RAPOR

Öncelikle veri setini tanımak için ne kadarının boş olduğunu, aykırı değer bulunup bulunmadığını tespit etmek için Python kullanarak gerekli işlemleri uyguladım. Bu işlemleri yaparken proje açıklamasında da belirtildiği gibi 2 kolonu (Pregnancy ve Diabetes) işlem dışı bıraktım ve aşağıdaki sonuçları elde ettim.

Eksik veriler

```
Glucose alanindaki boş veri sayısı, 5
BP alanindaki boş veri sayısı, 35
SkinThickness alanindaki boş veri sayısı, 227
Insulin alanindaki boş veri sayısı, 374
BMI alanindaki boş veri sayısı, 11
PedigreeFunc alanindaki boş veri sayısı, 0
Age alanindaki boş veri sayısı, 0
```

Kolon IQR'ları

```
Glucose icin limit degerler: (38.5, 202.5)
BP icin limit degerler: (40.0, 104.0)
SkinThickness icin limit degerler: (1.0, 57.0)
Insulin icin limit degerler: (-93.75, 360.25)
BMI icin limit degerler: (13.849999999999998, 50.25)
```

Aykırı Değer Sayıları (Kolon Başına)

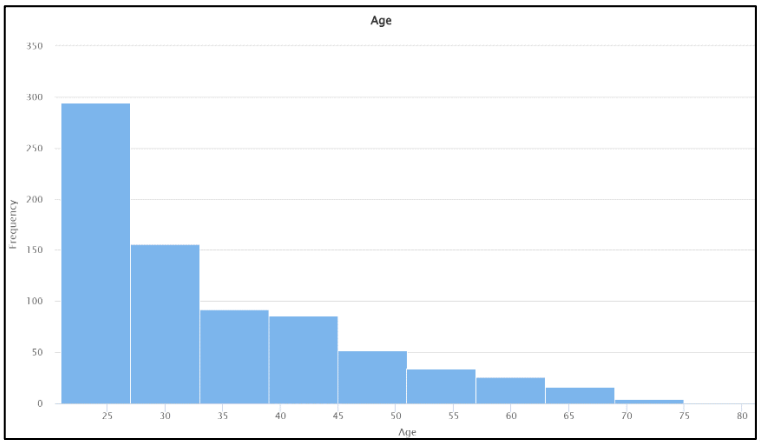
```
Glucose için aykırı değer sayısı 0.
BP için aykırı değer sayısı 14.
SkinThickness için aykırı değer sayısı 3.
Insulin için aykırı değer sayısı 24.
BMI için aykırı değer sayısı 8.
```

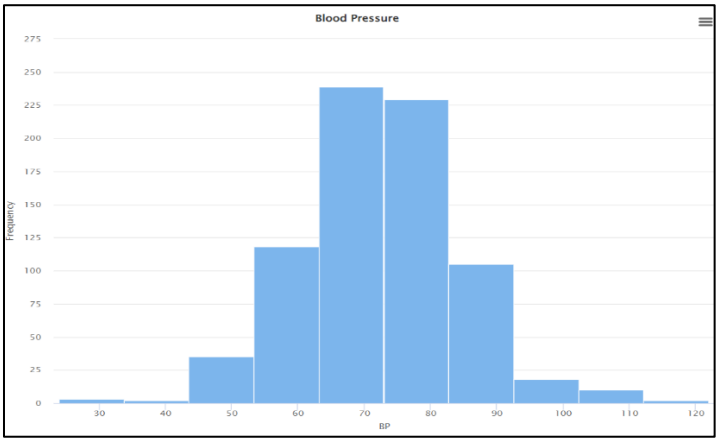
Bilgilerin Yarısından Fazlası Boş Olan Kayıtlar

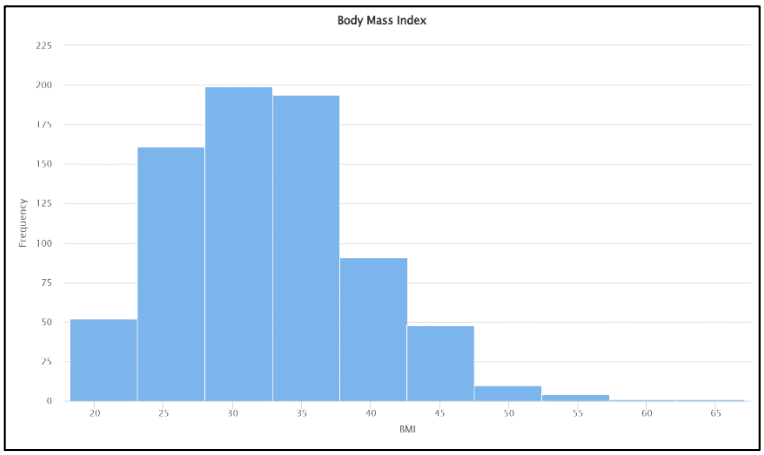
```
satir: 51 silinebilir, 4 eksik veri!!
satir: 62 silinebilir, 4 eksik veri!!
satir: 83 silinebilir, 4 eksik veri!!
satir: 428 silinebilir, 4 eksik veri!!
satir: 496 silinebilir, 4 eksik veri!!
satir: 524 silinebilir, 4 eksik veri!!
satir: 708 silinebilir, 4 eksik veri!!
```

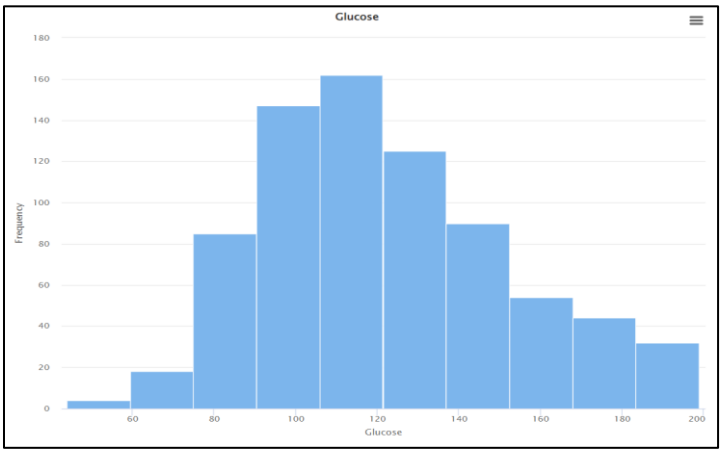
Bu işlemlerden sonra bilgilerin yarısı eksik olan satırları veri setinden tamamen çıkardım. Eksik verileri IQR sınırlarına ve sınıf etiketlerine dikkat ederek ait oldukları kümenin ortalama değerleri ile doldurdum.

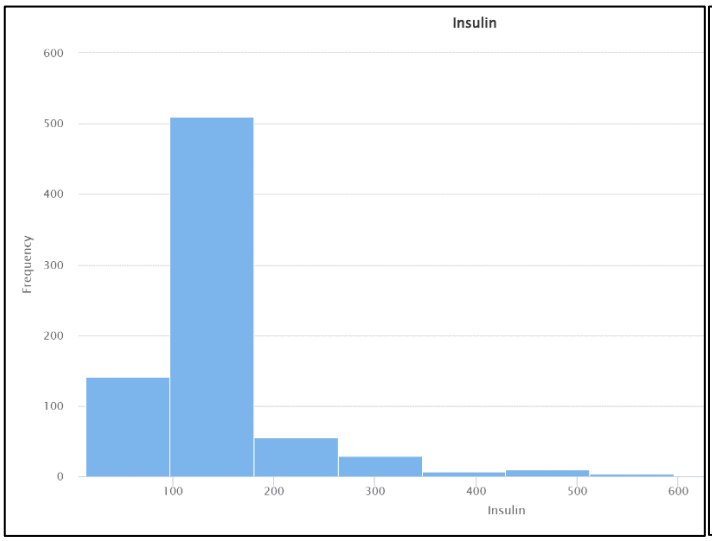
Sonrasında dağılım grafiklerini oluşturduğumda aşağıdaki sonuçları elde ettim.

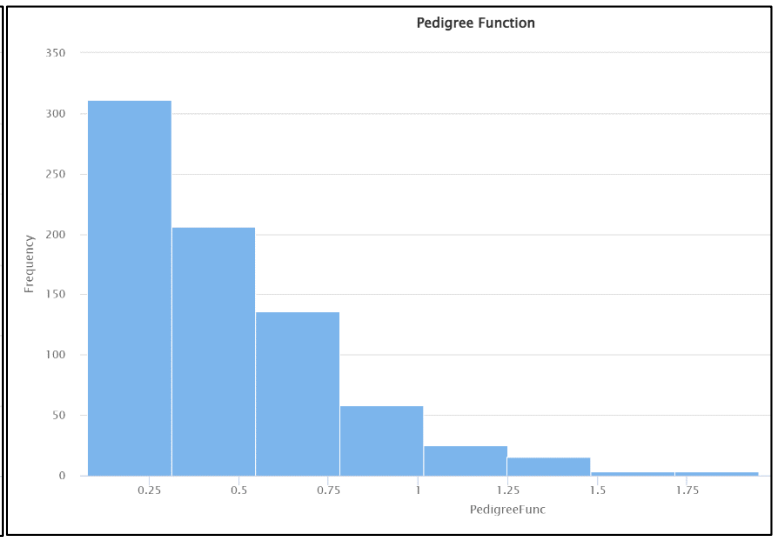


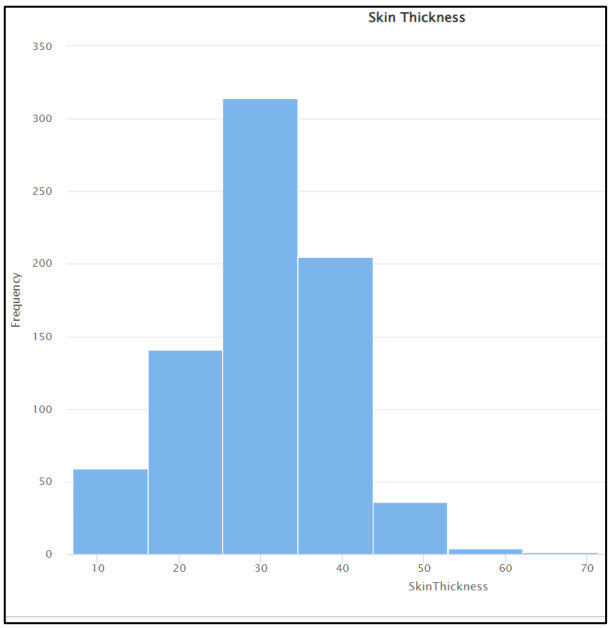


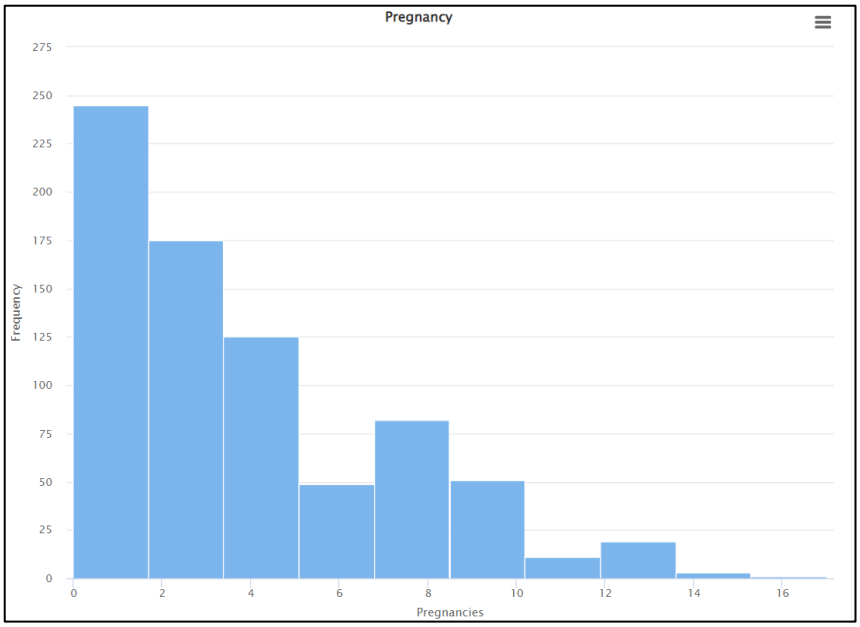





Insulin
600
500
400
300
200
100
0
Frequency
100
200
300
400
500
600
Insulin


Pedigree Function
350
300
250
200
150
100
50
0
Frequency
0.25
0.5
0.75
1
1.25
1.5
1.75
PedigreeFunc


Skin Thickness
350
300
250
200
150
100
50
0
Frequency
10
20
30
40
50
60
70
SkinThickness


Pregnancy
275
250
225
200
175
150
125
100
75
50
25
0
Frequency
0
2
4
6
8
10
12
14
16
Pregnancies

Eksik verileri doldurduktan/sildikten sonra sıra sınıflama algoritmalarını uygulamaya geldi. Bunu yaparken “RapidMiner” yazılımından yararlandım. Programda 2 seçenek bulunuyordu; algoritmaları el ile kurmak veya veri setini verip programın kıyas yapmasını beklemek. Ben öncelikle kendim 3 algoritma uyguladım daha sonra programın AutoModel özelliğiyle de kıyaslama yaptım.

Kendi kullandığım algoritmalar k-NN, Karar Ağacı ve Sinir Ağları.

Loop Parameters (15 rows, 3 columns)

itera...	k-NN.k	accuracy
1	1	0.846
2	3	0.860
3	5	0.864
4	7	0.882
5	9	0.882
6	11	0.899
7	13	0.904
8	15	0.890
9	16	0.895
10	18	0.899
11	20	0.882
12	22	0.882
13	24	0.882
14	26	0.873
15	28	0.855

k-NN:

Bu yöntem için önemli olan optimum “k” değerini bulmak olduğundan bir döngü içerisinde değerlerini 1 ile 28 arasında değerleri atayıp Accuracy sonuçlarını ölçtüm ve yandaki tabloda bulunan sonuçları elde ettim. Elde ettiğim sonuca göre optimum k değeri 13, maksimum Accuracy ise yaklaşık %90.

Karar Ağacı (Decision Tree)**:**

Yazılımda gerekli ayarlamaları yapıp verileri karar ağacına gönderdiğimde aşağıdaki tabloda da görüldüğü gibi yaklaşık olarak %84 accuracy elde ettim.

accuracy: 84.88% +/- 2.95% (micro average: 84.89%)

	true 1	true 0	class precision
pred. 1	212	60	77.94%
pred. 0	55	434	88.75%
class recall	79.40%	87.85%	

Sinir Ağı:

Yine gerekli ara işlemleri yapıp verileri bir sinir ağı sistemine oturttuğumda yaklaşık %90 acuuracy elde ettim. Buradaki sonuçlar karar ağacı sonuçlarına göre daha yüksek doğruluktaydı.

accuracy: 90.28%

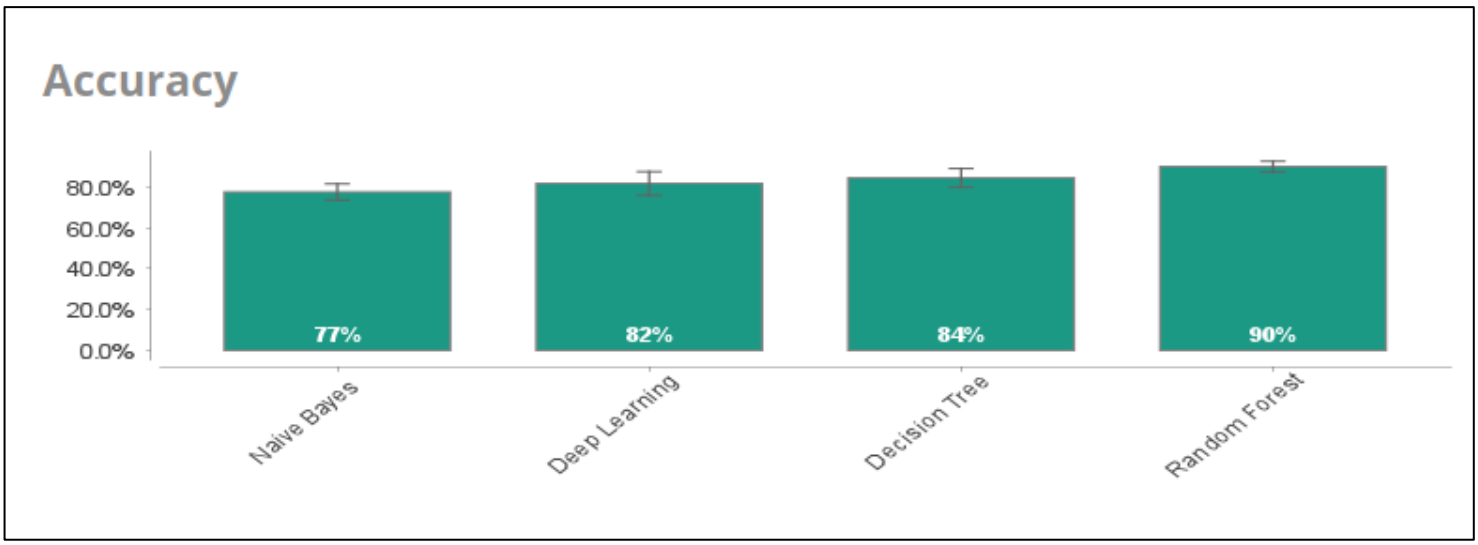
	true 1	true 0	class precision
pred. 1	221	28	88.76%
pred. 0	46	466	91.02%
class recall	82.77%	94.33%	

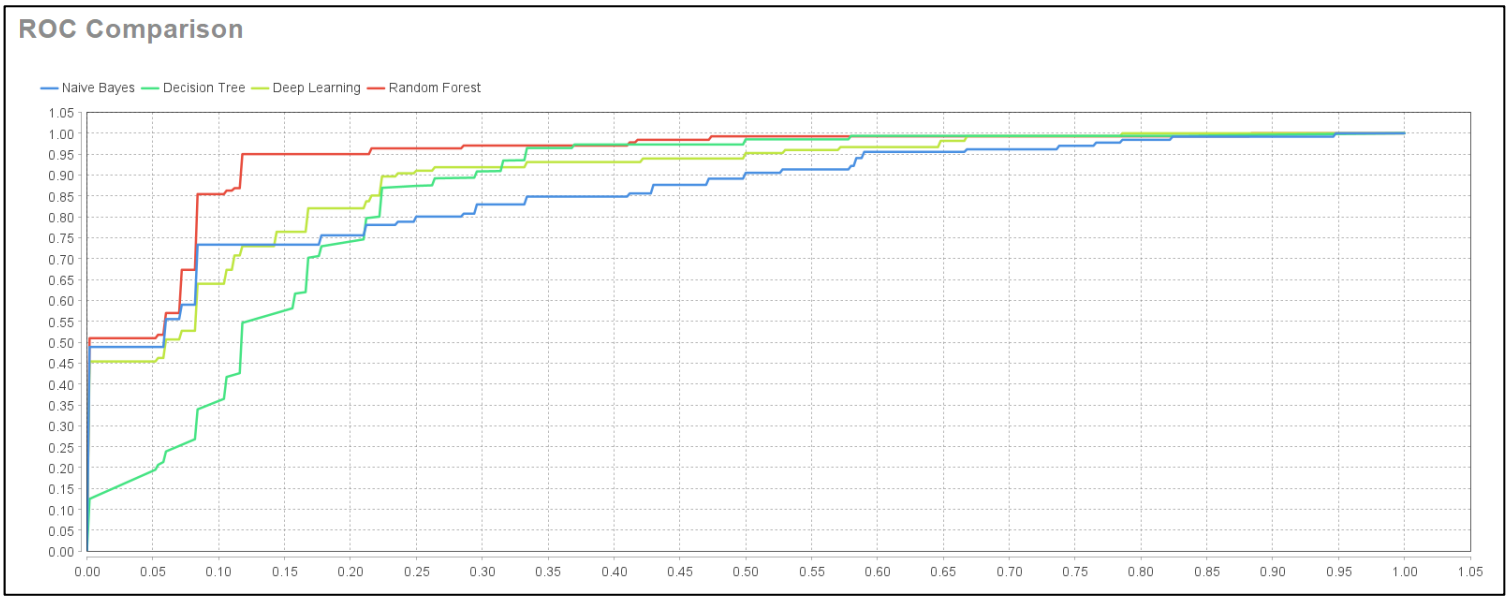
Sonuç:

Sonuç olarak 3 algoritmaya baktığımda -bu veri seti için- en başarılı olanlar sırasıyla k-NN, Sinir Ağları oldu. Karar ağacı da yeterince verimliydi fakat diğerlerine kıyasla daha az başarılı oldu.

AutoModel:

Kullandığım yazılımın bir özelliği olarak veri setini, tahmin etmek istediğim özelliği ve kullanmak/kıyaslamak istediğim yöntemleri seçtiğimde yazılım otomatik olarak algoritmaların optimum ayarlarını yaparak bana sonuçların bir çıktısını döndürüyor. Bu özelliği kullanarak Naive Bayes, Deep Learning, Decision Tree ve Random Forest algoritmalarını kıyaslattım ve aşağıdaki sonuçları elde ettim.





Bu iki grafiğe baktığımda Random Forest algoritmasının diğerlerine göre çok daha başarılı tahminler yaptığını görebiliyorum. Ayrıca Decision Tree'nin Acc. oranına baktığımda tekrar %84 çıktığını görüyorum ki bu en başta kendi uyguladığımda doğru sonuç elde ettiğimi gösteriyor.