**Yapay Sinir Ağları**

**3.Ödev**

**03.01.2021**

**Mehmet Şerbetçioğlu -- 040160056**

**Ahmet Hulusi Tarhan -- 040170738**

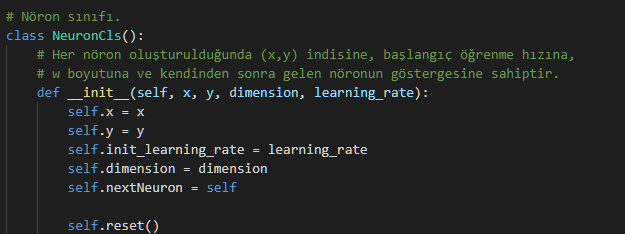
İlk problemde 200 noktadan oluşan üç boyutlu üç farklı noktalar kümesi oluşturup Kohonen ağıyla eğitmemiz isteniyor. Kohonen ağı yüksek boyutlu verilerin bilgisini düşük boyutlu bir uzay üzerinde ifade eder. Verilerin yoğun ve seyrek olduğu noktaları göstermeye yarar.

Eğitim sırasında veri, ağdaki tüm nöronlarla karşılaştırılır. Ağırlığı veriye en yakın olan nöron kazanan nöron seçilir ve ağdaki tüm nöronlar, kazanan nörona olan uzaklıkları doğrultusunda güncellenir. Bu işlem her veri için yapılır ve her veride kazanan nöron ve komşuları veriye yaklaşır.

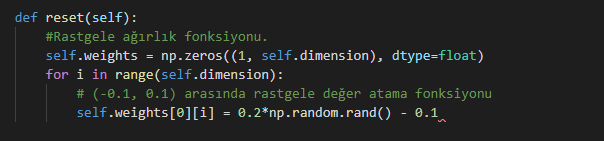
Eğitimin iki farklı aşaması vardır. Bunlar özdüzenleme ve yakınsama aşamalarıdır. Özdüzenleme aşaması ağırlıkların verileri kabaca ifade etmesini sağlar. Her iterasyonda komşuluğun etkisi ve öğrenme hızı düşmekte, ağırlıklar verilere daha dar bir komşulukta ve daha az bir miktarda yaklaşmaktadır. Yakınsama aşamasında ise verilerin daha isabetli bir ifadesi elde edilmeye çalışılır. Bu aşamada komşuluk etkisi ve öğrenme hızı düşük bir seviyededir ve sabittir.

Haykin “Neural Networks – A Comprehensive Foundation” kitabında, bu iki eğitimin sürmesi gereken toplam iterasyon sayısı için, özdüzenlemeye 1000, yakınsamaya ise nöron sayısının 500 katı olması gerektiği tavsiyesinde bulunmuştur. Ayrıca adaptif öğrenme hızı ve komşuluk etkisi için kullanılacak zaman sabitleri ve başlangıç öğrenme hızı için de tavsiyelerde bulunmuştur. Bu tavsiyeler, sebepleriyle birlikte kitabın 474-475. sayfalarında açıklayıcı bir şekilde anlatılmıştır.

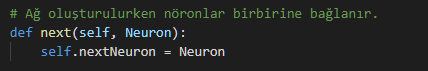
**Nöron Sınıfı:**

Ağı oluşturan nöronların olduğu sınıf. İlk tanıtıldığında (x,y) indisleri, başlangıçtaki öğrenme hızı, ağırlık vektörü için kullanılacak olan veri boyutu ve şimdilik kendisini gösteren sonraki nöron ifadesi bulunmaktadır. Ayrıca yeni nöron oluştuğunda ağırlıkların atanması için bir reset() fonksiyonu bulunmaktadır. 

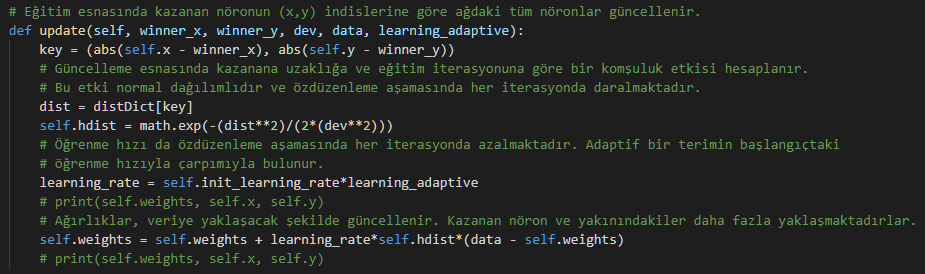
Reset fonksiyonu içerisinde nöron için boş bir ağırlık dizisi açılır. Ardından bu dizi (-0.1,0.1) aralığında “uniform” dağılımlı rastgele değerlerle doldurulur. Bu değer aralığı Haykin’in daha önce bahsedilen kitabındaki değerler sebebiyle seçilmiştir.

****

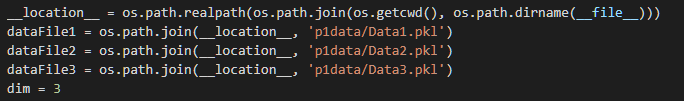
Nöron sınıfında ayrıca sonraki nöronu atayan bir fonksiyon bulunmaktadır. Bu fonksiyon nöronları birbirine bağlayarak ağın çiziminde kullanılacaktır.



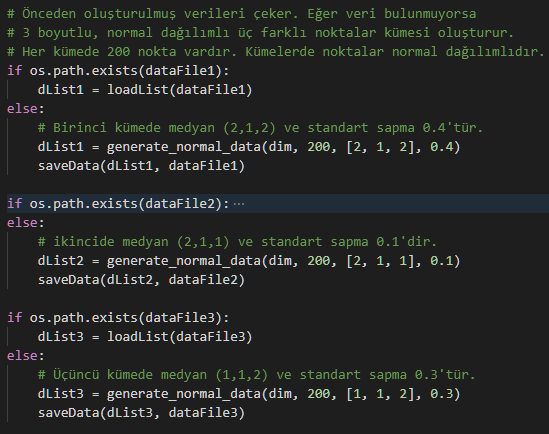
Son olarak ağırlıkların güncellendiği bir fonksiyon bulunmaktadır. Ağırlıklar güncellenirken komşuluk etkisi, adaptif öğrenme hızı, veri ve güncel ağırlıklar kullanılacaktır. Komşuluk etkisi için kazanan nöronun (x,y) indisleri kullanılır. Güncellenecek nöronun kazanan nörona uzaklığı için (x,y) farkına göre uzaklıkların bulunduğu bir sözlükten faydalanılır. İterasyon bilgisi içeren adaptif öğrenme hızı ve komşuluk etkisi standart sapması da bu fonksiyonun dışında hesaplanıp fonksiyona girilir.

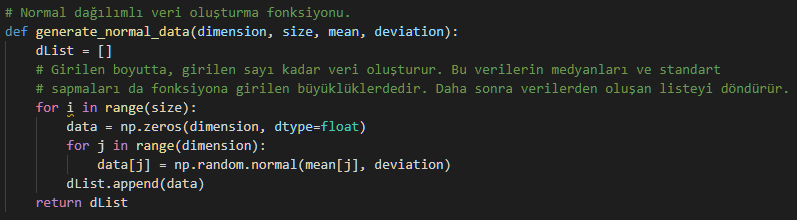


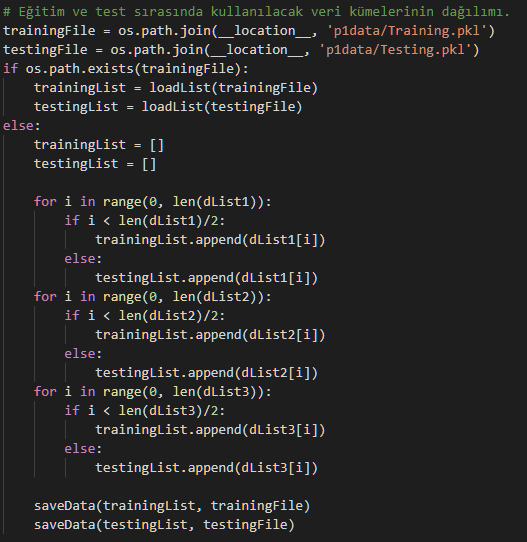
**Verilerin oluşumu:**

Veriler “p1data” klasöründe kaydedilip tekrar kullanılacaktır. Eğer önceden oluşturulmuşsa, veriler bu klasörden çekilir. Aksi takdirde yeni veriler oluşturulur. 

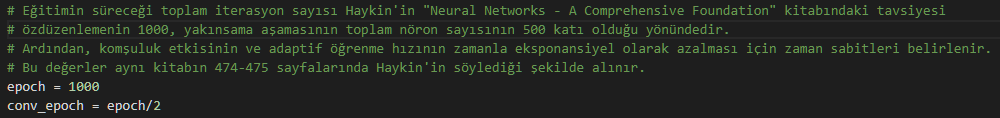
Oluşturulan veriler üç boyutlu olup medyan ve standart sapmaları sırasıyla (2,1,2) – 0.4, (2,1,1) – 0.1, (1,1,2) – 0.3 seçilmiştir.



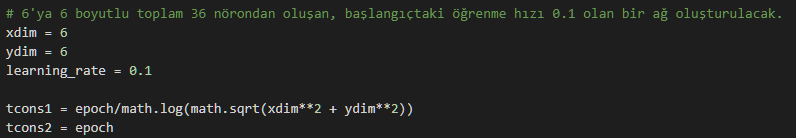
Veriler “generate\_normal\_data” fonksiyonu ile oluşturulmaktadır. Bu fonksiyon, girilen medyan etrafında, girilen standart sapmada normal dağılımlı, girilen boyutta ve girilen sayıda veri oluşturup bunları bir listede toplar. 

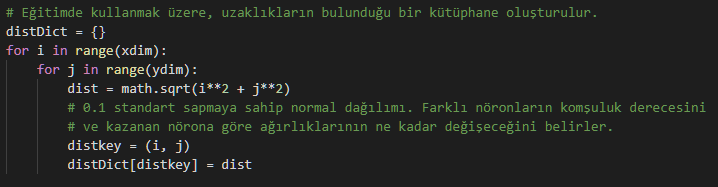
Bunun ardından daha önce oluşturulan eğitim ve test kümeleri yüklenir. Eğer oluşturulmamışsa, üç veri kümesi yarı yarıya eğitim ve test kümelerine dağıtılır. 

**Eğitim hazırlıkları:**

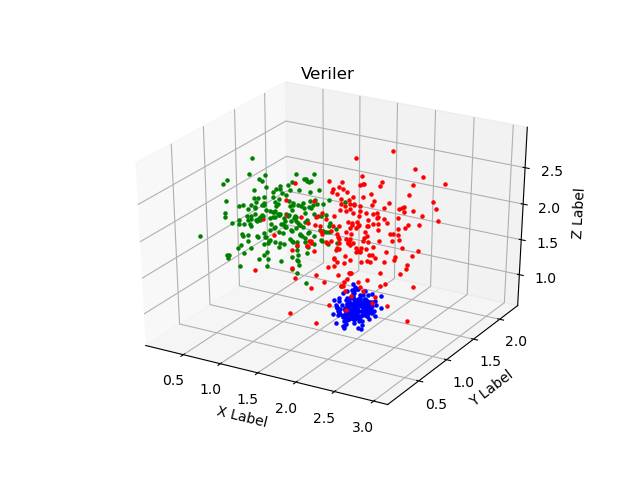
Eğitimde özdüzenleme ve yakınsama aşamalarının süreceği iterasyon sayıları seçilir. Haykin özdüzenleme için 1000, yakınsama için nöron sayısının 500 katı olmasını tavsiye ettiği için problemin devamı boyunca yakınsama iterasyonu, özdüzenleme iterasyonunun yarısının nöron sayısıyla çarpımı seçilecektir. 

Ağ boyutu 6’ya 6, başlangıç öğrenme hızı 0.1 seçilir. Adaptif öğrenme hızı ve komşuluk etkisi için kullanılacak olan zaman sabitleri de Haykin’in tavsiye ettiği gibi seçilir.

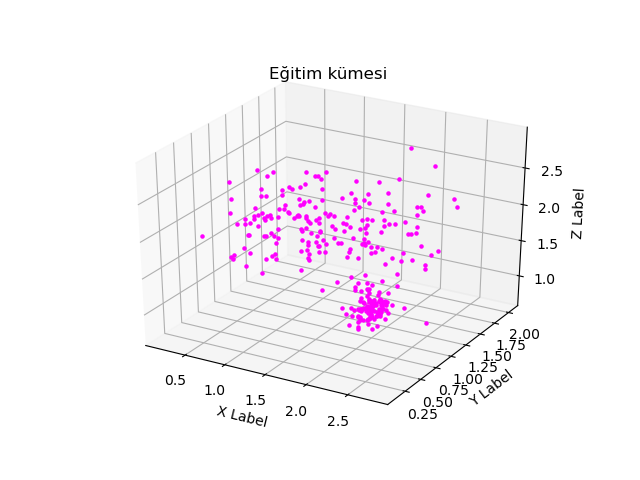


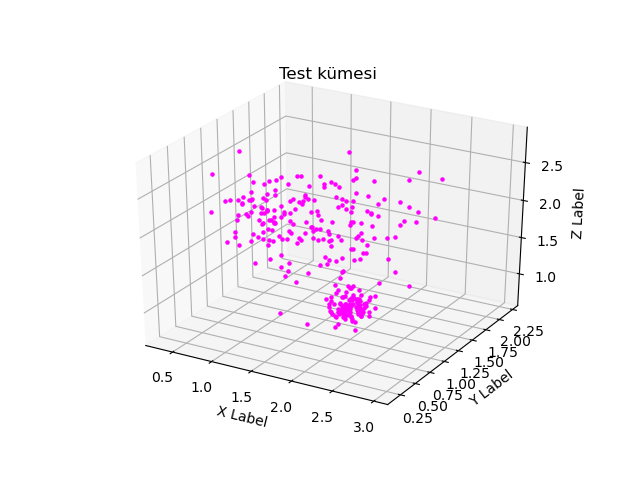
Ağ boyutu kullanılarak nöronlar arasındaki uzaklıkların bulunduğu bir sözlük hazırlanır. Bu sayede her seferinde uzaklık hesaplanmayacak, indisleri arasındaki farklarla sözlükteki değer seçilecektir. 

Oluşturulan veri, test ve eğitim kümeleri şekildeki gibidir.

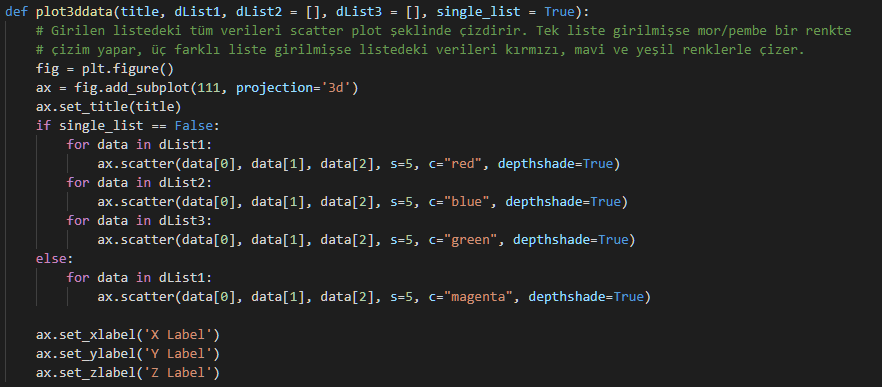


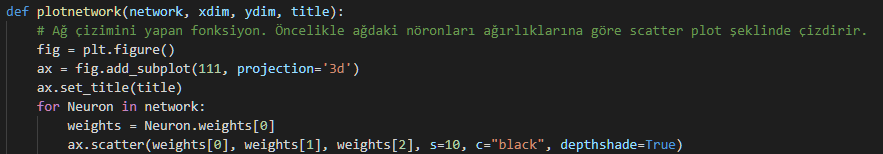
Eğitim ve test kümelerinde sınıf bilgisi olmadığı için tüm veriler aynı renkte çizilmiştir. (epoch = 1000 değerindeki eğitim kümesinin ölçeklenmesinde bir sorun yaşandığı için epoch = 50 değerindeki grafikler kullanılacaktır.)



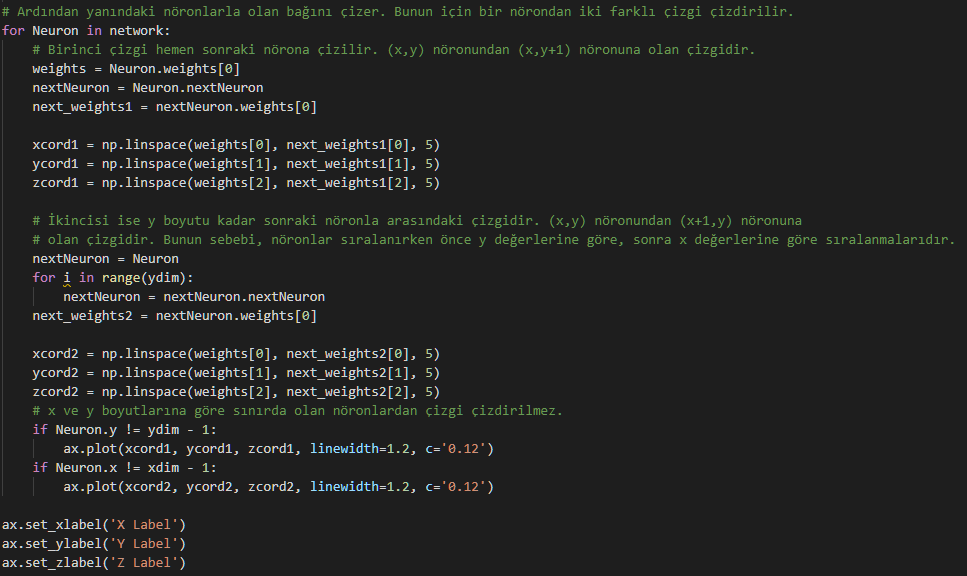


**Çizim için kullanılan fonksiyonlar:**

Sadece veriler çizilirken “plot3ddata” kullanılır. Bu fonksiyonda üç farklı veri kümesi girilip farklı renklerde veya tek veri kümesi girilip mor/pembe renkte çizdirilebilir.****

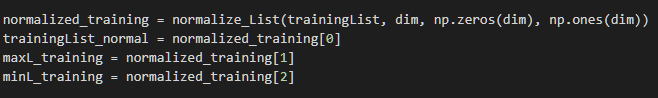
Ağın çizimi biraz daha karmaşıktır. Öncelikle nöronların ağırlıkları nokta halinde çizdirilir. 

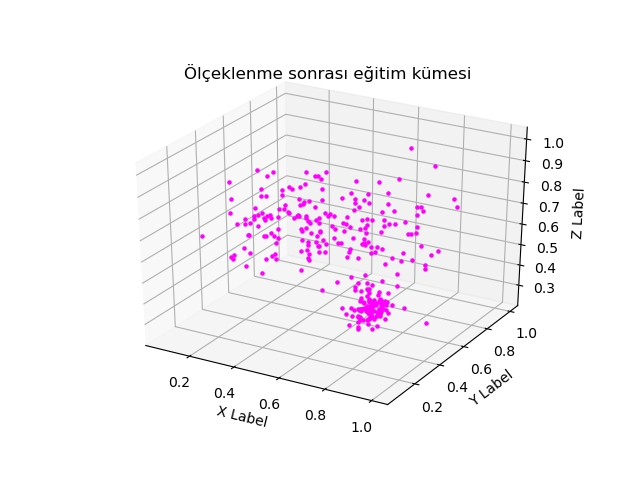
Ardından nöronlar arasındaki bağlar çizdirilir. (0,0) nöronundan (1,0) ve (0,1) nöronuna bağ çizilir. Bu, sınırlardaki nöronlar haricinde tüm nöronlarda uygulandıktan sonra bağlar çizilmiş olur. Bu çizimler yapılırken ağ oluşturulurken nöronlara atanan sonraki nöronlar kullanılır. Ağ oluşturulurken önce y indisindeki nöronlar atanıp, daha sonra x indisi kullanıldığı için (x,y) nöronu ile (x+1,y) nöronu arasındaki bağın çizimi için (x,y) nöronunun y boyutu kadar sonrasındaki nörona ulaşılır. Bu nöron (x+1,y) nöronu olacaktır. (x,y) nöronu ile (x,y+1) nöronu arasındaki bağ için ise nörondan hemen sonraki nörona ulaşılır.

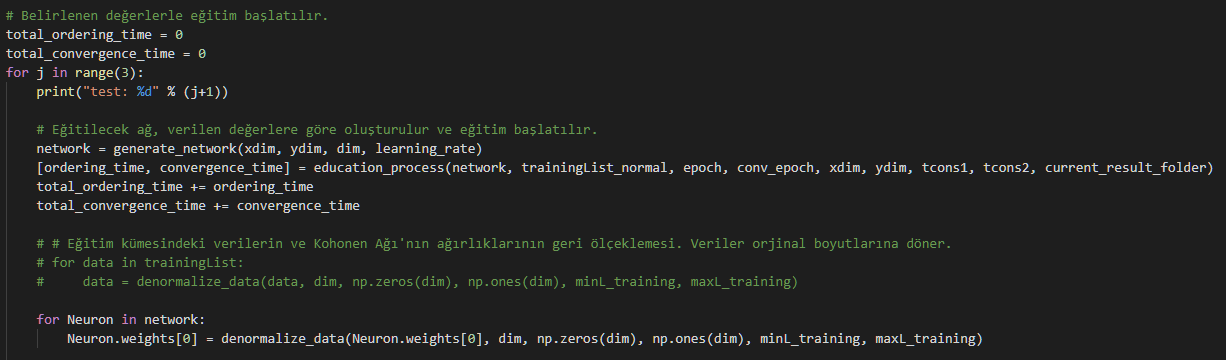


“plotnetworkanddata” fonksiyonu ise iki fonksiyonun birleşimidir. Aynı figür üzerinde çizilmesi için iki fonksiyon ayrı ayrı kullanılamamaktadır. 

**Eğitim hazırlıkları devamı:**

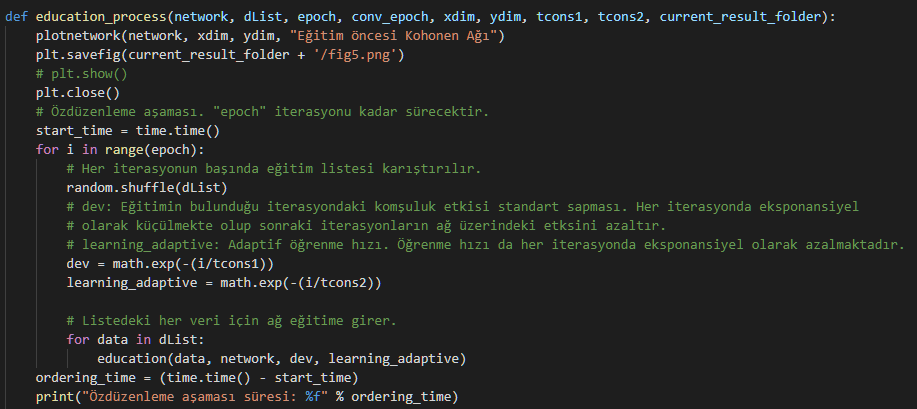
Eğitim kümesi eğitime girmeden önce (0,1) aralığında ölçeklenir. Ölçeklenme öncesindeki minimum ve maksimum değerleri geri ölçekleme için saklanır. 



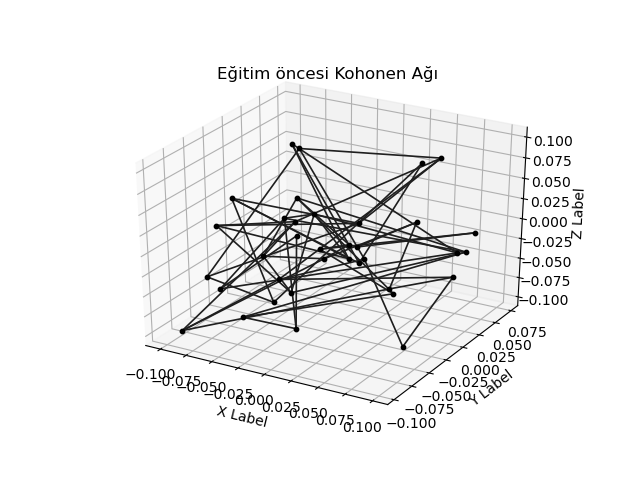
Eğitim üç kere tekrarlanıp özdüzenleme aşaması, yakınsama aşaması ve toplam eğitim süresinin ortalaması alınıp kaydedilecektir. Eğitim sonrasında daha önce elde edilen minimum ve maksimum değerleriyle ağ üzerinde geri ölçekleme yapılır. 

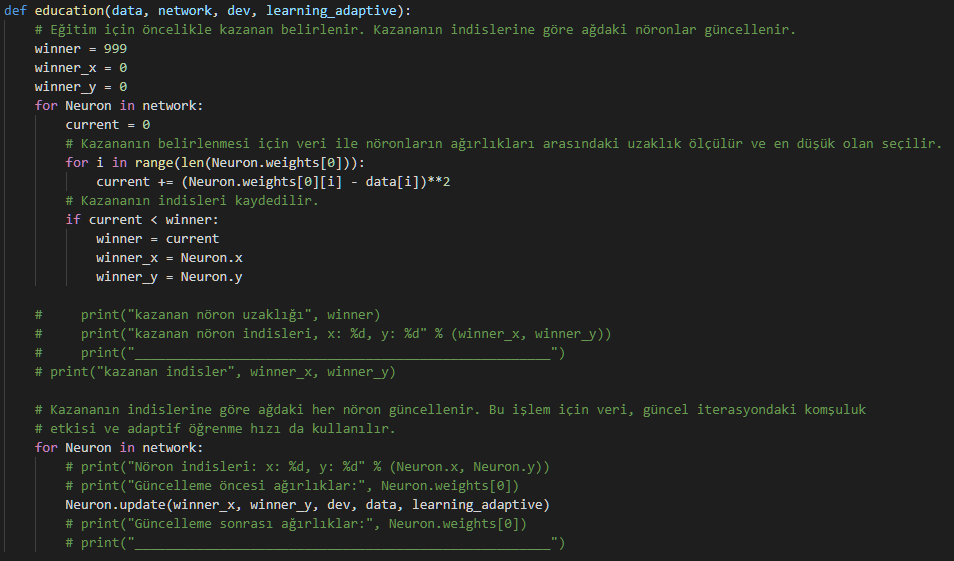
**Eğitim:**

İlk olarak özdüzenleme aşaması gerçekleşir. Her iterasyonun başında eğitim listesi karıştırılır. Her iterasyonda adaptif öğrenme hızı terimi ve komşuluk etkisi standart sapması hesaplanarak eğitimde kullanılır.

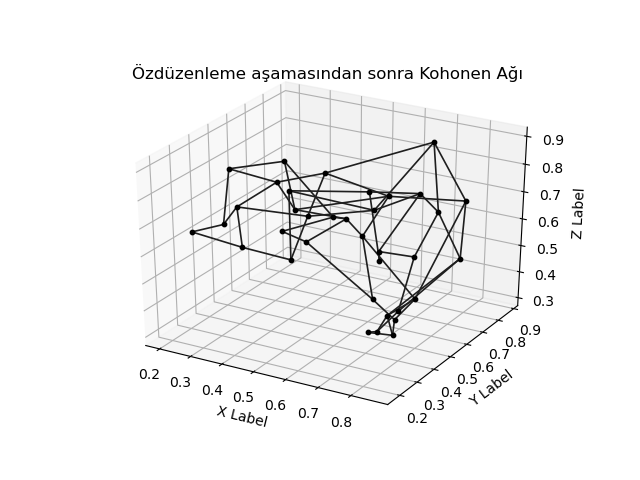


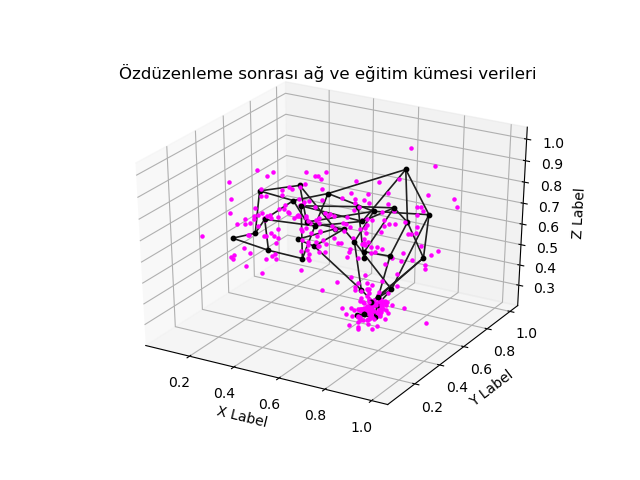
Eğitimden önce ağ iç içe geçmiş bir haldedir. Ağırlıklar rastgele dağıtıldığı için herhangi bir düzen gözlenmez.

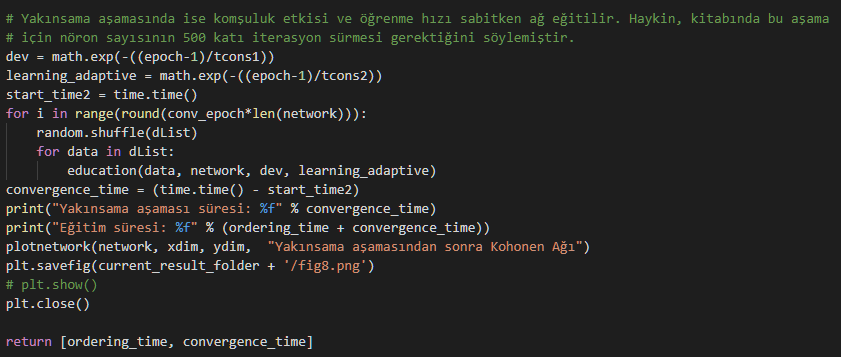


Eğitim sırasında kazanan nöronun belirlenmesi için veri ile nöron ağırlıkları karşılaştırılır. Veriye en yakın nöron kazanan nöron seçilir ve (x,y) indisleri kaydedilir. Ardından ağdaki tüm nöronların ağırlıkları kazanan nöronun indisine göre güncellenir. Kazanan nörona yakın nöronların ağırlıkları, veriye daha çok yaklaşacaktır.

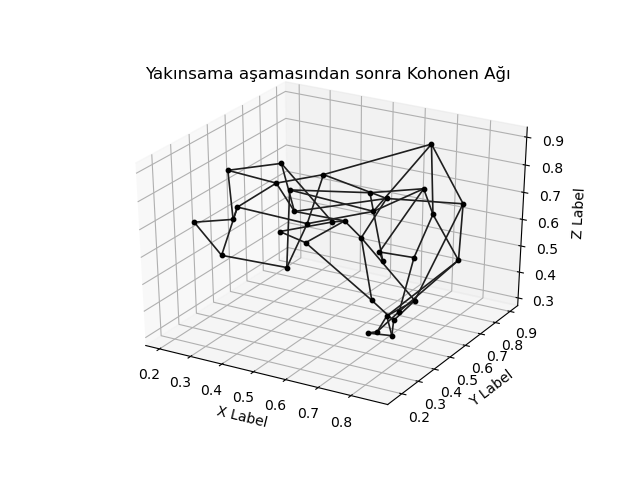
Özdüzenleme sonrasında ağ, Haykin’in örneğindeki kadar düzenli olmasa da gayet düzenli bir yapıdadır. Eğitilen verilerin oluşturduğu öbeklere yaklaştığı açık bir şekilde gözlenebilir. Standart sapması yüksek olan kümelerdeki nöronlar daha seyrekken, standart sapması 0.1 olan kümede gayet yakınlardır. Nöronlar bir şekilde verilerin dağılımını taklit ederler.



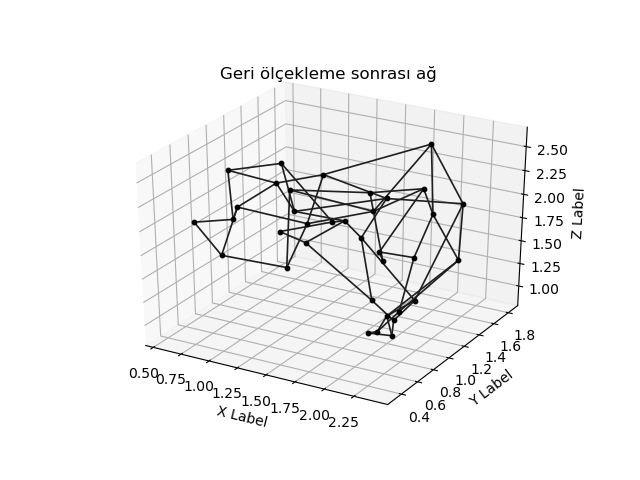


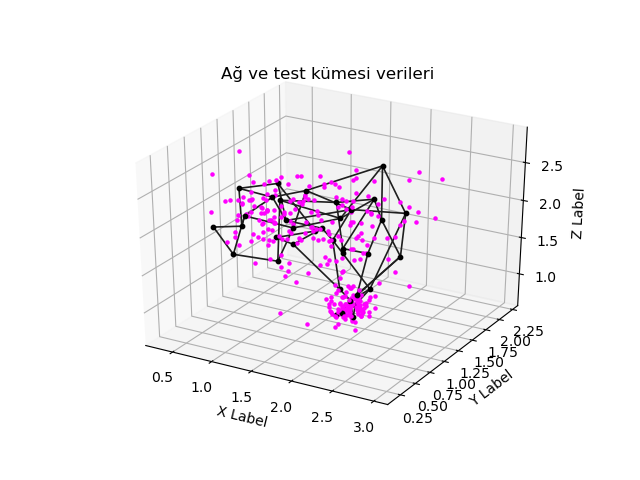
Özdüzenleme aşamasının ardından yakınsama aşaması gerçekleşir. Bu aşamada komşuluk etkisi standart sapması ve adaptif öğrenme hızı terimi sabit kalacaktır. 

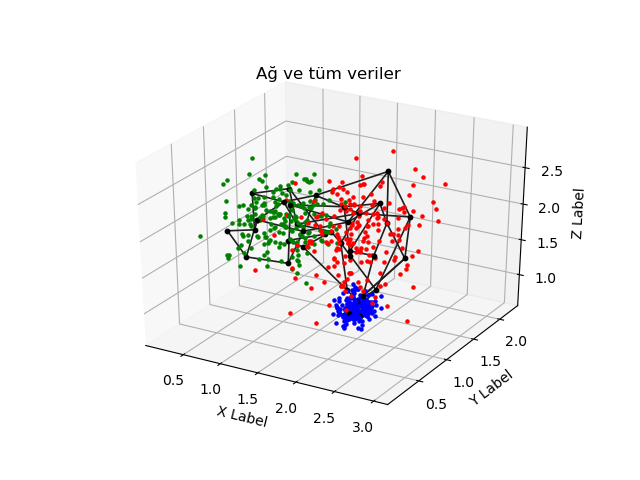
Yakınsama aşaması sonrası ağda daha detaylı değişimler gözlenmektedir.



Ardından ağ geri ölçeklenerek veri boyutlarına döndürülür.



Test kümesiyle karşılaştırıldığında da nöronların öbeklerde yoğunlaştığı görülebilir.,

Tüm verilerle karşılaştırıldığında, ağdaki bazı nöronların veri kümelerinin iç içe olduğu konumlarda bulunduğu görülür. Ağ, öbekleri ifade etse de verilerin iç içe olduğu durumları ayırt edememektedir.

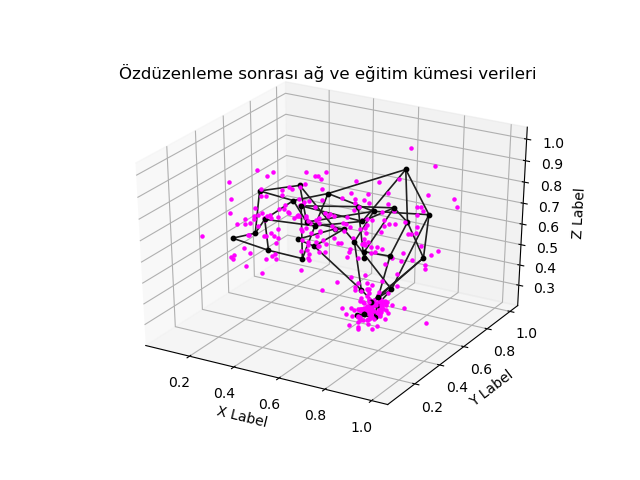
Maalesef, eğitim kümesi listesi ölçeklendirme fonksiyonun bu global eğitim kümesini etkilemiştir. Bu sebeple diğer testlerdeki verilerden elde edilen ağlar geri ölçeklenememiş, test kümesiyle karşılaştırılamaz haldedirler. Fakat eğitim sonucu elde edilen ağ yapılarına bakıldığında ilk testtekine benzer olduğu ve öbekleri başarılı bir şekilde ifade ettikleri görülür. Bu grafiklerin tümü “p1results” klasöründe ilgili test klasörleri altında bulunmaktadır. Yapılan testler;

* özdüzenleme iterasyon sayısının {50, 250, 500, 750, 1000} seçilmesi,
* öğrenme hızının {0.001, 0.01, 0.1, 0.3, 0.5} seçilmesi,
* ağ boyutunun 4’e 4, 5’e 5, 6’ya 6, 7’ye 7, 8’e 8 seçilmesidir.

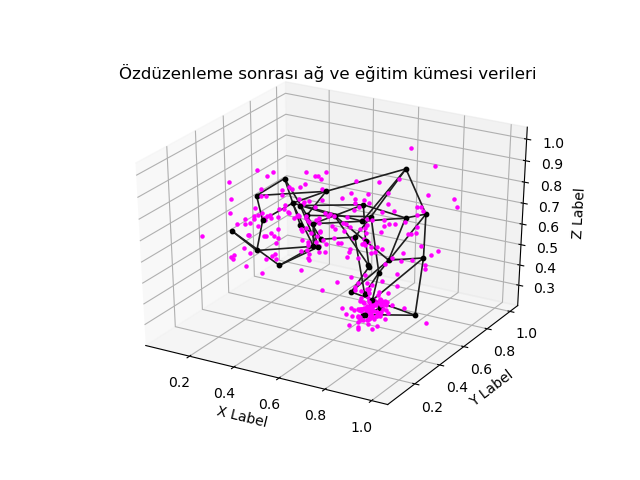
Aksi belirtilmedikçe özdüzenleme iterasyon sayısı 250, başlangıç öğrenme hızı 0.1 ve ağ boyutu 6’ya 6’dır. Test verileriyle olan sonuçlar bozuk olduğu için normalize eğitim kümesiyle ağın karşılaştırmaları yapılacaktır. Bu karşılaştırmalar yine aynı sebepten dolayı özdüzenleme sonrası ağ ile yapılacaktır.

**Özdüzenleme iterasyon sayısı testleri:**

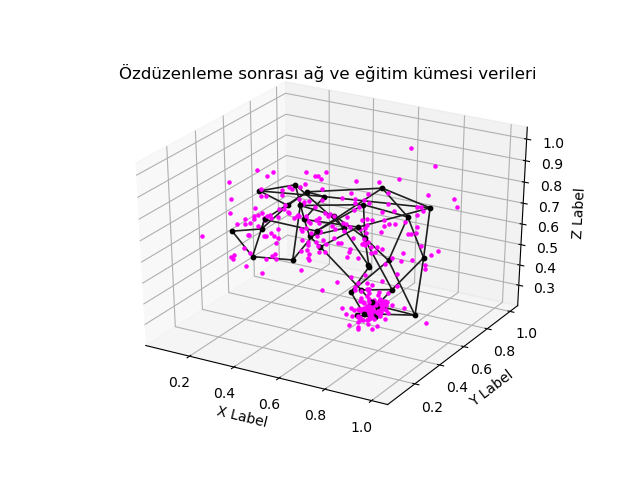
* Epoch = 50



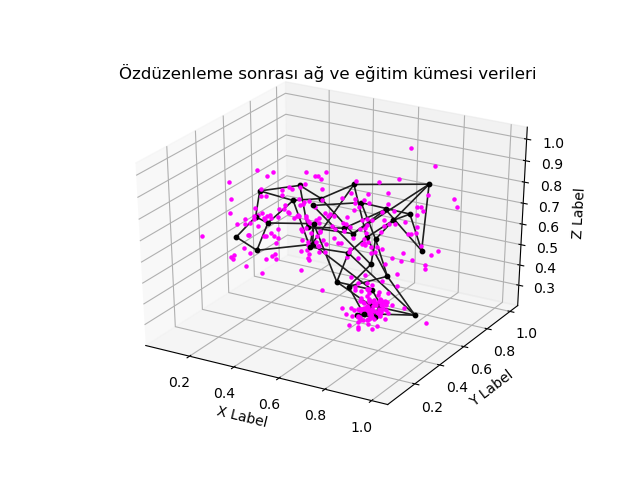
* Epoch = 250



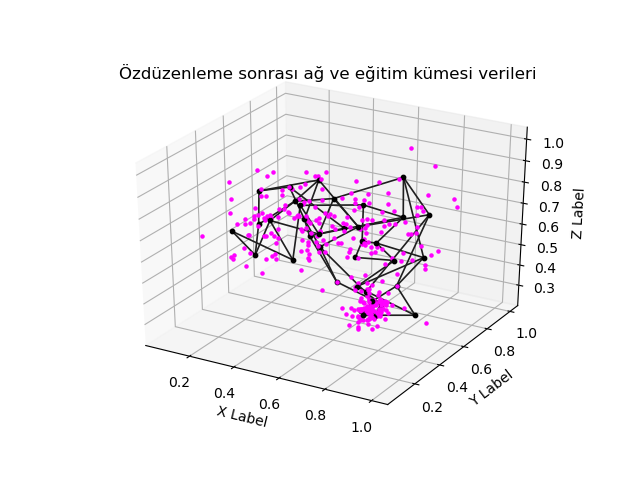
* Epoch = 500:



* Epoch = 750:



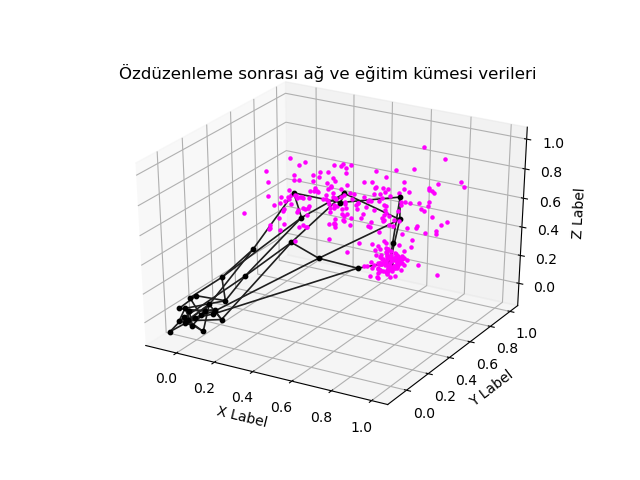
* Epoch = 1000:



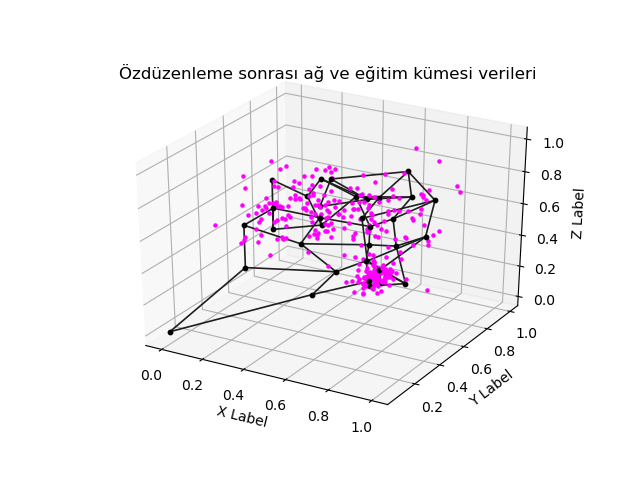
Elde edilen ağlar fazlasıyla çeşit gösterirken her seferinde öbekleri ifade etmeyi başarmıştır. Çeşitliliğin sebebi iterasyon sayısındaki farklılık yerine ağırlıkların başlangıç koşulları olabilir.

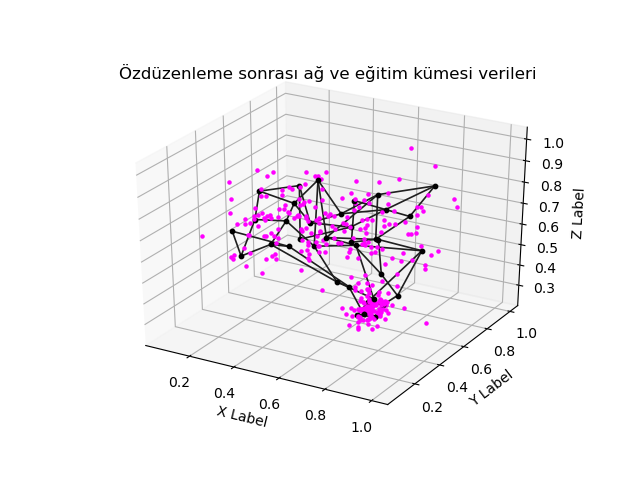
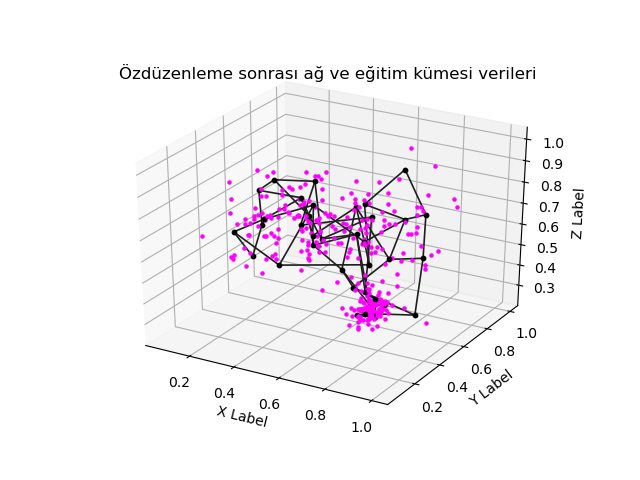
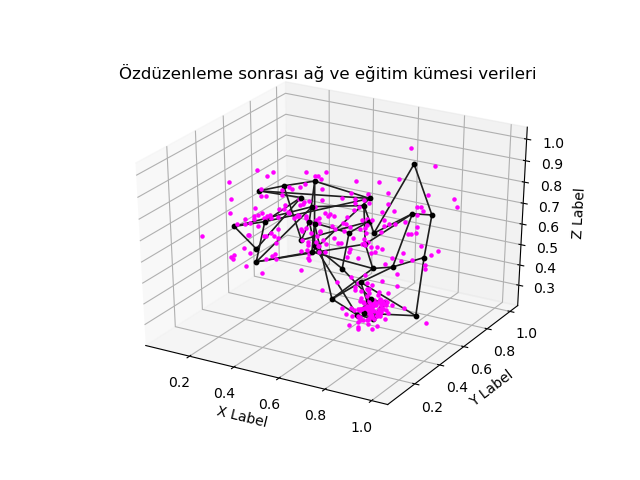
**Başlangıç öğrenme hızı testleri:**

* Öğrenme Hızı = 0.001:



* Öğrenme Hızı = 0.01:

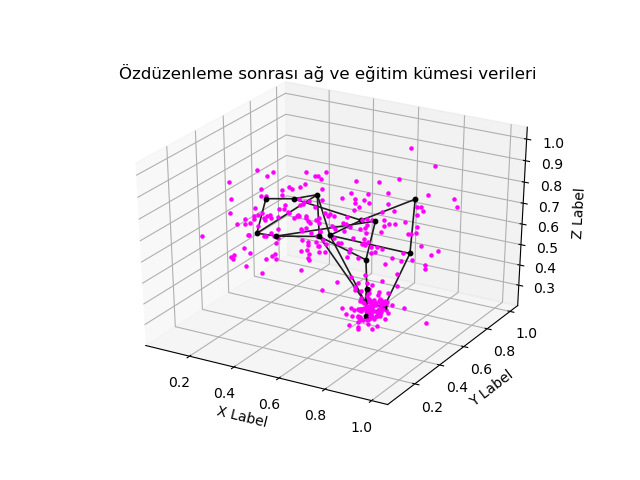
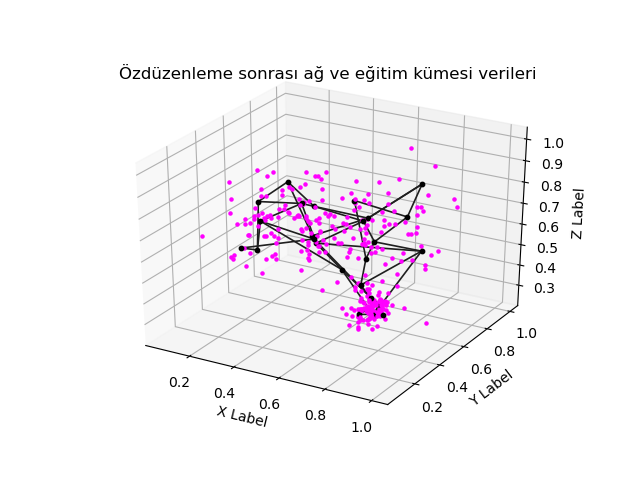
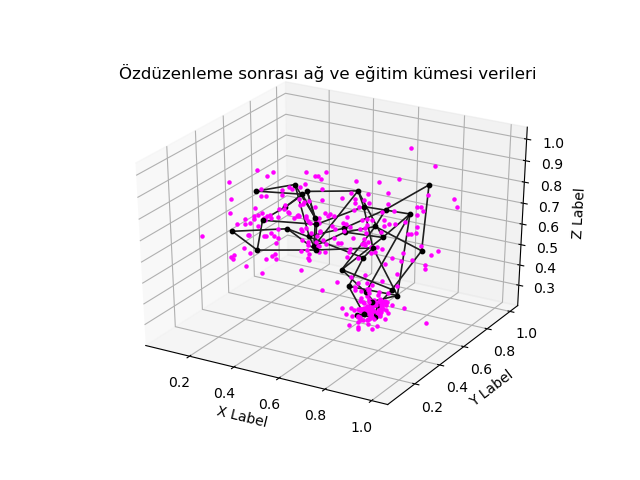
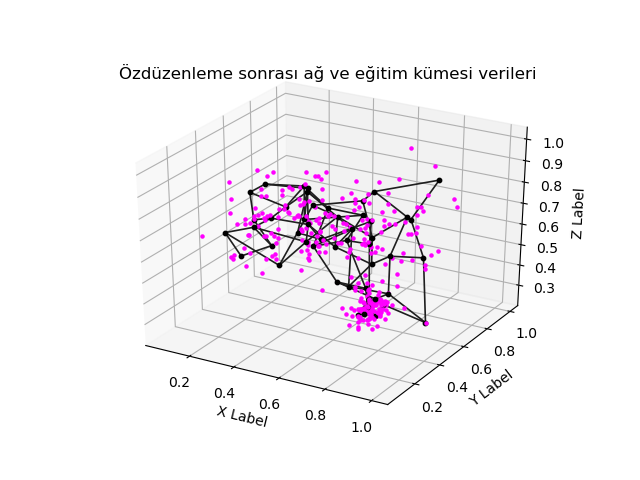
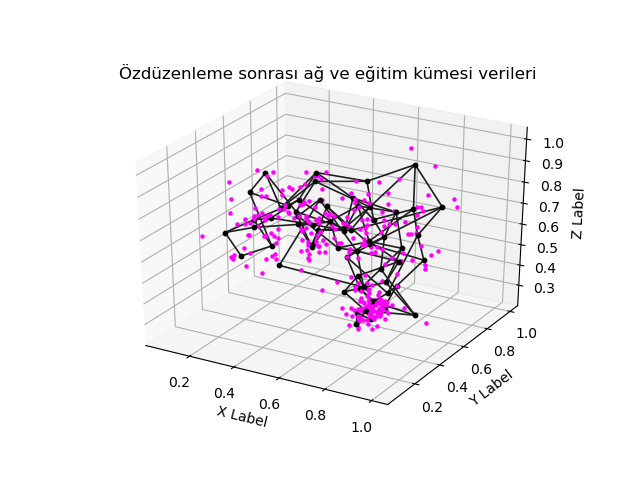


* Öğrenme Hızı = 0.1: 
* Öğrenme Hızı = 0.3: 
* Öğrenme Hızı = 0.5: 

Öğrenme hızı 0.1’in altındaki değerlerde kazanamayan nöronların dışarıda kaldıkları ve kazanmamaya devam ettikleri gözlenir. Bu durum 0.001 öğrenme hızı için çok daha çarpıcı bir şekilde gözlenir. Ağın yarısından azı öbekleri temsil etmektedir. Haykin’in tavsiyesinin başarısı bu testte gözlenebilir.

Öğrenme hızı yüksek olduğunda ise ağ öbekleri temsil edebilirken, 0.1 olduğu durumdaki kadar isabetli bir temsil olmadığı gözlemlenmektedir.

**Ağ boyutu testleri:**

* Ağ boyutu = 4x4:
* 
* Ağ boyutu = 5x5: 
* Ağ boyutu = 6x6: 
* Ağ boyutu = 7x7: 
* Ağ boyutu = 8x8: 

Ağ boyutunun büyümesi öbeklerin daha detaylı bir şekilde temsil edilmesini sağlarken, 4’e 4’lük ağ bile bu konuda başarılı olmaktadır.

**Eğitim süreleri:**

Eğitim süresi için epoch = özdüzenleme iterasyon sayısı ise

özdüzenleme aşamasının epoch ile,

yakınsama aşamasının (nöron sayısı)\*epoch/2 ile,

toplam eğitim süresinin ise iki aşamanın toplamı ile epoch\*(1 + (nöron sayısı)/2) ile

doğru orantılı olması gerekir. Ölçülen eğitim süreleri tablodaki gibidir.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Epoch | 50 | 250 | 500 | 750 | 1000 |
| Özdüzenleme aşaması [sn] | 7,0825 | 35,0963 | 70,9410 | 134,2541 | 160,4658 |
| Yakınsama aşaması [sn] | 127,7791 | 641,3150 | 1333,4954 | 2347,0576 | 2844,2079 |
| Eğitim süresi [sn] | 134,8616 | 676,4113 | 1404,4364 | 2481,3118 | 3004,6737 |
|  | | | | | |
| Başlangıç öğrenme hızı | 0,001 | 0,010 | 0,100 | 0,300 | 0,500 |
| Özdüzenleme aşaması [sn] | 12,4954 | 14,1313 | 13,7148 | 13,8666 | 15,2210 |
| Yakınsama aşaması [sn] | 235,8309 | 261,3103 | 270,7707 | 236,6021 | 244,8789 |
| Eğitim süresi [sn] | 248,3263 | 275,4417 | 284,4855 | 250,4686 | 260,0999 |
|  | | | | | |
| Ağ boyutu | 4x4 | 5x5 | 6x6 | 7x7 | 8x8 |
| Nöron sayısı | 16 | 25 | 36 | 49 | 64 |
| Özdüzenleme aşaması [sn] | 7,7551 | 10,6876 | 13,2846 | 19,6483 | 22,3913 |
| Yakınsama aşaması [sn] | 58,7694 | 182,4266 | 257,8314 | 474,3659 | 678,3452 |
| Eğitim süresi [sn] | 66,5245 | 193,1143 | 271,1160 | 494,0141 | 700,7365 |

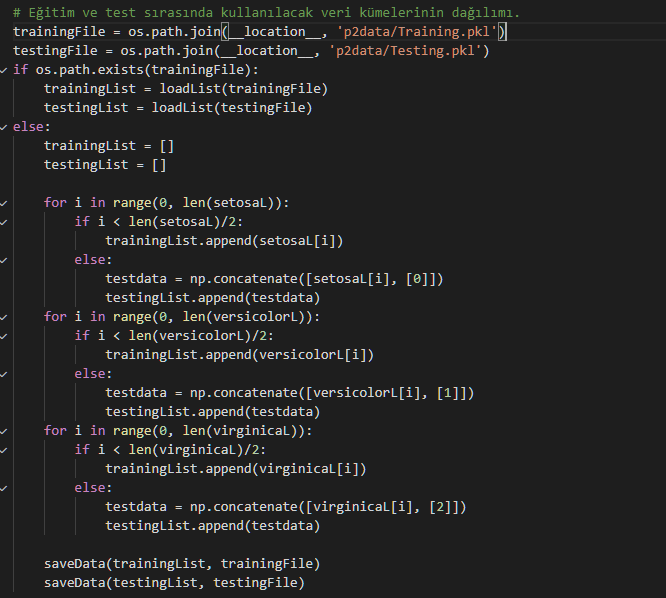
Tabloda, beklenen doğrultuda bir eğitim süresi gerçekleştiği gözlenlemlenebilir. İterasyon sayısı eğitim süresini her 250 iterasyonda yaklaşık 800 saniye artırmış, başlangıç öğrenme hızı eğitim süresini etkilememiş ve ağdaki nöron sayısı, nöron sayısıyla hemen hemen doğru orantılı bir şekilde eğitim süresini artırmıştır.

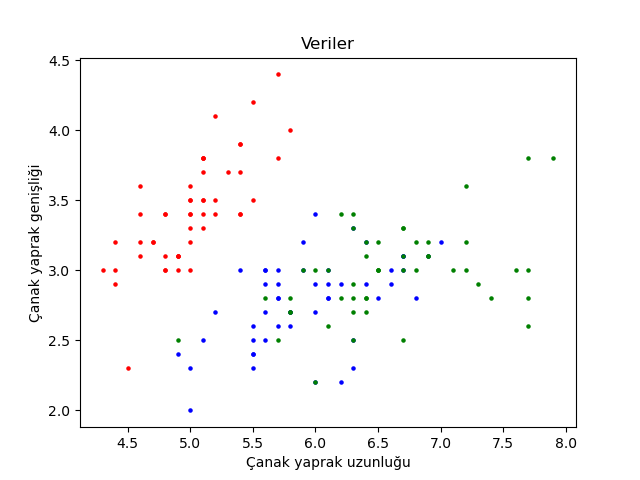
Ayrıca eğitim sırasında birkaç for döngüsü iç içe kullanıldığı ve eğitim her nöron sınıfının içinde yapıldığı için oldukça uzun sürebildiği görülmüştür. Eğitim süresinin 50 dakikaya kadar çıktığı gözlenir.

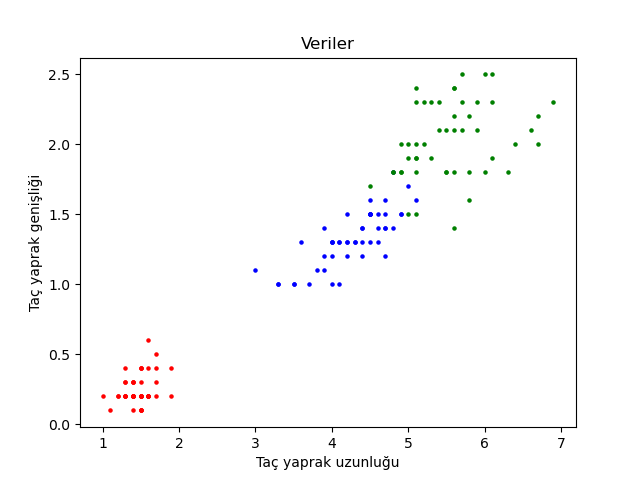
Kurulan Kohonen Ağı ile bir önceki ödevdeki İris verilerinin ayrıştırılması için bir yapı oluşturulur. Çok katmanlı algılayıcı ile iyi ihtimalle %3 civarında hata oranı sağlarken bu ağ yapısı ile nasıl bir performans aldığı gözlemlenecektir.

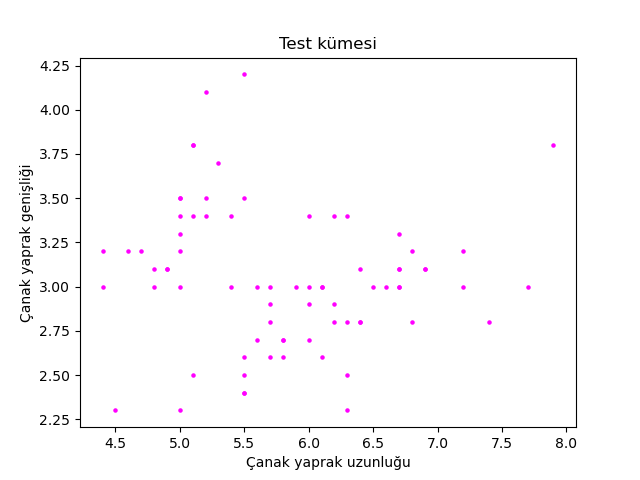
**Verilerin oluşumu:**

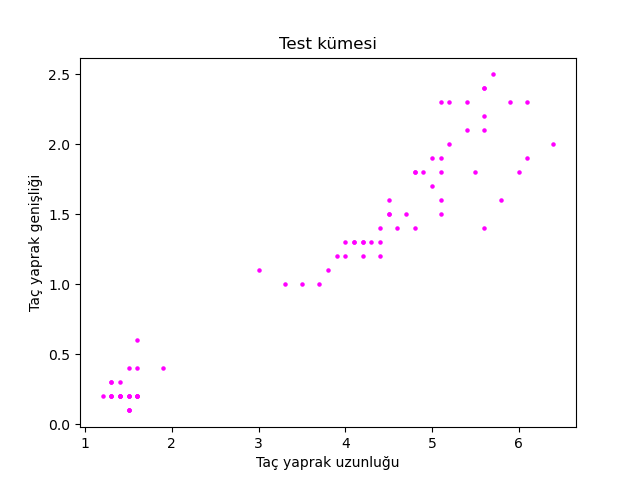
Veriler ilk olarak iris.data dosyasından çekilip setosa, versicolor ve virginica çiçekleri için ayrı oluşturulan listelere yerleştirilir. 

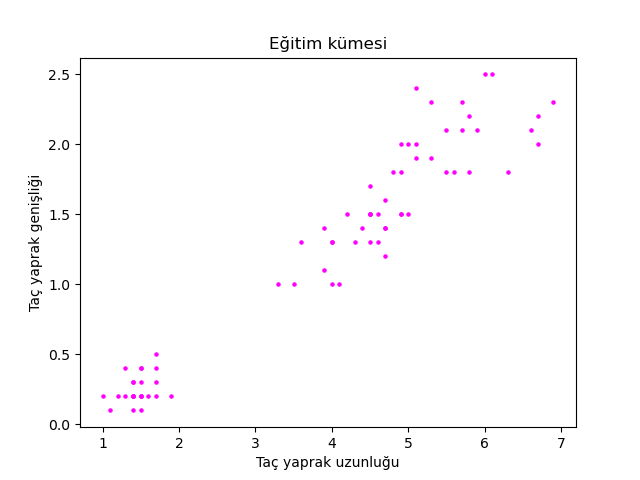
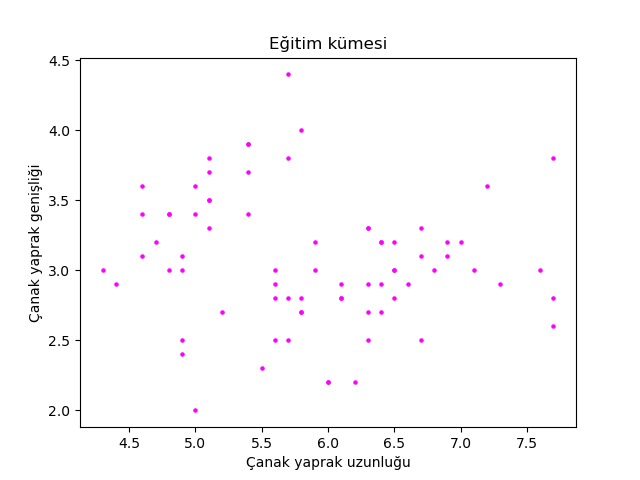
Daha sonra her çiçek kümesinin yarısı test, yarısı eğitim kümesine atanır. Eğitim kümesindeki verilerde hala sınıf bilgisi bulunmuyorken, bu sefer test kümesine bu bilgi verilir. Çiçek sınıfına göre verinin sonuna setosa için [0], versicolor için [1] ve virginica için [2] eklenir. Bu değerler test aşamasında indis olarak kullanılacaktır. 





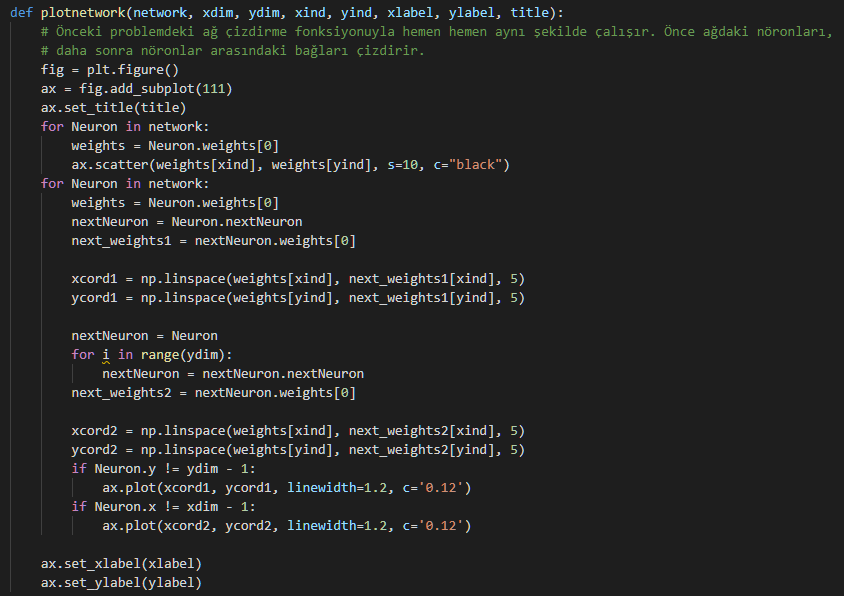
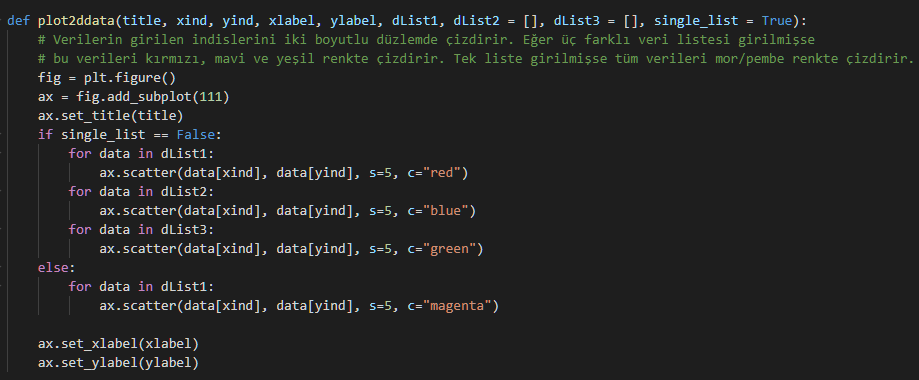




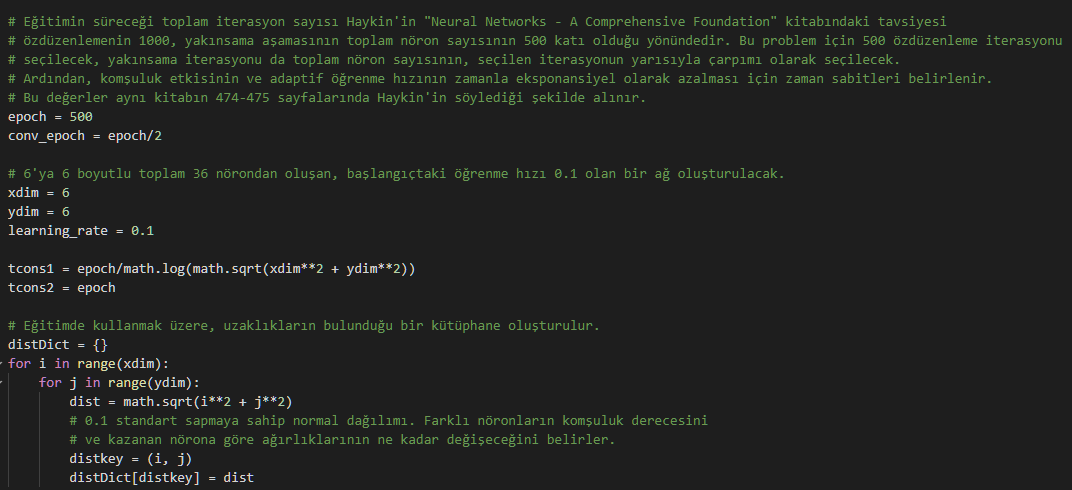


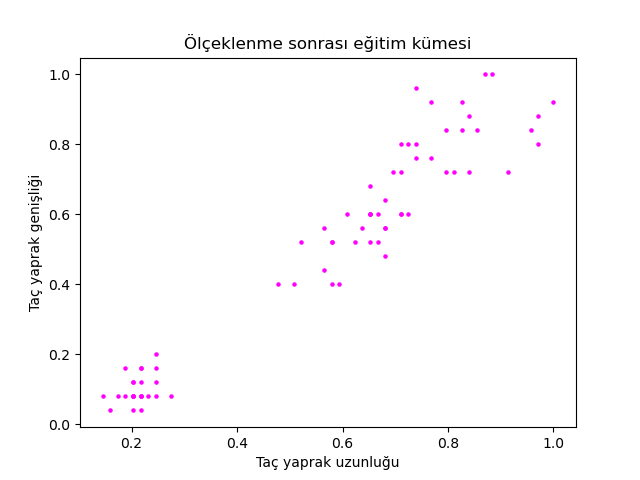
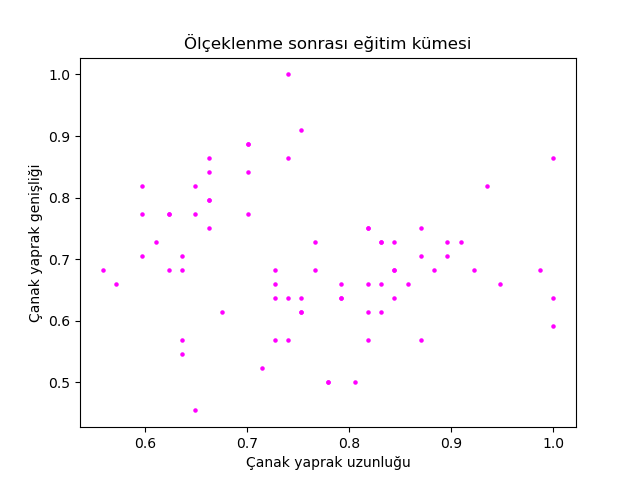
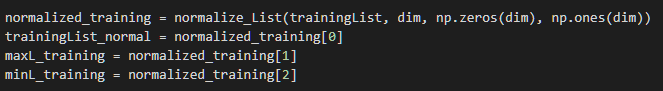
**Verilerin çizimi:**

Verilerin çizimi için önceki problemde kullanılan çizim fonksiyonları iki boyutlu çizim için değiştirilmiştir. Veriler 4 boyutlu olup çanak ve taç yaprakların genişlik ve uzunluklarını bulundurmaktadır. Çanak yaprak verileri ve taç yaprak verileri ayrı grafiklerle ifade edilir.

**Eğitim hazırlıkları:**

Önceki problemle birebir aynıdır. Epoch = 500, ağ boyutları 6x6 ve başlangıç öğrenme hızı 0.1 seçilir. Zaman sabitleri hesaplanır ve uzaklık sözlüğü oluşturulur. 

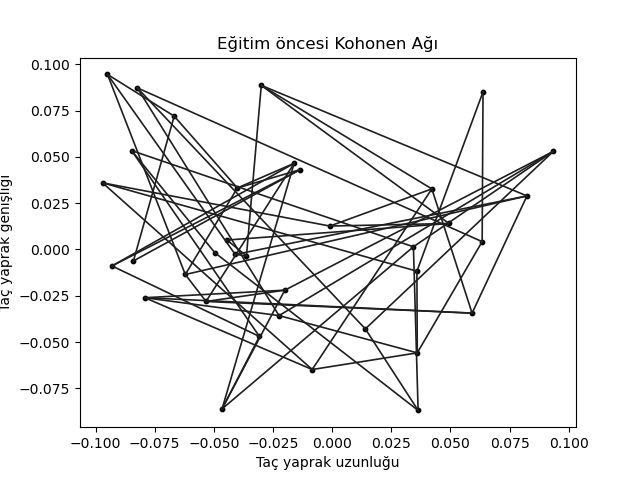
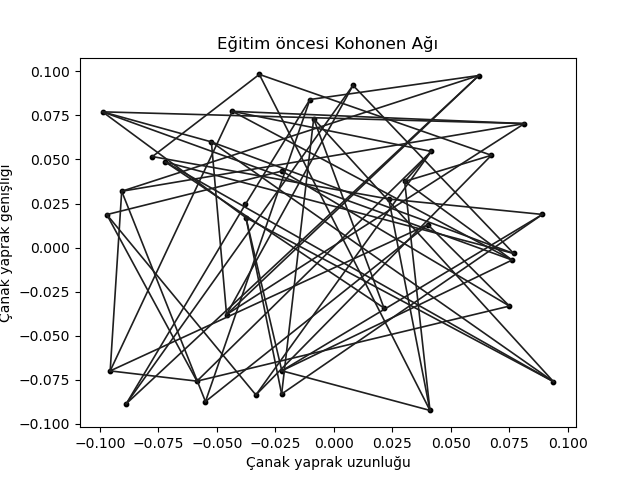
Eğitim kümesi (0,1) arasında ölçeklenir. 

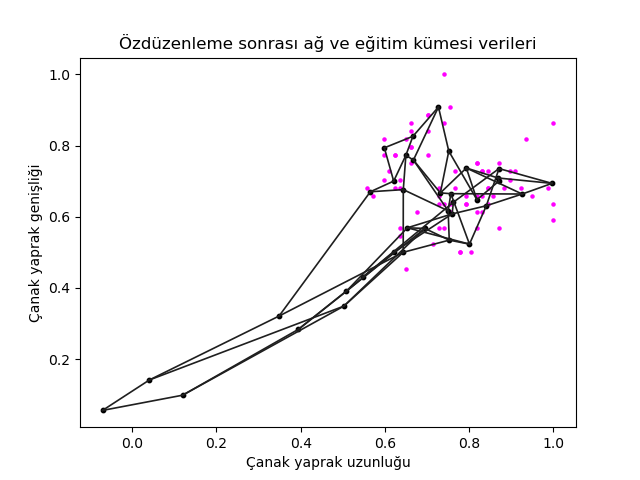
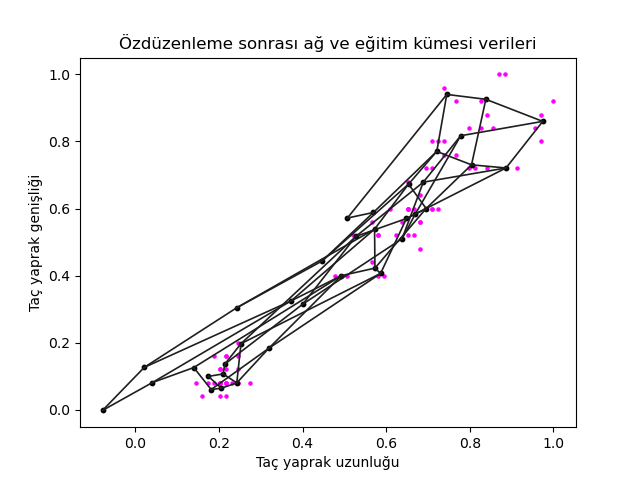
Eğitim için ağ oluşturulur ve eğitim başlatılır. Eğitim aşaması önceki problemle tamamen aynıdır, sadece veri boyutu ve ağın eğitildiği veriler farklıdır. Başlangıçtaki ağ yapısı beklendiği gibi düzensizdir.

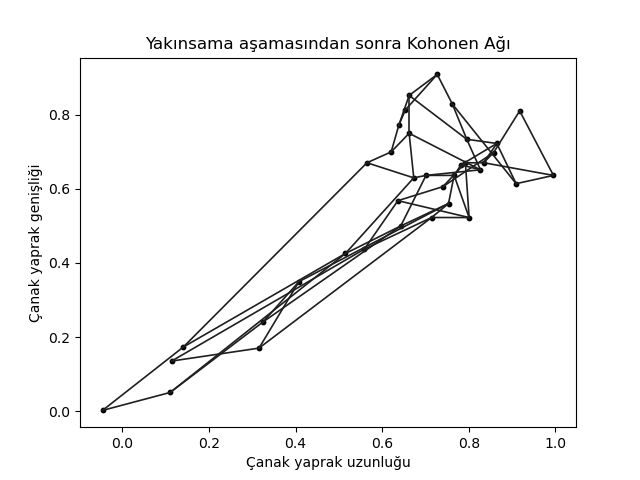
Özdüzenleme aşamasından sonra ağın veri öbeklerine göre düzenlendiği görülür. 4 boyutlu bir veri üzerinde eğitim yapıldığı için bu düzen hala Haykin’in örneğindeki gibi açık bir şekilde görülemez.

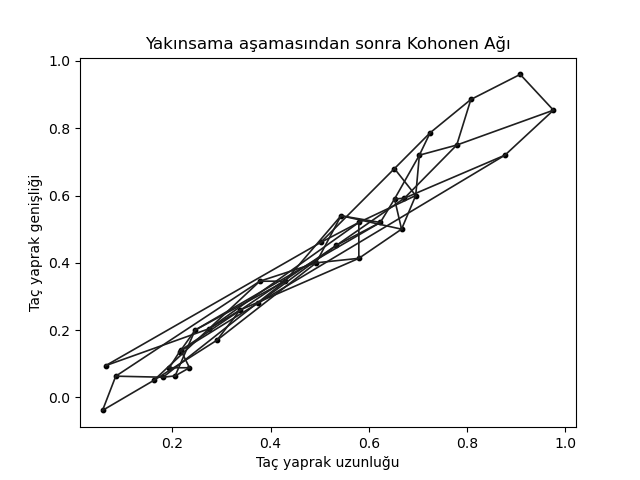
Yakınsama aşaması sonrası yine detaya yönelik değilimler olsa da bu problemde önceki problemden daha büyük değişimler görülür. Ağ, verileri temsil etmekte optimize olmaktadır.

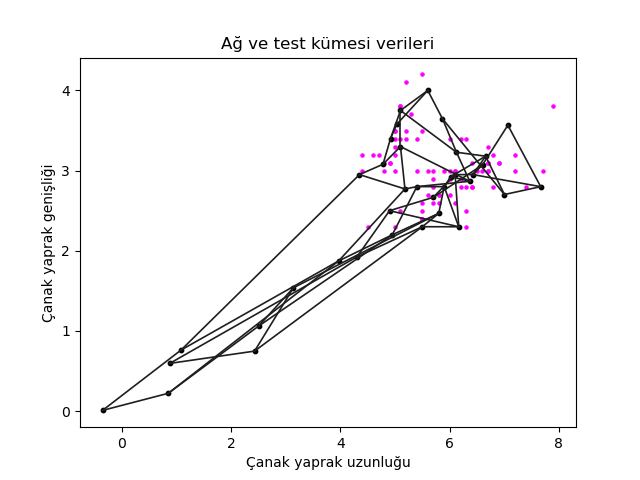
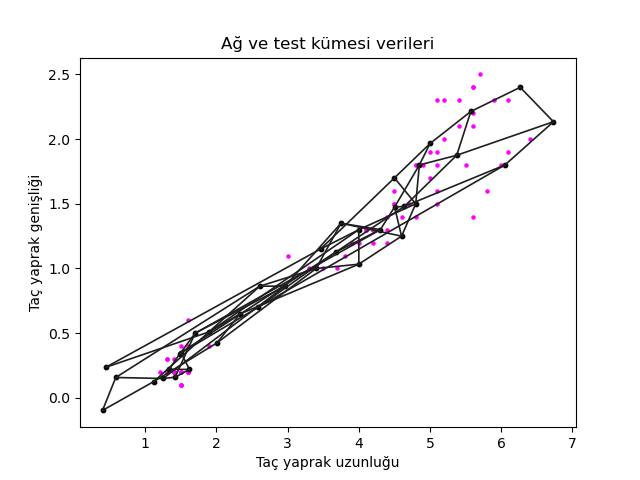
Eğitildikten sonra geri ölçeklenen ağ test kümesiyle karşılaştırıldığında veri öbeklerini başarılı bir şekilde temsil ettiği görülmektedir.



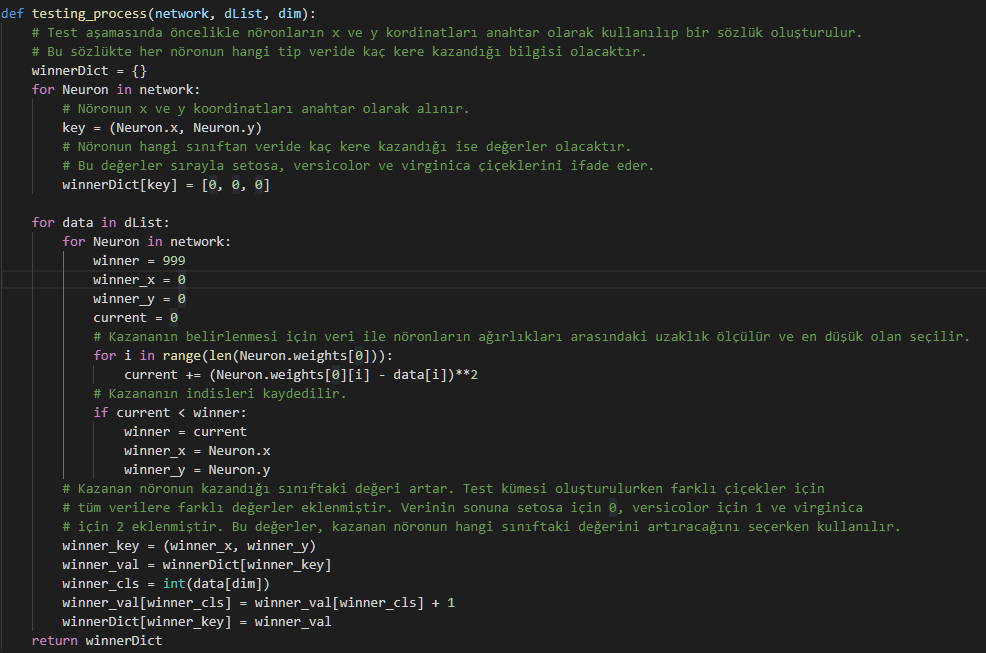
 

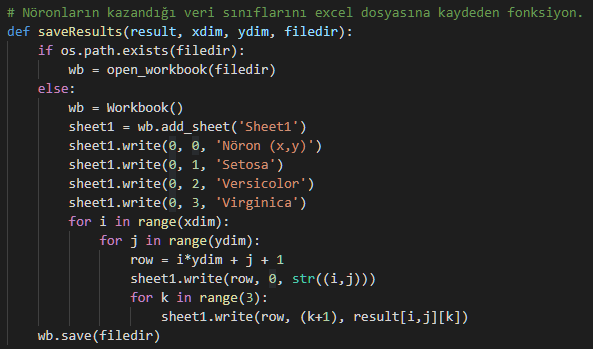




**Test aşaması:**

Eğitim fonksiyonuyla aynı şekilde her veri için kazanan nöron belirlenir. Bu nöron, (x,y) indislerinin anahtar ve setosa, versicolor ve virginica sınıflarının değer olduğu bir sözlükteki kazanılan veri sınıfının artırılmasında kullanılır. 

Elde edilen kazananlar sözlüğü, “p2results” klasöründe “irispred.xls” excel dosyasına yazdırılır. 

Nöronların temsil ettiği veri sınıfların karşılaştırıldığında, iki nöron haricinde birden fazla sınıfı temsil eden nöron olmadığı görülür. İki nöronda da azınlıkta olan sınıftan sadece tek bir veri temsil edilmektedir. Ek olarak çoğu nöronun herhangi bir veride kazanmadığı görülür.

Test kümesi 75 veriden oluşmaktadır. Bu durumda 2 verinin hatalı sınıflandırıldığı düşünülürse hata oranı %2.67’lik bir hata oranı gözlenir. Bu oran daha büyük bir ağ veya daha yüksek iterasyonlu bir eğitim ile düşürülebilir. Çok katmanlı algılayıcının %3’lük hata oranı ile karşılaştırıldığında daha isabetli bir sınıflandırma yaptığı görülür.