

Karar Ağacı (Decision tree) nedir?

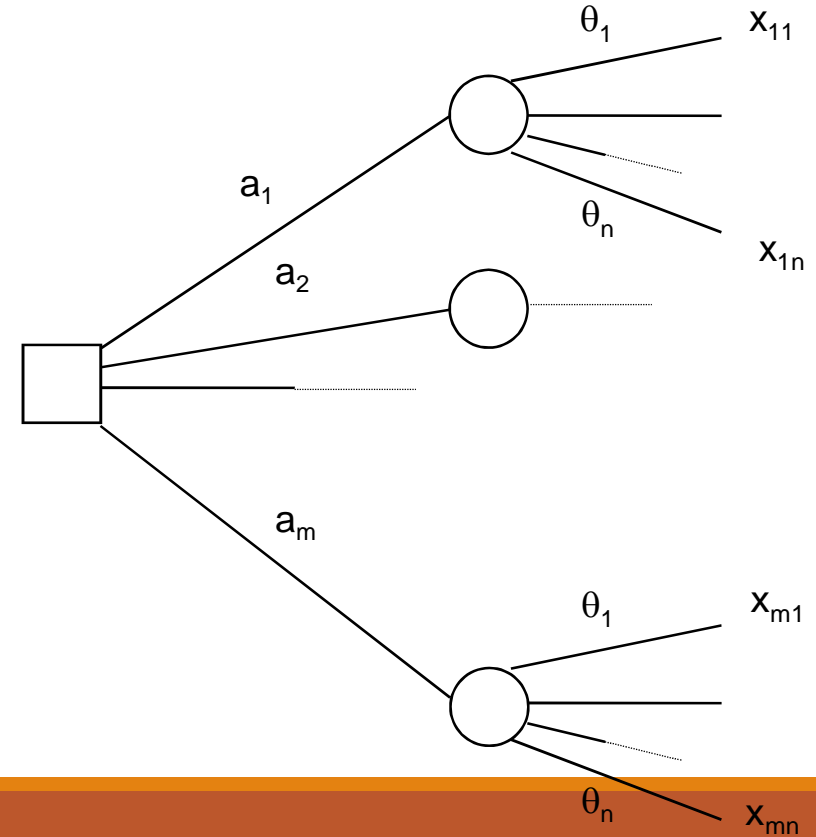
Bir işletme yönetimi tarafından tercihlerin, risklerin, kazançların, hedeflerin tanımlanmasında yardımcı olabilen ve birçok önemli yatırım alanlarında uygulanabilen, birbirini izleyen şansa bağlı olaylarla ilgili olarak çıkan çeşitli karar noktalarını incelemek için kullanılan bir tekniktir.

KARAR AĞACI YÖNTEMİ

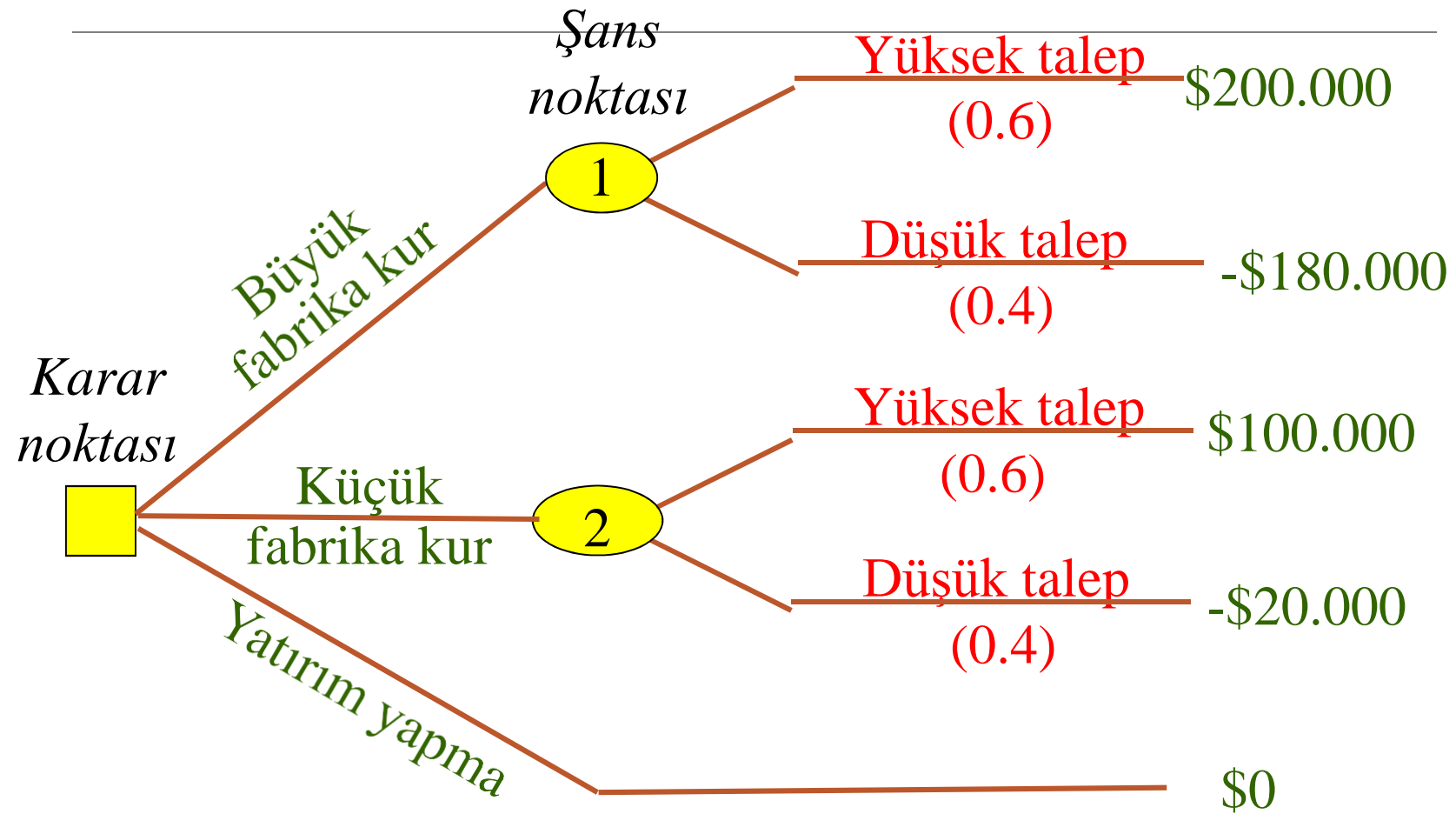
1. Sorunun tanımlanması
2. Karar ağacının çizilmesi / yapılandırılması
3. Olayların oluşma olasılıklarının atanması
4. Beklenen getirinin (veya faydanın) ilgili şans noktası için hesaplanması - geriye doğru, işlem
5. En yüksek beklenen getirinin (faydanın) ilgili karar noktasına atanması - geriye doğru, karşılaştırma
6. Önerinin sunulması

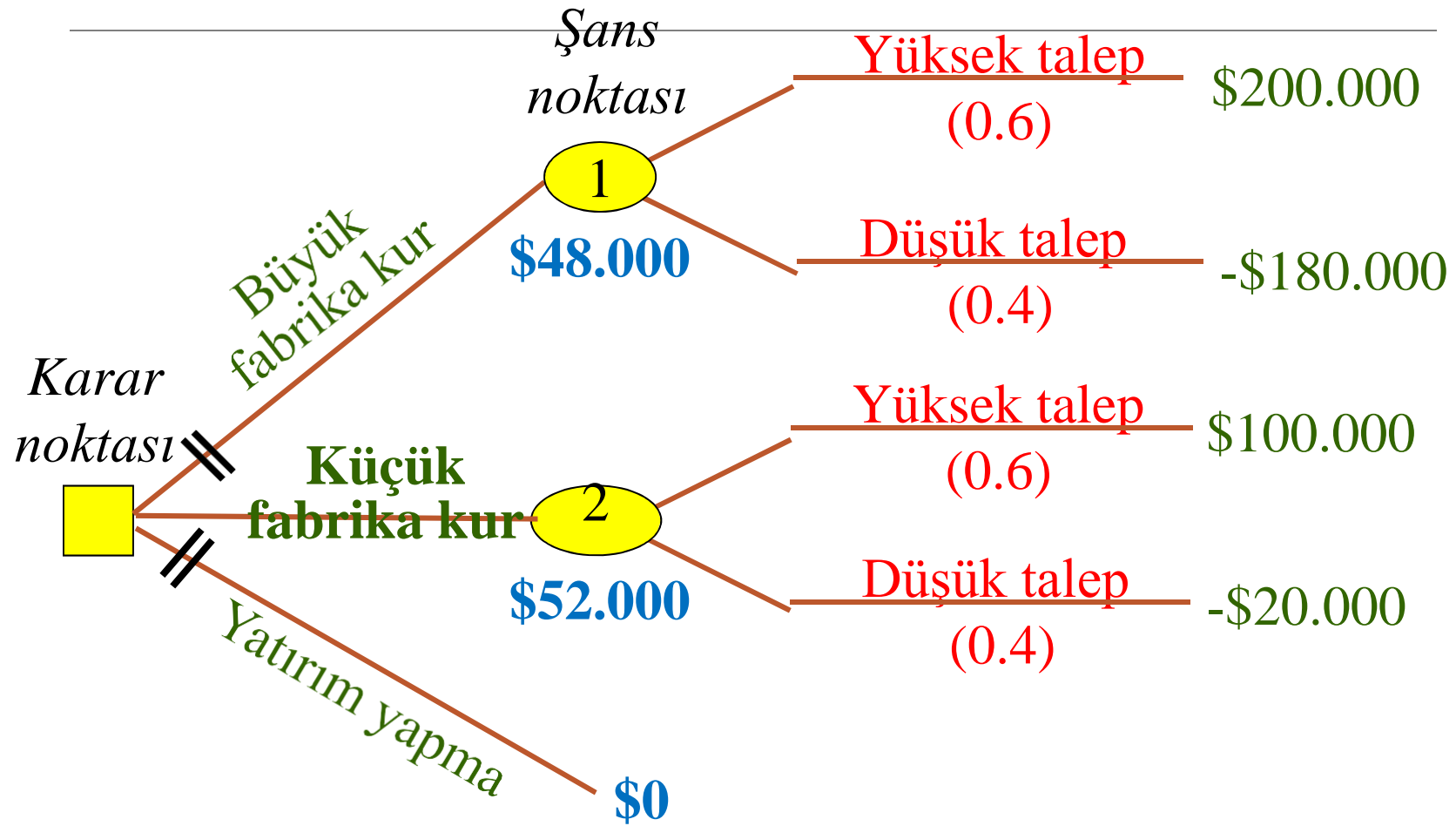
KARAR TABLOSUNUN KARAR AĞACINA DÖNÜŞTÜRÜLMESİ

SEÇENEKLER	OLAYLAR			
	θ_1	θ_2	...	θ_n
a_1	x_{11}	x_{12}	...	x_{1n}
a_2	x_{21}	x_{22}	...	x_{2n}
.
a_m	x_{m1}	x_{m2}	...	x_{mn}



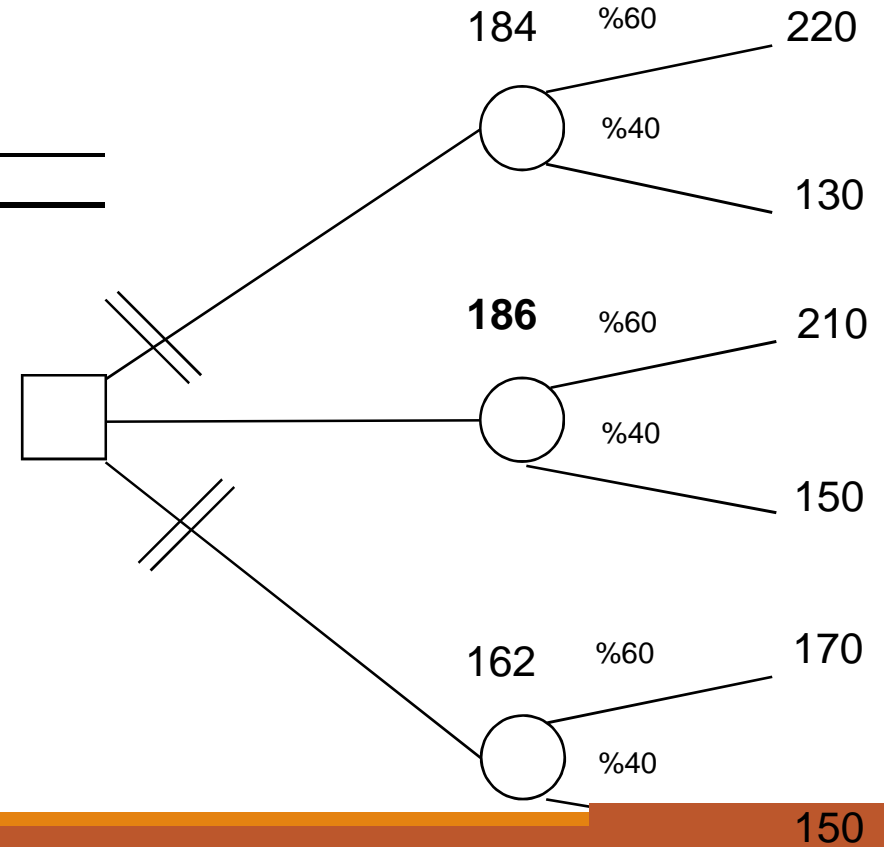
ÖRNEK 1





ÖRNEK 2

STRATEJİLER	OLAYLAR	
	Düşüş	Yükseliş
Yeni donanım (S_1)	130	220
Fazla mesai (S_2)	150	210
Bir şey yapmama (S_3)	150	170
OLASILIKLAR	40%	60%



Karar Ağaçları

Karar ağaçları eğitici öğrenme için çok yaygın bir yöntemdir. Algoritmanın adımları:

1. T öğrenme kümesini oluştur
2. T kümesindeki örnekleri en iyi ayıran niteliği belirle
3. Seçilen nitelik ile ağacın bir düğümünü oluştur ve bu düğümden çocuk düğümleri veya ağacın yapraklarını oluştur. Çocuk düğümlere ait alt veri kümesinin örneklerini belirle
4. 3. adımda yaratılan her alt veri kümesi için
 - Örneklerin hepsi aynı sınıfa aitse
 - Örnekleri bölecek nitelik kalmamışsa
 - Kalan niteliklerin değerini taşıyan örnek yoksa işlemi sonlandır. Diğer durumda alt veri kümesini ayırmak için 2. adımdan devam et.

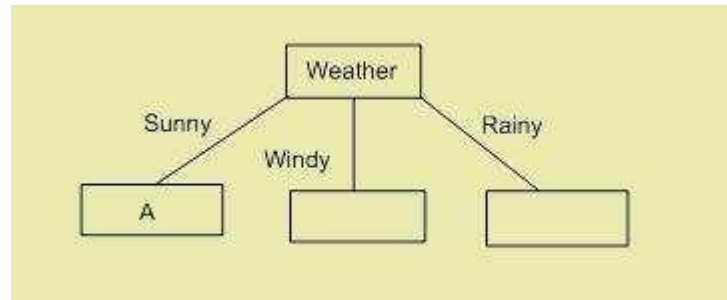
Karar Ağaçları: Haftasonu örneği

1. Adım: Veri setinden T öğrenme kümesi oluşturulur.

Weekend (Example)	Weather	Parents	Money	Decision (Category)
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W3	Windy	Yes	Rich	Cinema
W4	Rainy	Yes	Poor	Cinema
W5	Rainy	No	Rich	Stay in
W6	Rainy	Yes	Poor	Cinema
W7	Windy	No	Poor	Cinema
W8	Windy	No	Rich	Shopping
W9	Windy	Yes	Rich	Cinema
W10	Sunny	No	Rich	Tennis

Karar Ağaçları: Haftasonu örneği

2. Adım: Veri setindeki en ayırt edici nitelik belirlenir ve ağacın kökü olarak alınır.



- 3. Adım: Ağacın çocuk düğümü olan A düğümüne ait alt veri kümesi belirlenir.

Weekend (Example)	Weather	Parents	Money	Decision (Category)
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W10	Sunny	No	Rich	Tennis

Karar Ağaçları: En ayırt edici nitelik nasıl bulunur?

Bilgi Kazancı (Information Gain): ID3, C4.5 gibi karar ağacı metotlarında en ayırt edici niteliği belirlemek için her nitelik için **bilgi kazancı** ölçülür.

Bilgi Kazancı ölçümünde **Entropy** kullanılır.

Entropy rastgeleliği, belirsizliği ve beklenmeyen durumun ortaya çıkma olasılığını gösterir.

Karar Ağaçları

Bilgi Kazancı:Entropy

The information entropy of a discrete random variable X , that can take on possible values $\{x_1, \dots, x_n\}$ is

$$\begin{aligned} H(X) = E(I(X)) &= \sum_{i=1}^n p(x_i) \log_2 (1/p(x_i)) \\ &= - \sum_{i=1}^n p(x_i) \log_2 p(x_i) \end{aligned}$$

where

$I(X)$ is the information content or self-information of X , which is itself a random variable; and $p(x_i) = \Pr(X=x_i)$ is the probability mass function of X .

Karar Ağaçları:Entropy

Haftasonu veri kümesindeki (T kümesi) 10 örnekten

- 6 örnek için karar sinema
- 2 örnek için karar tenis oynamak
- 1 örnek için karar evde kalmak ve
- 1 örnek için karar alışverişe gitmek olduğuna göre

Entropy:

$$H(T) = - (6/10) \log_2(6/10) - (2/10) \log_2(2/10) - (1/10) \log_2(1/10) - (1/10) \log_2(1/10)$$

$$H(T) = 1,571$$

Karar Ağaçları: Bilgi Kazancı

A niteliğinin T veri kümesindeki bilgi kazancı:

$$\text{Gain}(T, A) = \text{Entropy}(T) - \sum P(v) \text{Entropy}(T(v))$$

- v: Values of A
- $P(v) = |T(v)| / |T|$

Karar Ağaçları: Bilgi Kazancı

Gain(T, weather)= ?

- Sunny=3 (1 Cinema, 2 Tennis)
- Windy=4 (3 Cinema, 1 Shopping)
- Rainy=3 (2 Cinema, 1 Stay in)

- Entropy(T_{sunny})= $-(1/3) \log_2 (1/3) - (2/3) \log_2 (2/3) = 0,918$
- Entropy(T_{windy})= $-(3/4) \log_2 (3/4) - (1/4) \log_2 (1/4) = 0,811$
- Entropy(T_{rainy})= $-(2/3) \log_2 (2/3) - (1/3) \log_2 (1/3) = 0,918$

$$\text{Gain}(T, \text{weather}) = \text{Entropy}(T) - ((P(\text{sunny})\text{Entropy}(T_{\text{sunny}}) + \\ P(\text{windy}) \text{Entropy}(T_{\text{windy}}) + P(\text{rainy}) \text{Entropy}(T_{\text{rainy}}))$$

$$= 1,571 - ((3/10)\text{Entropy}(T_{\text{sunny}}) + (4/10)\text{Entropy}(T_{\text{windy}}) + (3/10)\text{Entropy}(T_{\text{rainy}}))$$

$$\text{Gain}(T, \text{weather}) = 0,70$$

Karar Ağaçları:Bilgi Kazancı

Gain(T, parents)= ?

- Yes=5 (5 Cinema)
- No =5 (2 Tennis, 1 Cinema, 1 Shopping, 1 Stay in)
- Entropy(T_{yes})= - (5/5) \log_2 (5/5) = 0
- Entropy(T_{no})= - (2/5) \log_2 (2/5) - 3(1/5) \log_2 (1/5) =1,922

$$\text{Gain}(T, \text{parents}) = \text{Entropy}(T) - ((P(\text{yes})\text{Entropy}(T_{yes}) + P(\text{no}) \text{Entropy}(T_{no}))$$

$$=1,571 - ((5/10)\text{Entropy}(T_{yes})+(5/10)\text{Entropy}(T_{no}))$$

$$\text{Gain}(T, \text{parents})=0,61$$

Karar Ağaçları:Bilgi Kazancı

Gain(T, money)= ?

- Rich=7 (3 Cinema, 2 Tennis, 1 Shopping, 1 Stay in)
- Poor=3 (3 Cinema)
- Entropy(T_{rich})= 1,842
- Entropy(T_{poor})= 0

$$\text{Gain}(T, \text{money}) = \text{Entropy}(T) - ((P(\text{rich})\text{Entropy}(T_{rich}) + \\ P(\text{poor}) \text{Entropy}(T_{poor}))$$

$$=1,571 - ((5/10)\text{Entropy}(T_{rich})+(5/10)\text{Entropy}(T_{poor}))$$

$$\text{Gain}(T, \text{money})=0,2816$$

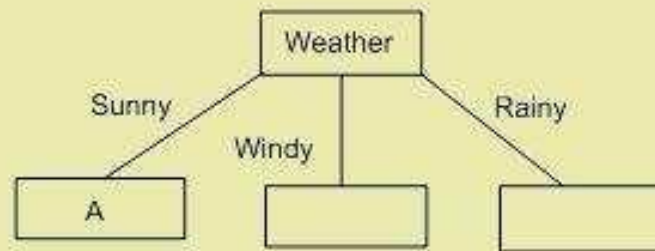
Karar Ağaçları: Bilgi Kazancı

$\text{Gain}(T, \text{weather}) = 0,70$

$\text{Gain}(T, \text{parents}) = 0,61$

$\text{Gain}(T, \text{money}) = 0,2816$

Weather özelliği en büyük bilgi kazancını sağladığı için ağacın kökünde yer alacak özellik olarak seçilir. Bu özellik en ayırt edici özellik olarak bulunmuştur.



Karar Ağaçları:

3. Adım: Ağacın çocuk düğümü olan A düğümüne ait alt veri kümesi belirlenir.

Weekend (Example)	Weather	Parents	Money	Decision (Category)
W1	Sunny	Yes	Rich	Cinema
W2	Sunny	No	Rich	Tennis
W10	Sunny	No	Rich	Tennis

Her alt küme için tekrar bilgi kazancı hesaplanarak en ayırt edici özellik belirlenir.

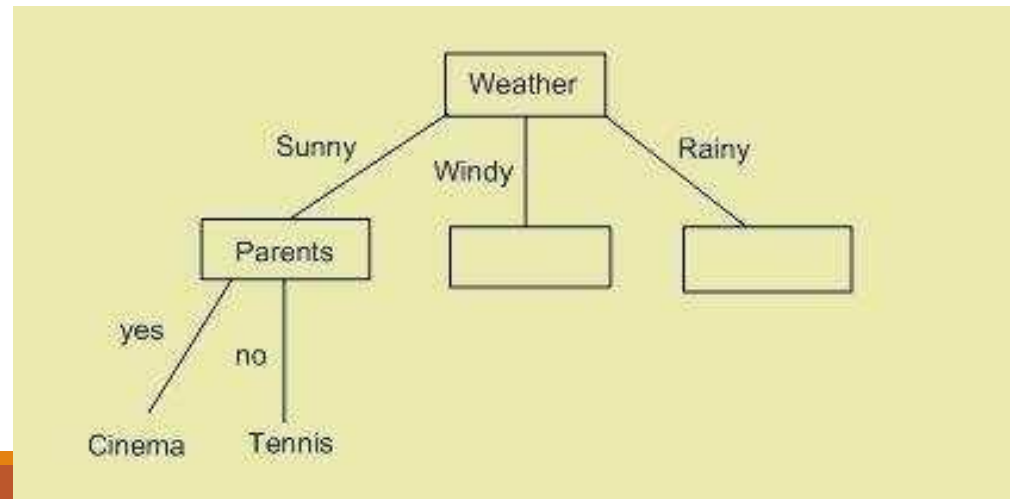
$$\begin{aligned}\text{Gain}(S_{\text{sunny}}, \text{parents}) &= 0.918 - (|S_{\text{yes}}|/|S|) * \text{Entropy}(S_{\text{yes}}) - (|S_{\text{no}}|/|S|) * \text{Entropy}(S_{\text{no}}) \\ &= 0.918 - (1/3) * 0 - (2/3) * 0 = 0.918\end{aligned}$$

$$\begin{aligned}\text{Gain}(S_{\text{sunny}}, \text{money}) &= 0.918 - (|S_{\text{rich}}|/|S|) * \text{Entropy}(S_{\text{rich}}) - (|S_{\text{poor}}|/|S|) * \text{Entropy}(S_{\text{poor}}) \\ &= 0.918 - (3/3) * 0.918 - (0/3) * 0 = 0.918 - 0.918 = 0\end{aligned}$$

Karar Ağaçları

Yeni düğüm için en ayırt edici özellik **Parents** olarak belirlenmiştir. Bu işlemler her düğüm için aşağıdaki durumlardan biri oluşuncaya kadar devam eder

- Örneklerin hepsi aynı sınıfa ait
- Örnekleri bölecek özellik kalmamış
- Kalan özelliklerin değerini taşıyan örnek yok



Karar Agacı kullanarak sınıflandırma

Avantajları:

- Karar ağacı oluşturmak zahmetsizdir
- Küçük ağaçları yorumlamak kolaydır
- Anlaşılabilir kurallar oluşturulabilir
- Sürekli ve ayrık nitelik değerleri için

kullanılabilir

Karar Agacı kullanarak sınıflandırma

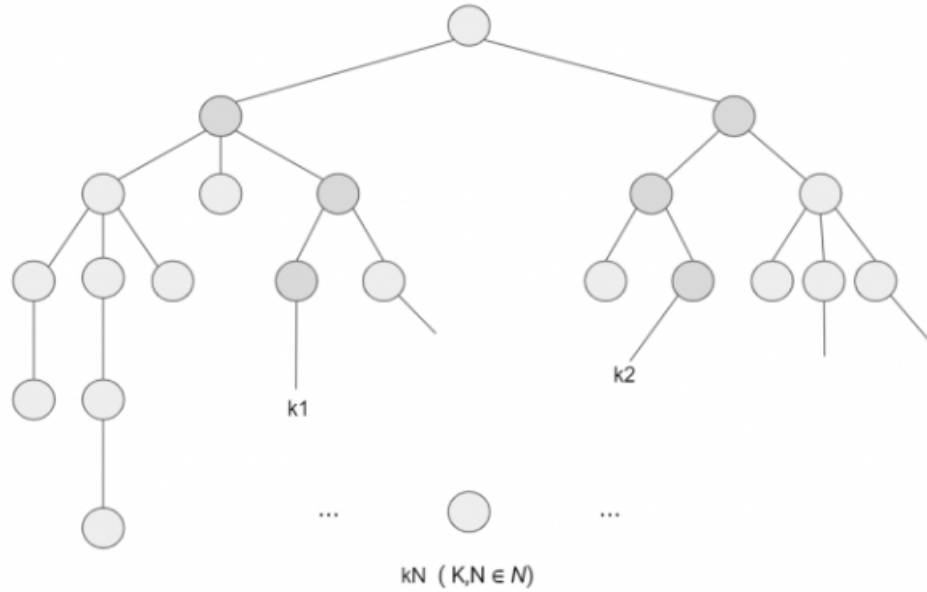
Dezavantajları:

- Sürekli nitelik değerlerini tahmin etmekte çok başarılı değildir
- Sınıf sayısı fazla ve öğrenme kümesi örnekleri sayısı az olduğunda model oluşturma çok başarılı değildir
- Zaman ve yer karmaşıklığı öğrenme kümesi örnekleri sayısına, nitelik sayısına ve oluşan ağacın yapısına bağlıdır
- Hem ağaç oluşturma karmaşıklığı hem de ağaç budama karmaşıklığı fazladır

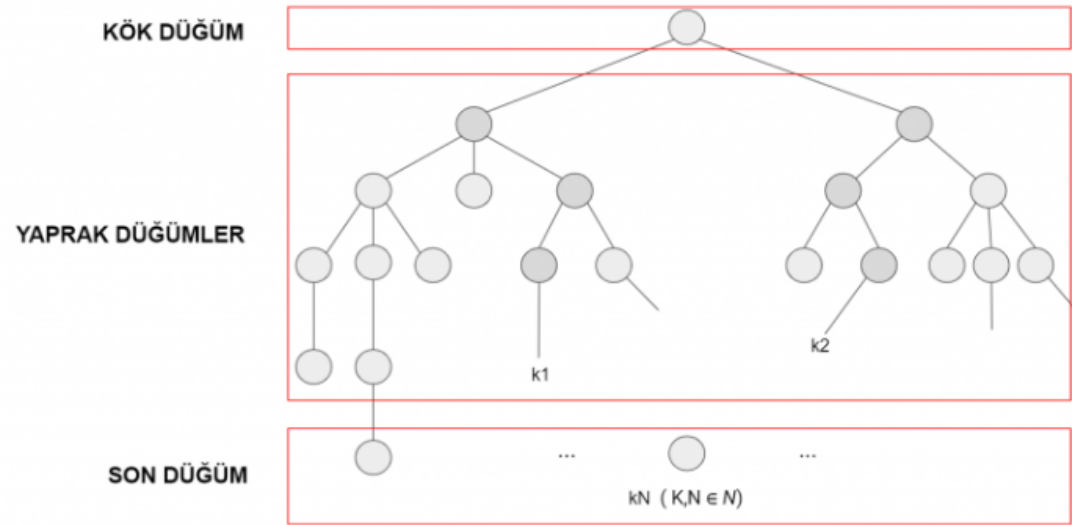
1-Karar Ağaçları - Decision Tree

Karar ağaçları; **Regresyon** modellerine benzeyen tek bağımlı değişken ve birçok bağımsız değişken içeren bir sınıflandırma algoritmasıdır.

Karar Ağacı uygulamalarında ağırlıklı olarak veriler üzerinde uygulanan çeşitli regresyon modelleri için alternatif örüntü modeli keşfi gerçekleştirilir.



Karar ağacının araştırmayı hedeflediğimiz sınıfı için başlangıcı sağlayan düğüme **kök düğüm**, ara adımlar içeren düğümlerine **yaprak düğüm**, ağacın bittiği son adıma ise **son düğüm** denir.



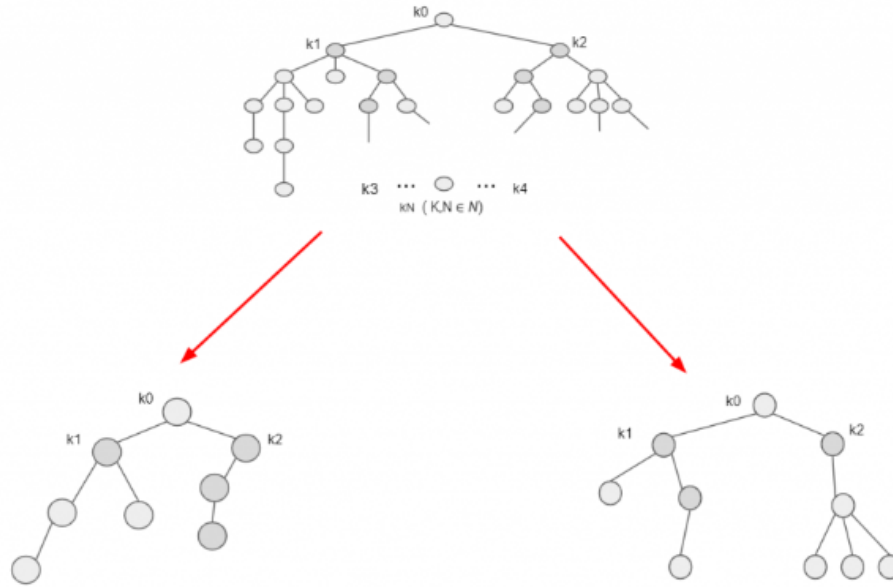
İlgili veri kümesine uygulanacak hedef araştırma içeriğine göre **yaprak düğüm sayısı** artabilir ya da azalabilir. Araştırma üzerinde uygulanan karar ağacı modelindeki toplam düğüm sayısı ise karar ağacının **derinliğini** ifade etmektedir.

Karar Ağacı ile analiz ve araştırma gerçekleştirilirken her bir düğüm adımındaki değişkenler test edilir. (Çeşitli senaryolar kurgulanır ve test edilmiş olunur.)

Belirlenen veya baz alınan değişkenlerin takibi ile ana karar ağacı yeniden oluşturulur ve yeni oluşumda başlangıca göre karar ağacı derinliği değişkenlik gösterebilir.

Nedeni ise aranan örgü yapısının son düğümünden önce sonuçlanması (yeni son düğüm) veya yaprak düğümlerin çeşitli dallanmalarının budanmasıdır (kırpılmasıdır). Aynı nedensel durumlar aranan örgü yapısından ziyade baz alınan parametreler içinde geçerlidir.

En verimli modeli saptayabilmek için, çokça çeşitli örgüleri ya da parametreleri baz alıp yeni karar ağacı modelleri oluşturmaya çalışmak gerekmektedir.



Budama (kırpma) yöntemleri genel olarak **Ön Budama** ve **Son Budama** olarak iki sınıfta ele alınmaktadır.

Ön budama; ağacın dallandırılma aşamasında, ayırma için kullanılan istatistiki kriterler, gini indeksi veya kazanım oranı için belirli eşik değerler konularak ağacın o düğümden sonra büyümemesini esas alınır.

Son Budama'da ise, bütün karar ağacı oluşturularak, son hali üzerinden küçültme işlemi gerçekleştirilir.

En çok tercih edilen karar ağacı algoritmaları;

- **CART**
- **CHAID**
- **C4.5**
- **ID3**
- **MARS**

Karar Ağacı Testi

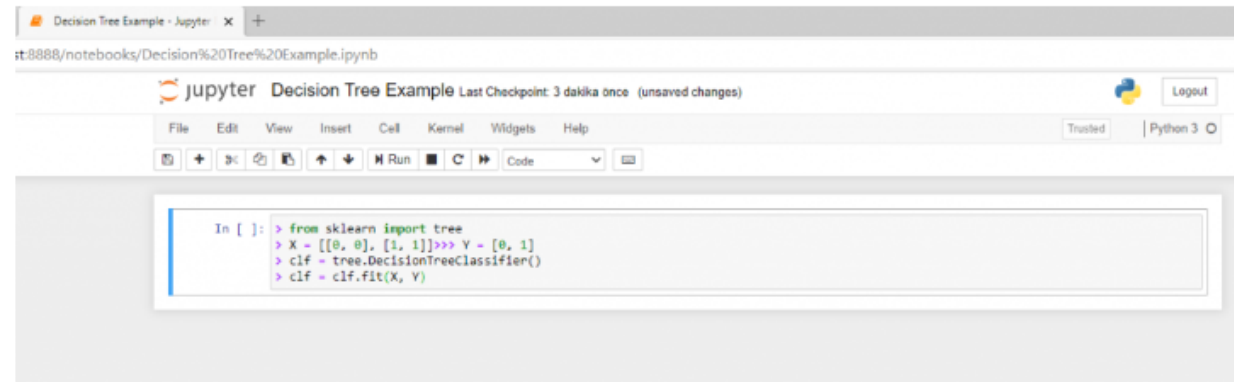
İfadesel olarak dile getirdiğimiz Karar Ağacı algoritmasını uygulamalı olarak Python ve Microsoft Power BI üzerinde test etmeye çalışalım.

Python üzerinde test etmeden önce ilgili derleyici için Anaconda Navigator üzerinden Jupyter Notebook ile işlemleri ele almaya çalışacağım. [Anaconda Navigator](#)

Hızlı bir şekilde scikit-learn üzerinde bulunan açıklayıcı örnek ile karar ağacı algoritmasını ele alalım;

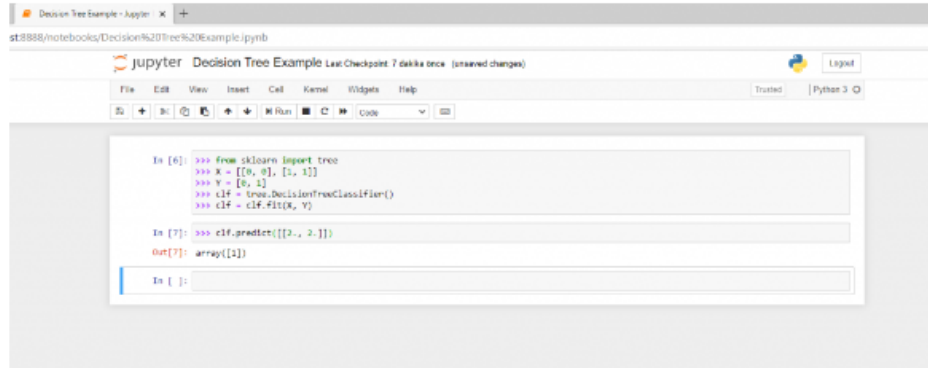
1-Karar Ağacı algoritmam için scikit-learn kütüphanemi yükleyerek eğitim ve değer dizilerimin tanımlamasını gerçekleştiriyorum;

```
>>> from sklearn import tree
>>> X = [[0, 0], [1, 1]]
>>> Y = [0, 1]
>>> clf = tree.DecisionTreeClassifier()
>>> clf = clf.fit(X, Y)
```



2-Yapımı algoritmaya uygun kurduktan sonra örnek sınıfını tahmin için kullanmaya başlıyorum;

```
>>> clf.predict([[2., 2.]])
```



A screenshot of a Jupyter Notebook titled "Decision Tree Example". The notebook is open to a file named "st8888/notebooks/Decision%20Tree%20Example.ipynb". The interface shows a menu bar with options like File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu bar, there are icons for running, saving, and other notebook functions. The main area contains two code cells. The first cell (In [6]) contains the following code:

```
from sklearn import tree
X = [[0, 0], [1, 1]]
Y = [0, 1]
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)
```

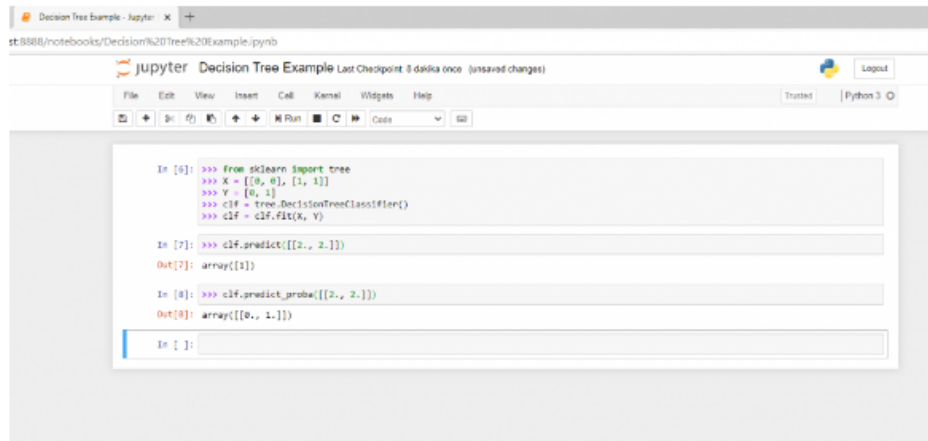
 The second cell (In [7]) contains the code:

```
clf.predict([[2., 2.]])
```

 The output of the second cell is displayed as "Out[7]: array([1])".

3-Alternatif olarak aynı eğitim sınıfında olan eğitim örnekleri ile, diğer yapraklardaki değer olasılığını tahminlemeye başlıyorum;

```
>>> clf.predict_proba([[2., 2.]])
```



A screenshot of a Jupyter Notebook titled "Decision Tree Example". The notebook is open to a file named "st8888/notebooks/Decision%20Tree%20Example.ipynb". The interface shows a menu bar with options like File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu bar, there are icons for running, saving, and other notebook functions. The main area contains three code cells. The first cell (In [6]) contains the following code:

```
from sklearn import tree
X = [[0, 0], [1, 1]]
Y = [0, 1]
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)
```

 The second cell (In [7]) contains the code:

```
clf.predict([[2., 2.]])
```

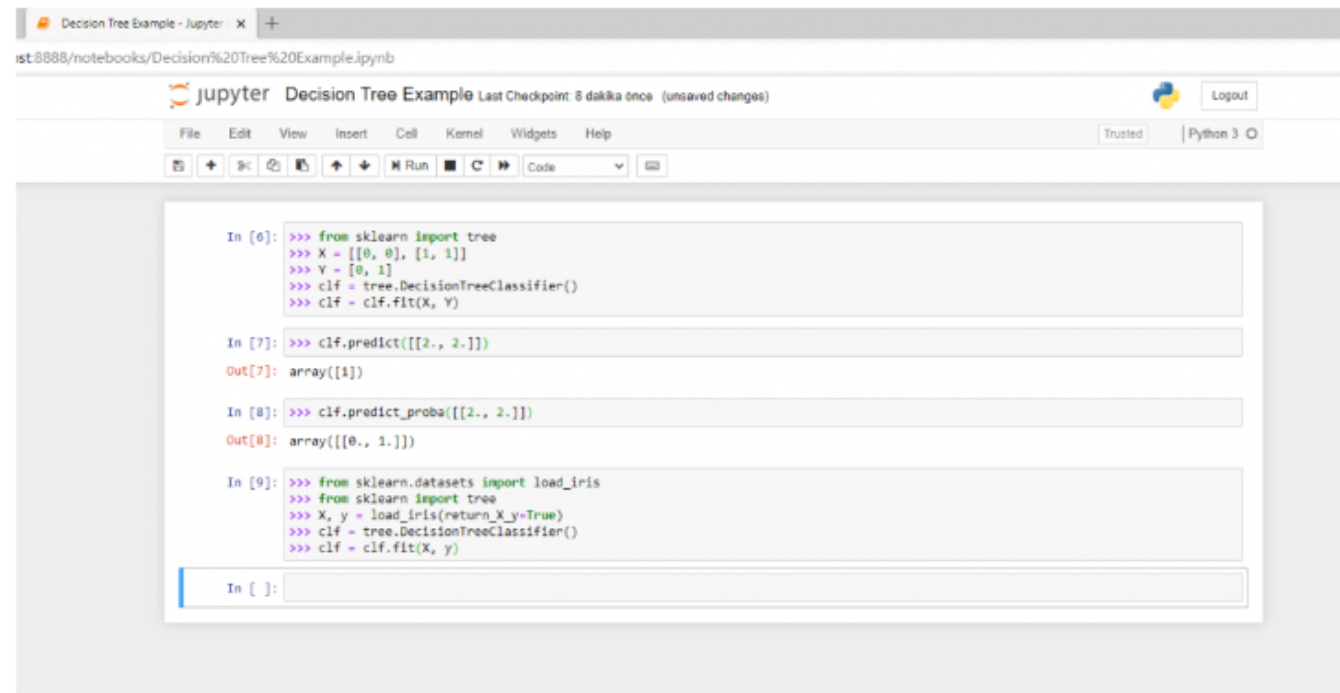
 The output of the second cell is displayed as "Out[7]: array([1])". The third cell (In [8]) contains the code:

```
clf.predict_proba([[2., 2.]])
```

 The output of the third cell is displayed as "Out[8]: array([[0., 1.]])".

4-Modeli oluşturduktan sonra model eğitimi için, hazır veriler içeren **iris** veritabanı (Genel bitki özellikleri veri bankası) üzerinden çeşitli bitki özellikleri ile oluşan genetik yapılanmayı ağaç diagramı olarak kurgulayıp-yapılandırıyorum;

```
>>> from sklearn.datasets import load_iris
>>> from sklearn import tree
>>> X, y = load_iris(return_X_y=True)
>>> clf = tree.DecisionTreeClassifier()
>>> clf = clf.fit(X, y)
```



The screenshot shows a Jupyter Notebook interface with the title 'Decision Tree Example'. The notebook contains three code cells. The first cell imports the necessary libraries and loads the Iris dataset. The second cell trains a Decision Tree classifier. The third cell uses the trained classifier to predict the class for a specific input. The output of the third cell is an array containing the predicted class.

```
In [0]: >>> from sklearn import tree
>>> X = [[0, 0], [1, 1]]
>>> y = [0, 1]
>>> clf = tree.DecisionTreeClassifier()
>>> clf = clf.fit(X, y)

In [7]: >>> clf.predict([[2., 2.]])
Out[7]: array([1])

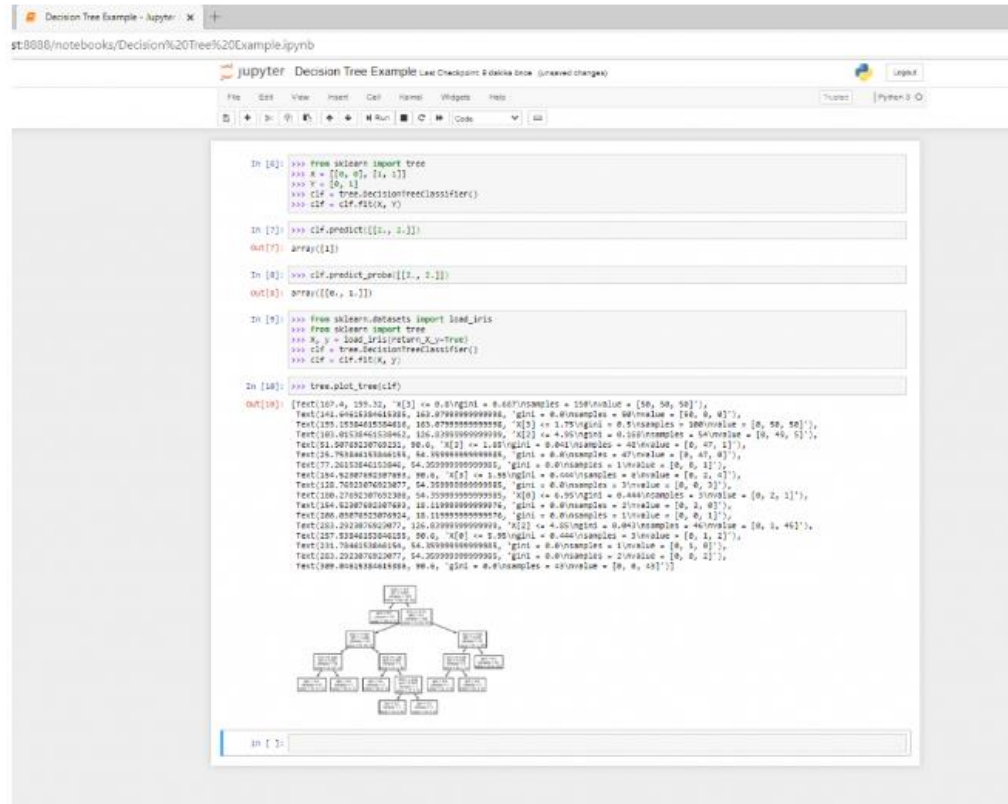
In [8]: >>> clf.predict_proba([[2., 2.]])
Out[8]: array([[0., 1.]])

In [9]: >>> from sklearn.datasets import load_iris
>>> from sklearn import tree
>>> X, y = load_iris(return_X_y=True)
>>> clf = tree.DecisionTreeClassifier()
>>> clf = clf.fit(X, y)

In [ ]:
```

5-Model eğitimi tamamlayıp yapılandırmayı oluşturduktan sonra da karar ağacını çizdiriyoruz;

```
>>> tree.plot_tree(clf)
```



sonuç çıktısı olarak tüm değerleri ve ağaç diagramını elde ediyoruz.

Burada ki örnek üzerinde budama işlemine yönelik herhangi bir faaliyet ya da ek işlem gerçekleştirmedik. İlgili veritabanı üzerinden çeşitli değerleri baz alarak eğitim ya da analiz-değer verilerinden budanacak verileri değerleri çıkartabilir ve diagram yapımızı yeniden yapılandırabiliriz.

Çıktı olarak aldığımız karar ağacı modelimizin derinliği ise 6'dır; 1 kök, 4 yaprak, 1'de son düğümden