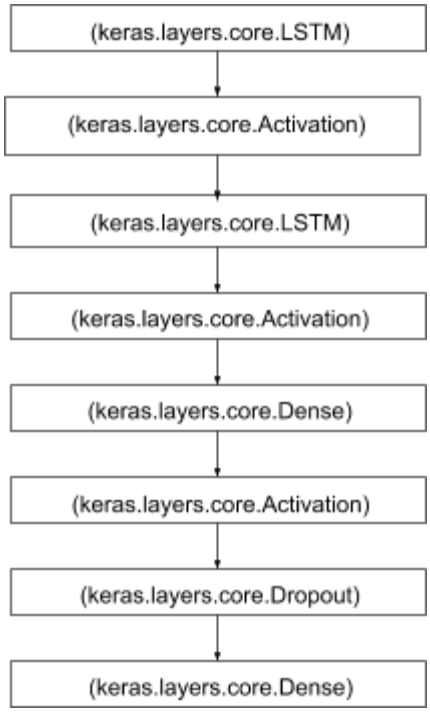
A	G	T	C	N	Y	S	K	W	M	R
0	1	2	3	4	5	6	7	8	9	10

Sisteme yüklenen veri seti 4 lü gruplara bölünerek, yani her bir snp kendisinden sonra sırayla gelen 3 SNP le bir grup haline getirilip, 4.SNP çıkış verisi olarak belirlenerek giriş ve çıkış vektörleri oluşturulmuş olup eğitim verisindeki hizalama bilgileri de sinir ağına 5. giriş olarak verilmiştir (Çizelge 3.7). Dörtlü grupların hizalama bilgisi olarakta her bir grubun ilk SNP nin birey dizilimi içerisindeki karakter indisi seçilmiştir. Eğitim verisinin ise %20 si test verisi olarak kullanılmıştır.

Çizelge 3.7. Eğitim verisi gruplama

Eğitim Grup					Çıktı
A	T	T	G	indis	C
T	T	G	C	indis	A

512 düğümden oluşan iki adet Uzun - Kısa Zamanlı Hafıza (*LSTM*) katmanı sonrasına 256 düğümlü yoğunluk (*Dense*) katmanı ve Dropout katmanı eklenerek Keras ortamında model oluşturuldu (Şekil 3.5). Sonrasında ise sisteme verdiğimiz eğitim veri seti ile Yapay Sinir Ağı eğitilip test edildi. Son olarak ta eldeki kayıp veri bölgeleri içeren sentetik veri seti üzerinde eğitilmiş sinir ağıyla tahminleme gerçekleştirildi.



Şekil 3.5. Sentetik genotip verisi için elde edilen model

4. ARAŞTIRMA SONUÇLARI VE TARTIŞMA

Kullanılan algoritmalar veri setindeki ilk dört kromozoma ait veri genotip dizileri üzerinde uygulanmış ve sonuçlar elde edilmiştir. Sonuçlar değerlendirilirken tahminleme sonrası markör sayıları ve markörlerin kromozom üzerindeki sıralamaları baz alınmıştır. Yani en fazla doğru sırada markörün elde edilmesi hedeflenmiştir.

Algoritmalardan elde edilen sonuçların başarısını ölçmek için öncelikle doldurulamayan SNP bölgesinin kalıp kalmadığı tez çalışması sırasında geliştirilen Python betikleri ile kontrol edilmiştir. Sonrasında tahmin edilen verilerin doğruluğunu ölçmek adına joinMap aracı ile bağlantı (*Linkage*) analizi yapılmıştır. Aynı zamanda kayıp veri tahmini işlemleri sonrasında kayıp SNP bölgeleri doldurulan genotip dizilerinden yeni bir sentetik veri seti oluşturulmuştur.

4.1. KNN Algoritmasının Mevcut Veri Seti Üzerindeki Sonuçları

KNN algoritması K'nın 7 ve 9 değeri için ayrı ayrı işletilmiştir.

4.1.1. K = 7 İçin KNN Algoritması

En yakın 7 komşuluk değeri için KNN algoritması mevcut veri setinin 1. kromozomuna uygulandıktan sonraki kayıp SNP, toplam SNP ve algoritma başarı durumu Çizelge 4.1. daki gibi olmuştur.

Çizelge 4.1. K'nın 7 olduğu durumda KNN algoritma sonucu

KAYIP SNP	TOPLAM SNP	TAHMİN EDİLEN SNP
5055	40766	76.25%

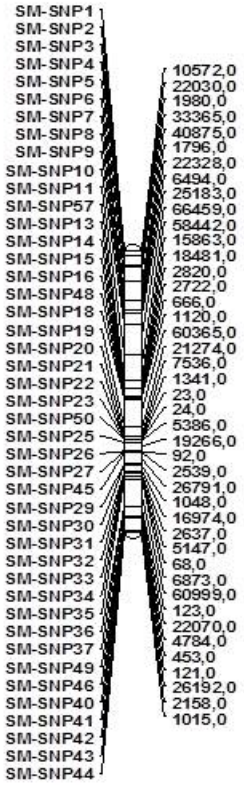
4.1.2. K = 9 İçin KNN Algoritması

En yakın 9 komşuluk değeri için KNN algoritması mevcut veri setinin yine 1. kromozomuna uygulandıktan sonraki kayıp SNP, toplam SNP ve algoritma başarı

durumu Çizelge 4.2. deki gibi olmuştur. KNN algoritmasının K=9 değer için 1. kromozomda bulduğu SNP'ler Şekil 4.1. deki gibidir.

Çizelge 4.2. K'nın 9 olduğu durumda KNN algoritma sonucu

KAYIP SNP	TOPLAM SNP	TAHMİN EDİLEN SNP
5055	40766	76.25%



Şekil 4.1. KNN algoritmasının 1. kromozomda k=9 için bulduğu markörler

Mevcut veri setinde KNN tahminleme öncesi ve sonrası marker sayıları Çizelge 4.3 deki gibidir. Bağlantı (*Linkage*) analizi sonucunda Şekil 4.1 de görüldüğü gibi bazı markörlerin (SM-SNP-45, SM-SNP-46, SM-SNP-48, SM-SNP-49, SM-SNP-50) kromozom üzerindeki sıralamalarında hata oluştuğu ve özellikle kromozomun alt bölgelerindeki markör yerleşimlerinin hatalı olduğu gözlemlenmiştir.

Çizelge 4.3. Veri setindeki markör sayıları

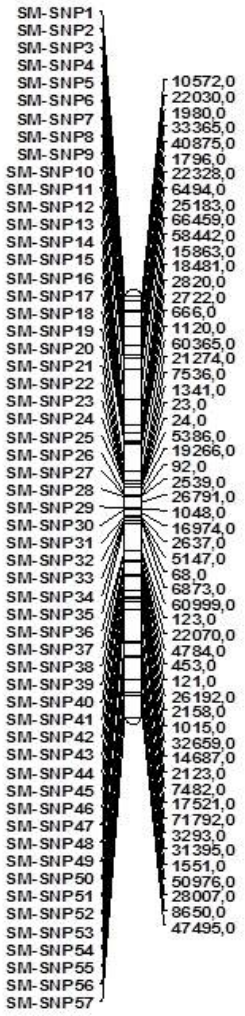
TAHMİNLEME ÖNCESİ MARKÖR SAYISI	TAHMİNLEME (KNN) SONRASI MARKÖR SAYISI
34	44

4.2. Hidden Markov Model in Mevcut Veri Seti Üzerindeki Sonuçları

Hidden Markov Model veri seti içerisinden kromozom 1, kromozom 2, kromozom 3 ve kromozom 4 üzerinde ayrı ayrı uygulanmıştır. Algoritmanın ilk 4 kromozom üzerindeki sonuçları Çizelge 4.4. deki gibidir. Hidden Markov Model in 1. kromozomda bulduğu SNP ler Şekil 4.2. deki gibidir.

Çizelge 4.4. Hidden Markov Modelin ilk 4 kromozom üzerindeki sonuçları

KROMOZOM	KAYIP SNP	TOPLAM SNP	TAHMİN EDİLEN SNP
Kromozom 1	0	40766	100%
Kromozom 2	0	28985	100%
Kromozom 3	0	31042	100%
Kromozom 4	0	31042	100%



Şekil 4.2. Hidden Markov Modelin 1. kromozomda bulduğu markörler

Bağlantı (*Linkage*) analizi sonrası Hidden Markov Model öncesi ve sonrası eldeki veri setinde bulunan markör sayısı Çizelge 4.5 deki gibidir. Bulunan markörlerin kromozom üzerinde doğru bir sırada yer aldığı bağlantı (*Linkage*) analizi sonrasında gözlemlenmiştir.

Çizelge 4.5. Veri setindeki markör sayırları

TAHMİNLEME ÖNCESİ	TAHMİNLEME (HMM) SONRASI
34	57

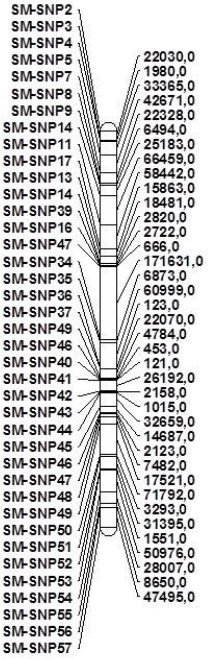
4.3. Derin Öğrenme Algoritmalarının Mevcut Veri Seti Üzerindeki Sonuçları

Materyal ve Yöntem bölümünde bahsedilen Derin Öğrenme algoritma mevcut veri seti üzerinde %57 lik bir öğrenme başarısına ulaşmıştır (Çizelge 4.6).

Çizelge 4.6. Derin Öğrenme algoritmasının mevcut veri seti üzerindeki öğrenme başarısı

Kromozom	Öğrenme Başarısı
1. Kromozom	57%
2. Kromozom	57%
3. Kromozom	57%
4. Kromozom	57%

Derin Öğrenme (*Deep Learning*) algoritmalarının bu çalışma kapsamında geliştirilen veri tahmin senaryosu ile eldeki sentetik veri üzerinde gerçeklenmesi sonucu elde edilen veri seti üzerinde JoinMap uygulamasıyla gerçekleştirilen bağlantı (*Linkage*) analizi sonuçları Şekil 4.3 deki gibidir.



Şekil 4.3. Derin Öğrenme ile 1. Kromozomda bulunan markörler

Derin Öğrenme (*Deep Learning*) sonrası elde edilen veri seti üzerinde bağlantı (*Linkage*) analizi sonrası bulunan markör sayıları Çizelge 4.7 deki gibidir. Şekil 4.3. de görüldüğü gibi Derin Öğrenme ile gerçekleştirilen tahminleme işleminde kromozom üzerindeki markör sıralamaları daha az hata ile gerçekleşmiştir.

Çizelge 4.7. Sentetik Genotip verisi üzerindeki markör sayıları

TAHMİNLEME ÖNCESİ	**TAHMİNLEME (Derin Öğrenme) SONRASI**
34	39

5. SONUÇLAR VE ÖNERİLER

5.1 Sonuçlar

Genotip veri setlerinde kayıp veri tahminleme işlemi için halihazırdaki maksimum performansa sahip yaklaşım Hidden Markov Model yaklaşımıdır. Hidden Markov Model in eldeki veri seti içerisindeki kayıp veri bölgelerini tahmin ederek elde ettiği markörleri kromozom üzerinde doğru lokasyonlara doğru sıralamalarla yerleştirerek genotip analiz çalışmaları için en uygun veri setini elde ettiği gözlemlenmiştir.

Eldeki veri setinde bulunan kayıp veri bölgelerinin fazla olması KNN algoritmasının imputation performansını düşürmüştür. Çünkü fazla kayıp veri bölgesi demek daha fazla komşu değerin kayıp veri bölgesi olması olasılığını artırmaktadır. Bu durumda da Kayıp veri bölgeleri de KNN için bir sınıf teşkil etmektedir. Hem tahminlenemeyen veri bölgeleri açısından hemde tahmin edilen kayıp veri bölgelerinden elde edilen markörlerin kromozom üzerine yerleşimi açısından eldeki veri seti için KNN optimum performans gösterememiştir. Tahmin edilen SNP oranı %76.25 olmasına rağmen, bağlantı (*Linkage*) analizi gerçekleştirildiğinde kromozom üzerindeki tahmin edilen SNP lerin yer ve sırasının da yanlış olduğu saptanmıştır.

Bu çalışma bünyesinde geliştirilen Derin Öğrenme senaryosu ile eldeki sentetik veri seti üzerinde tahmin edilen kayıp veri bölgelerinden elde edilen markörleri kromozom üzerinde KNN e göre daha doğru sıralama ile yerleştirdiği gözlemlenmiştir. KNN daha fazla markör tahmin etmiş olsada kromozom üzerindeki markör sıralamasında hatalar bulunması sebebiyle genel performans olarak Derin Öğrenme senaryosunun altında kalmıştır.

Hidden Markov Model DNA dizilim verilerinde göstermiş olduğu performans dikkate alındığında Gen Analiz işlemleri için en uygun analiz modeli olmaktadır. Eldeki veri setinin bi kromozomu için işlem süresi 3 dakika olmaktadır.

Hidden Markov Model, KNN ve Derin Öğrenme yaklaşımlarının veri seti nin kayıp bölgeleri üzerindeki tahmin kapasite oranları Çizelge 5.1 deki gibidir.

Çizelge 5.1. Algoritmaların veri setindeki (1. kromozom) tahmin kapasitesi oranları

Algoritma	Veri Setindeki Kayıp Veri Oranı	Tahmin Oranı
KNN	52 %	76.25 %
Hidden Markov Model	52 %	100 %
Derin Öğrenme	52 %	100 %

Tahmin edilen kayıp veri bölgelerinden bağlantı (Linkage) analizi sonucu elde edilen markörlerin kromozom üzerindeki yerleşimleri baz alınarak KNN, Hidden Markov Model ve Derin Öğrenme algoritmaları üzerinde yapılan karşılaştırma Çizelge 5.2 deki gibidir.

Çizelge 5.2. Bağlantı (linkage) analizi sonucu kromozom yerleşimine göre algoritmaların performansları

Algoritma	Yanlış Yerleşen Markör Sayısı	Doğru Yerleşen Markör Sayısı
KNN	11	33
Hidden Markov Model	57	57
Derin Öğrenme	11	29

Algoritmaların tahminleme işlemleri sonucunda elde edilen veri setlerinde bağlantı (linkage) analizi sonucu bulunan markör sayılarına göre karşılaştırmaları Çizelge 5.3 deki gibidir.

Çizelge 5.3. Bağlantı (linkage) analizi ile bulunan markör sayılarına göre algoritmaların performansları

Algoritma	Taminleme Sonrası Markör Sayısı
KNN	44
Hidden Markov Model	57
Derin Öğrenme	39

5.2 Öneriler

Bu çalışmada Markov Zincirleri modeli ve yapay öğrenme algoritmaları belirtilen veri setleri üzerinde uygulanmış ve sonuçlar değerlendirilmiştir. Farklı yapay öğrenme algoritmalarının veri seti üzerinde gerçeklenmesi veri tahmin başarısı üzerinde olumlu etkileri olabilecektir. Derin öğrenme algoritmaları için sinir ağında yapılacak optimizasyonlar öğrenme oranını daha yukarılara taşıyacaktır. Sonraki çalışmalarda Derin öğrenme süreci öncesi gerçekleştirilecek bir bağlantı analizi ile giriş nöronlarına uygulanan SNP karakter sayısı üzerinde yapılan düzenlemelerde kayıp veri tahmin başarısını olumlu yönde etkileyecektir. Bu yöntem sadece kayıp veri tahmini için değil farklı genetik analizler içinde kullanılabilecektir. Genotip haritalarını öğrenen yapay sinir ağları ile geliştirilen uygulamalar Dünyanın her yerinden araştırmacıların hizmetine sunularak genetik analizler daha hızlı yapılabileceği gibi hastalık tanılamada da daha yüksek performans kaydedilebilecektir. Sonraki çalışmalarda daha optimize yapay sinir ağı modelleriyle daha performanslı tahminleme ve genotip öğrenme işlemlerinin gerçekleştirilmesi hedeflenmektedir.

KAYNAKLAR

Awais, K., 2016, Introduction to different measures of linkage disequilibrium (LD) and their calculation, University of Illinois, Urbana-Champaign.

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., Bengio, Y., 2010, Theano: a CPU and GPU math expression compiler, In Proceedings of the Python for Scientific Computing Conference (SciPy)-June 2010.

Chollet, F., 2015, Keras, *https://github.com/fchollet/keras* [Ziyaret Tarihi: 20 Ağustos 2018]

Clancy, S., 2008, Genetic Recombination, *https://www.nature.com/scitable/topicpage/genetic-recombination-514*, [Ziyaret Tarihi: 20 Nisan 2017].

Berry, D.P., Kearney, J.F., 2011, Imputation of genotypes from low- to high-density genotyping platforms and implications for genomic selection.

Bradbury, P.J., Zhang, Z., Kroon, D. E., Casstevens, T.M., Ramdoss, Y., Buckler, E. S., 2007, TASSEL: Software for association mapping of complex traits in diverse samples. Bioinformatics 23:2633-2635.

Browning, S. R., 2008, Missing data imputation and haplotype phase inference for genome-wide association studies, National Institutes of Health.

Browning, B. L., Zhou, Y., Browning, S. R., 2018, A one-penny imputed genome from next generation reference panels. Am J Hum Genet 103(3):338-348.

Büyükyılmaz, M., 2017, Mikroskobik Görüntüler Üzerinde Derin Öğrenme Algoritmaları Kullanılarak Hastalıklı Hücrelerin Otomatik Tanımlanması, Necmettin Erbakan Üniversitesi Endüstri Mühendisliği Ana Bilim Dalı, Yüksek Lisans Tezi.

Devlin B., Risch N.,1995, A Comparison of Linkage Disequilibrium Measures for Fine-Scale Mapping, Genomics, 29(2): 311-322

Duda, R. O., Hart, P. E., Stork, D. G., 2012, Pattern classification. John Wiley & Sons.

Ellinghaus, D., Schreiber, S., Franke, A., Nothnagel, M., 2009, Current Software for Genotype Imputation, Henry Stewart publications 1479 –7364. Human Genomics. Vol. 3. No 4. 371–380.

Elmas, Ç., 2011, Yapay Zeka Uygulamaları, Ankara.

Greff K., Srivastava R. K., Koutnik J., Steunebrink B. R., Schmidhuber J., 2015, LSTM: A Search Space Odyssey, IEEE Transaction on Neural Networks and Learning Systems, (Volume: 28, Issue: 10, Oct. 2017) Pages: 2222 - 2232

Haykin, S., 2009, Neural Networks and Learning Machines, Hamilton, Ontario, Canada.

Hinton, G. E., Nair, V.,2010, Proceeding ICML'10 Proceedings of the 27th International Conference on Machine Learning, Pages 807-814

Hochreiter S., Schmidhuber J., 1997, Long Short-Term Memory, Neural Computation, 9(8): 1735-1780

Howie, B. N., Donnelly, P., Marchini, J., 2009, A flexible and accurate genotype imputation method for the next generation of genome-wide association studies. PLoS Genetics 5(6): e1000529PLoS Genetics 5(6): e1000529.

Kandemir, N., 2010, Genetik, İstanbul.

Krizhevsky, A., Sutskever, I., Hinton, G. E., 2012, ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Proceeding Systems 25 (NIPS2012), pages 1-9.

Lewontin R.C. 1988. On Measures of Gametic Disequilibrium, Genetics, 120(3): 849-852

Li, Y., Willer C., Sanna S., Abecasis G., 2009, Genotype Imputation, National Institutes of Health

Money, D., Gardner K., Schwaninger H., Zhong G., Myles S., 2015, LinkImpute: fast and accurate genotype imputation for non-model organisms, NCBI

Olah C., 2015, Understanding LSTM Networks, [Ziyaret Tarihi: 10 Ağustos 2018].

Pausch, H., Aigner, B., Emmerling R., Edel C., Götz, KU., Fries R., 2013, Imputation of high-density genotypes in the Fleckvieh cattle population, Genetic Selection Evolution.

Rosenblatt, F. 1962. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms, New York: Spartan.

Stam, P., 1993. Construction of integrated genetic linkage maps by means of a new computer package: JoinMap. The Plant Journal 3: 739-744.

Tataru, P., Sand, A., Hobolth, A., Mailund, T., Pedersen, C.N.S., 2013, Algorithms for Hidden Markov Models Restricted to Occurrences of Regular Expressions, Biology 2013, 2(4), 1282-1295.

Van Der Walt, S., Colbert, S. C., Varoquaux, G., 2011, The NumPy array: a structure for efficient numerical computation, Computing in Science & Engineering 13.2, p.22-30

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı	**:**	Mithat Raşit ÖZÇIKRIKCI
Uyruğu	**:**	T.C.
Doğum Yeri ve Tarihi	**:**	Konya/01.02.1988
Telefon	**:**	(505)9370451
Faks	**:**	
e-mail	**:**	mirasit.oz@gmail.com

EĞİTİM

Derece		**Adı, İlçe, İl**	**Bitirme Yılı**
Lise	:	Meram Konya Lisesi, Meram, Konya	2005
Üniversite	:	Dumlupınar Üniversitesi, Bilgisayar Mühendisliği, Kütahya	2013
Yüksek Lisans	:		
Doktora	:		

İŞ DENEYİMLERİ

Yıl	**Kurum**	**Görevi**
2014	Necmettin Erbakan Üniversitesi, Bilgi İşlem Daire Başkanlığı	Yazılım Uzmanı

UZMANLIK ALANI

- **Programlama Dilleri:** C, C++, Python, Php, Javascript
- **İşletim Sistemleri:** Linux
- **Framework:** Django, Laravel, QT, Angularjs, Jquery, Bootstrap,

YABANCI DİLLER

- **İngilizce** (Okuma: Orta, Konuşma: Orta, Yazma: Orta)

BELİRTMEK İSTEDİĞİNİZ DİĞER ÖZELLİKLER

YAYINLAR

- **Mithat Raşit Özçıkrıkcı**, Ali Osman Çıbıkdiken, "A Research Of Missing Genotype Data Estimation And Its Important On Genetic Analysis Of Diseases", VI. International KOP Regional Development Symposium