2.3 Veriyi Almak

Probleminize uygun veri setini bulmalısınız. Ne kadar veriye ihtiyacınız olduğunu ve veriyi çalışma alanınıza yüklediğinizde nasıl bir formatta çalışacağınızı kararlaştırmalısınız

1. Hangi veriye ve ne kadar veriye ihtiyacınız var?
2. Veriniz ne kadar saklama alanına mal olacaktır?
3. Çalışma alanınızı oluşturun.
4. Veriyi yükleyin.
5. Veriyi kolay kullanabileceğiniz bir biçime dönüştürün.
6. Verilerin boyutunu ve türünü kontrol edin.
7. Test setinizi oluşturun.

2.3.1 Çalışma Alanı Oluşturmak

Çalışma alanı oluşturulurken kullanılacak programlama diline ve programlara karar verilmedir. Biz örneğimizi python ile gerçekleştireceğiz. İsterseniz sanal bir ortam kurabilir ve kullanılacak kütüphaneleri oraya yükleyebilirsiniz.

2.3.2 Veriyi Yüklemek

Önce gerekli kütüphaneleri sonra da veri setini çalışma alanımıza yükleyelim.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt


veri_seti = sns.load_dataset('tips')
```

2.3.3 Veriyi Hızlıca Gözden Geçirmek

Veri setinin içeriğini anlamak için veriyi hızlıca gözden geçirelim.

```
print veri_seti.head()
```

```
 total_bill   tip     sex smoker  day    time  size
0     16.99  1.01  Female    No  Sun  Dinner     2
1     10.34  1.66    Male    No  Sun  Dinner     3
2     21.01  3.50    Male    No  Sun  Dinner     3
3     23.68  3.31    Male    No  Sun  Dinner     2
4     24.59  3.61  Female    No  Sun  Dinner     4
```

Veri setinde 7 özellik var; *total_bill, tips, sex, smoker, day, time, size*. Veri seti cinsiyet ve gün gibi kategorik değişkenler ve hesap ile bahşiş gibi sayısal değişkenler içeriyor.

```
print veri_seti.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 244 entries, 0 to 243
Data columns (total 7 columns):
total_bill    244 non-null float64
tip           244 non-null float64
sex           244 non-null category
smoker        244 non-null category
day           244 non-null category
time          244 non-null category
size          244 non-null int64
dtypes: category(4), float64(2), int64(1)
memory usage: 7.2 KB
None
```

Veri setinde 244 tane örnek var. Hiç eksik değerimiz yok. Makine öğrenmesi için bu kadar veri, çok az olmasına rağmen başlangıç için mükemmel.

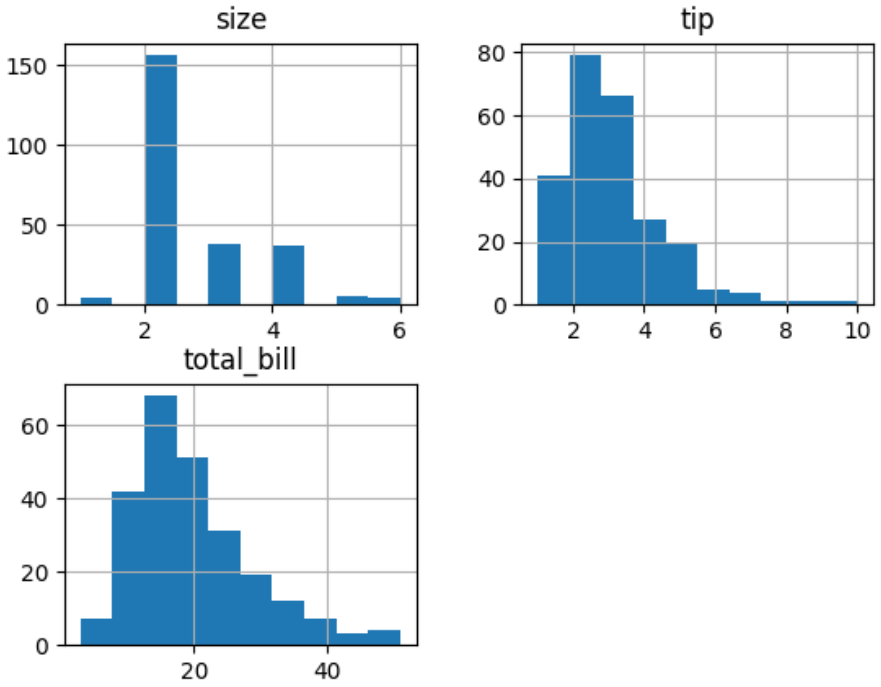
```
print veri_seti.describe()
```

```
       total_bill         tip        size
count  244.000000  244.000000  244.000000
mean    19.785943    2.998279    2.569672
std      8.902412    1.383638    0.951100
min      3.070000    1.000000    1.000000
25%     13.347500    2.000000    2.000000
```

```
50%   17.795000   2.900000   2.000000
75%   24.127500   3.562500   3.000000
max    50.810000  10.000000   6.000000
```

Sayısal değişkenler ile ilgili bilgilere baktığımızda daha önceden tanımladığımız problemler için varsayımlarda bulunabileceğimizi görebilirsiniz. 26.5 lira hesap ödeyen bir kadın ne kadar bahşiş bırakır sorusuna örneğin ortalama 3 lira bahşiş bırakır diyebiliriz.

```
veri_seti.hist()
plt.show()
```



Yukarıdaki grafiklere baktığımızda ise ödenen hesap ve bahşiş miktarlarının dağılımlarını görebiliyoruz.

2.3.4 Veriyi Bölmek

Rastgele bazı örnekleri, genellikle veri kümesinin % 20'sini seçmek test setini oluşturmanın en kolay yoludur. Biz de rastgele % 20'lik verilerden test setini aşağıdaki gibi oluşturabiliriz.

```
def bol_test_egitim(veri, oran):
  karisik_indeks = np.random.permutation(len(veri))
```

```
  test_buyuklugu = int(len(veri) * oran)
  test_indeksleri = karisik_indeks[:test_buyuklugu]
  egitim_indekleri = karisik_indeks[test_buyuklugu:]
  return veri.iloc[egitim_indekleri], veri.iloc[test_indeksleri]


egitim_seti, test_seti = bol_test_egitim(veri_seti, oran=0.2)
```

Verileri bu metodla ayırmak aslında pek de mükemmel bir yöntem değil. Bu şekilde verileri ayırdığımızda aslında program her çalıştırıldığında birbirinden farklı rastgele indeksler oluşturacağı için, hep farklı test setimiz olacaktır, ancak şimdilik bunun bizim için bir önemi yok.