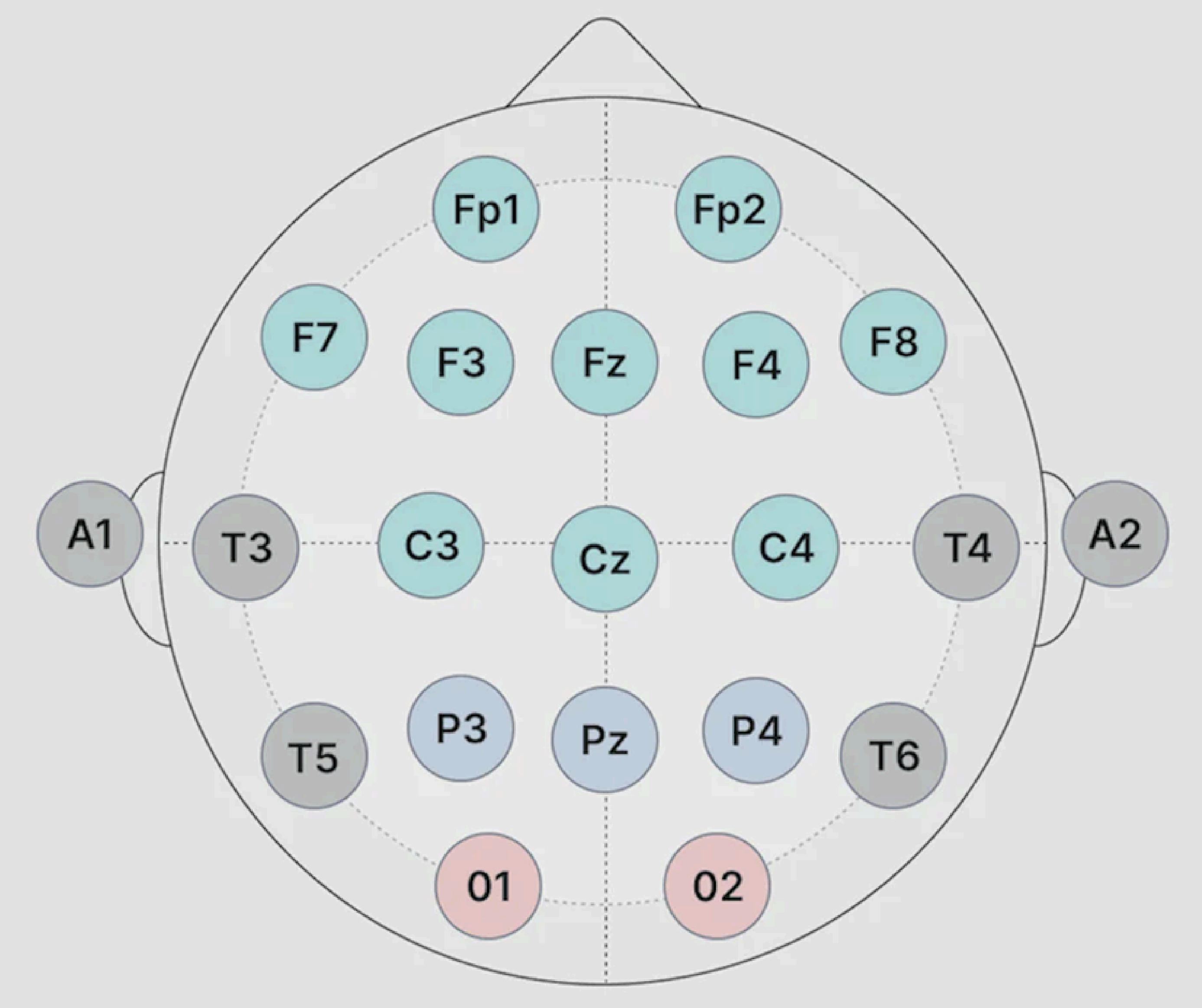
EEG Sinyalleri ile Psikolojik Rahatsızlık Analizi

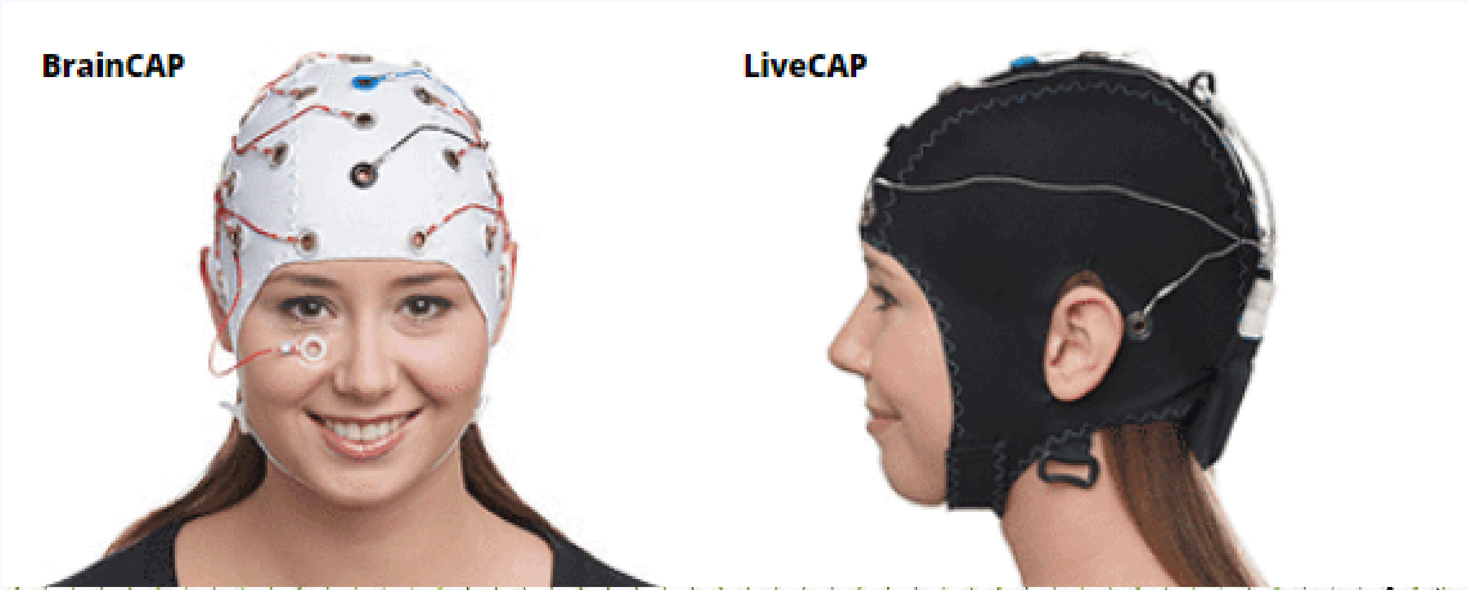
Beyda Bucak 1211602099

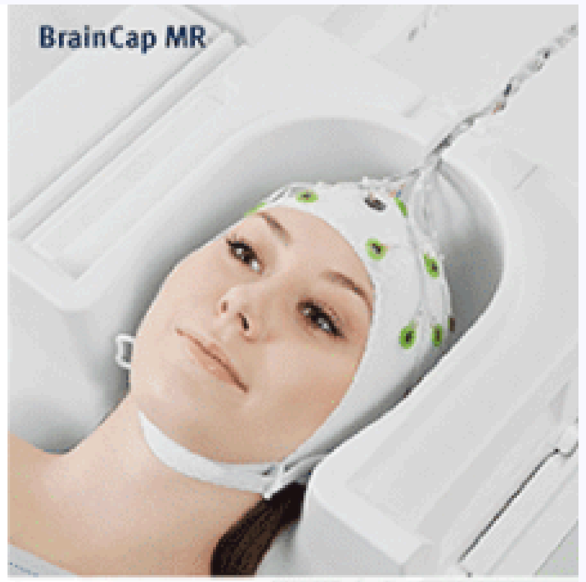
İçindekiler

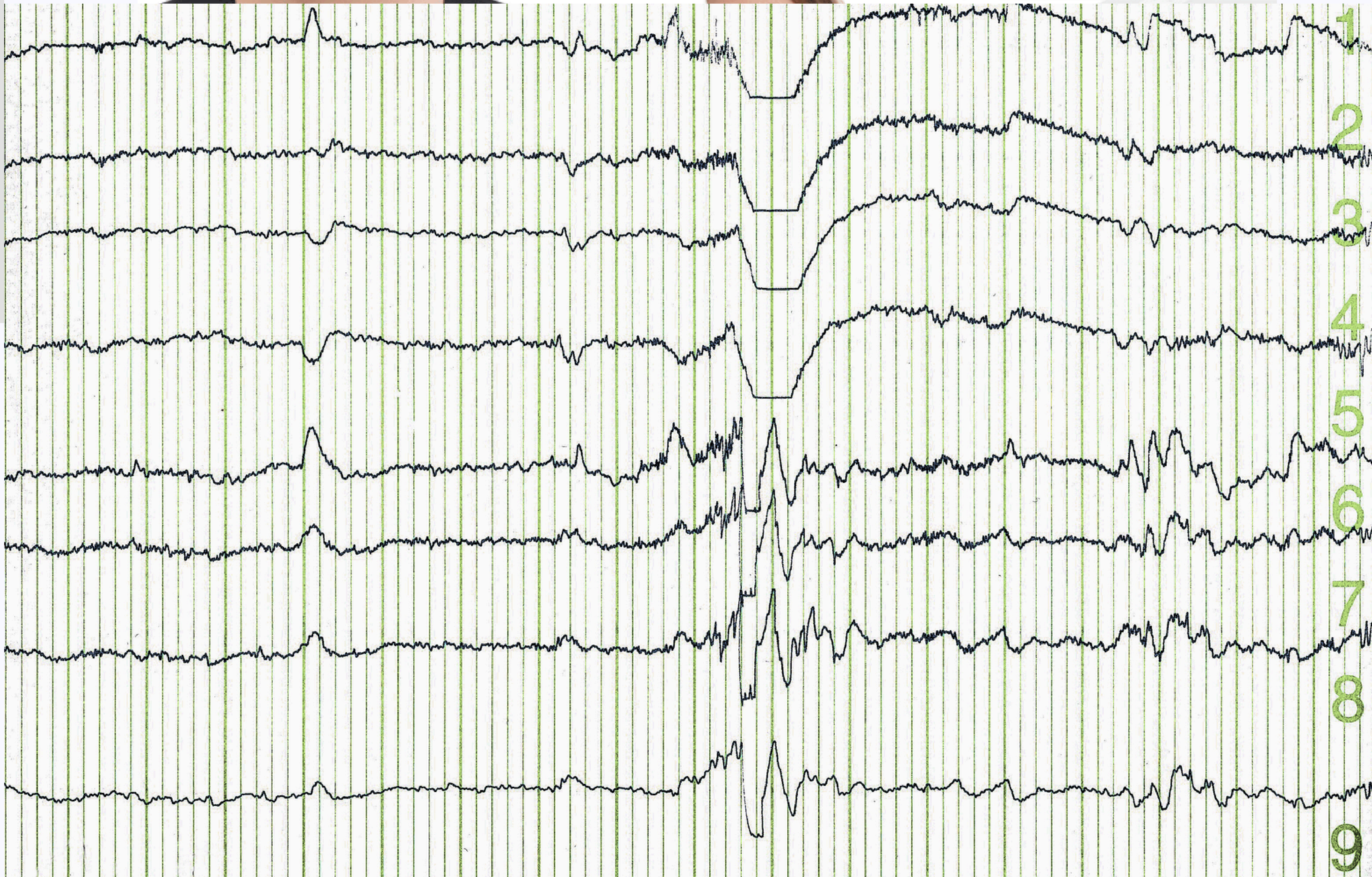
EEG

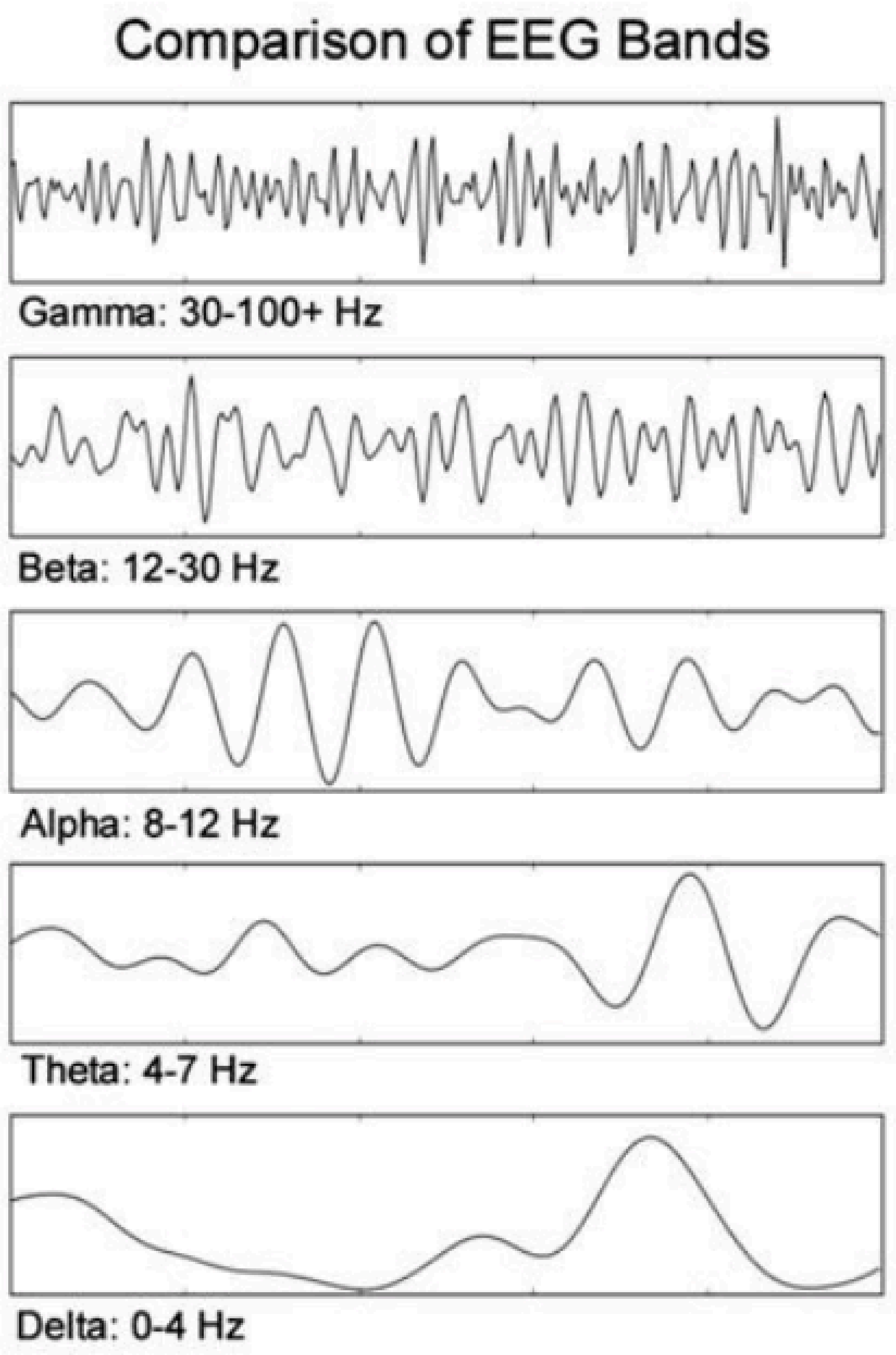
EEG (Elektroensefalografi), beynin elektriksel aktivitelerini ölçmek için kullanılan bir nörolojik görüntüleme yöntemidir. Bu teknik, farklı zihinsel durumları analiz etmek ve beyin fonksiyonlarını değerlendirmek amacıyla kafa derisine yerleştirilen elektrotlar aracılığıyla beyin dalgalarını kaydeder.


Fp1
Fp2
F7
F3
Fz
F4
F8
A1
T3
C3
Cz
C4
T4
A2
T5
P3
Pz
P4
T6
01
02


BrainCAP
LiveCAP


BrainCap MR


1
2
3
4
5
6
7
8
9



yüksek bilinçsel işlevler

karar verme, odaklanma

sakinlik, meditasyon

kapalı göz, dinlenme (yüzeysel gevşeme)

derin anestezi, bilinçsizlik (derin gevşeme)

Depresyon

Kişilik Bozuklukları

Anksiyete

Şizofreni

Yeme Bozuklukları

Bağımlılık

Proje Amacı

- Hangi dalgalar, nereden elde edilmiş?
- Elde edilen dalgalar bize neyi göstermekte?
- Psikolojik rahatsızlıklarla aralarında bir bağlantı var mı?

Variables | Data | Analyses | Edit

Paste | Clipboard | Edit | Setup | Compute | Transform | Weights | Add | Delete | Variables | Filters | Add | Delete | Rows

	no.	sex	age	eeg.date	education	IQ	main.dis...	specific....	AB.A.delt...	AB.A.delt...	AB.A.delt...	AB.A.delt...	AB.A.delt...	AB.A.delt...
1	1	M	57.00	2012.8.30			Addictive dis...	Alcohol use ...	35.999	21.717	21.518	26.825	26.612	25.733
2	2	M	37.00	2012.9.6	6	120	Addictive dis...	Alcohol use ...	13.425	11.003	11.943	15.272	14.152	12.456
3	3	M	32.00	2012.9.10	16	113	Addictive dis...	Alcohol use ...	29.942	27.545	17.150	23.609	27.088	13.541
4	4	M	35.00	2012.10.8	18	126	Addictive dis...	Alcohol use ...	21.496	21.847	17.364	13.834	14.101	13.101
5	5	M	36.00	2012.10.18	16	112	Addictive dis...	Alcohol use ...	37.776	33.608	21.866	21.771	22.855	21.456
6	6	F	24.00	2012.11.21	14	105	Addictive dis...	Alcohol use ...	13.482	14.096	12.855	11.727	13.129	11.627
7	7	F	26.00	2012.12.3	16	103	Addictive dis...	Alcohol use ...	21.781	26.655	17.688	28.341	34.357	30.450
8	8	M	23.00	2013.1.17	12	104	Addictive dis...	Alcohol use ...	11.704	10.600	9.208	16.624	16.483	14.706
9	9	M	24.00	2013.2.15	17	89	Addictive dis...	Alcohol use ...	8.532	8.668	6.937	7.355	9.104	7.530
10	10	F	30.00	2013.2.8	12	98	Addictive dis...	Alcohol use ...	14.854	12.614	21.729	19.998	17.082	28.024
11	11	M	34.00	2013.2.18	13	91	Addictive dis...	Alcohol use ...	37.149	43.506	23.915	40.980	49.382	49.083
12	12	M	32.00	2013.3.8	12	93	Addictive dis...	Alcohol use ...	26.104	26.800	24.656	22.653	27.484	22.751
13	13	M	20.00	2013.2.20	13	105	Addictive dis...	Alcohol use ...	30.399	32.379	19.366	24.372	24.735	24.702
14	14	M	28.00	2013.4.15	14	118	Addictive dis...	Alcohol use ...	15.840	29.317	12.466	8.301	10.526	10.018
15	15	M	23.00	2013.4.11	13	116	Addictive dis...	Alcohol use ...	16.085	15.888	20.982	26.458	19.353	20.170
16	16	M	20.00	2013.4.25	12	93	Addictive dis...	Alcohol use ...	16.620	14.605	17.133	25.394	24.444	23.575
17	17	M	26.00	2013.8.29	12	107	Addictive dis...	Alcohol use ...	16.338	17.716	12.433	12.625	14.075	11.786
18	18	M	30.00	2013.9.27		86	Addictive dis...	Alcohol use ...	12.443	12.504	12.016	12.328	12.045	11.446
19	19	M	31.00	2013.10.18	18	102	Addictive dis...	Alcohol use ...	14.806	14.013	12.508	18.161	12.947	10.905
20	20	M	25.00	2013.11.20	16	120	Addictive dis...	Alcohol use ...	34.557	27.539	17.655	20.828	21.968	16.950
21	21	M	39.00	2013.12.27	12	128	Addictive dis...	Alcohol use ...	18.595	17.021	13.520	21.328	22.884	19.776
22	22	M	20.00	2014.10.23		116	Addictive dis...	Alcohol use ...	28.287	22.412	27.143	18.875	21.998	23.883
23	23	M	30.00	2014.7.7	16	102	Addictive dis...	Alcohol use ...	31.039	22.107	17.404	18.918	22.044	21.096
24	24	M	23.00	2014.7.25	12	104	Addictive dis...	Alcohol use ...	18.802	17.190	12.524	14.841	13.124	14.141
25	25	M	29.00	2014.7.25	13	122	Addictive dis...	Alcohol use ...	28.266	23.889	38.896	21.617	16.127	25.948
26	26	M	32.00	2014.9.19	16	130	Addictive dis...	Alcohol use ...	23.862	25.012	23.325	29.775	30.932	25.793
27	27	M	34.00	2014.9.25	13	122	Addictive dis...	Alcohol use ...	35.108	42.888	40.949	31.440	34.893	31.680
28	28	M	26.00	2014.12.18	17	108	Addictive dis...	Alcohol use ...	16.084	19.081	13.875	19.343	20.263	17.433
29	29	M	25.00	2014.11.24	13	117	Addictive dis...	Alcohol use ...	48.202	30.872	15.091	19.769	22.563	34.516
30	30	M	30.00	2015.11.12	16	122	Addictive dis...	Alcohol use ...	15.989	19.304	11.830	15.917	20.771	18.524
31	31	F	31.00	2017.1.18	16	94	Addictive dis...	Alcohol use ...	7.445	8.637	9.627	8.094	8.225	8.054
32	32	F	19.70	2013.1.17	13	103	Trauma and ...	Acute stress ...	11.907	9.707	12.455	14.209	14.627	19.885
33	33	F	19.94	2013.2.4	13	81	Trauma and ...	Acute stress ...	19.850	15.705	17.863	25.395	23.916	17.033
34	34	F	19.26	2013.8.5	12		Trauma and ...	Acute stress ...	28.676	25.369	25.912	29.497	27.662	26.305

945 adet katılımcı/denek

1149 adet sütun

Jupyter Lab

Jamovi

Python, HTML, CSS

```
[32]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 945 entries, 0 to 944
Columns: 1149 entries, no. to COH.F.gamma.r.01.s.02
dtypes: float64(1144), int64(1), object(4)
memory usage: 8.3+ MB
```

Veri Temizleme

age	eeg.date	education	IQ	main.dis...
57.00	2012.8.30			Addictive dis...
37.00	2012.9.6	6	120	Addictive dis...

```
[55]: df = pd.read_csv("eegdata.csv")
      print(df.head(10))
```

```
   no. sex   age    eeg.date  education     IQ       main.disorder     specific.disorder  AB.A.delta.a.FP
1  AB.A.delta.b.FP2  ...  COH.F.gamma.o.Pz.p.P4  COH.F.gamma.o.Pz.q.T6  COH.F.gamma.o.Pz.r.O1  COH.F.gamm
a.o.Pz.s.O2  COH.F.gamma.p.P4.q.T6  COH.F.gamma.p.P4.r.O1  COH.F.gamma.p.P4.s.O2  COH.F.gamma.q.T6.r.O1
COH.F.gamma.q.T6.s.O2  COH.F.gamma.r.O1.s.O2
0    1   M  57.0   2012.8.30        NaN    NaN  Addictive disorder  Alcohol use disorder         35.99855
7          21.717375   ...          55.989192              16.739679              23.452271
45.678820              30.167520              16.918761              48.850427              9.422630
34.507082              28.613029
1    2   M  37.0    2012.9.6        6.0  120.0  Addictive disorder  Alcohol use disorder         13.42511
8          11.002916   ...          45.595619              17.510824              26.777368
28.201062              57.108861              32.375401              60.351749             13.900981
57.831848              43.463261
2    3   M  32.0   2012.9.10       16.0  113.0  Addictive disorder  Alcohol use disorder         29.94178
0          27.544684   ...          99.475453              70.654171              39.131547
69.920996              71.063644              38.534505              69.908764             27.180532
64.803155              31.485799
3    4   M  35.0   2012.10.8       18.0  126.0  Addictive disorder  Alcohol use disorder         21.49622
6          21.846832   ...          59.986561              63.822201              36.478254
47.117006              84.658376              24.724096              50.299349             35.319695
79.822944              41.141873
4    5   M  36.0  2012.10.18       16.0  112.0  Addictive disorder  Alcohol use disorder         37.77566
7          33.607679   ...          61.462720              59.166097              51.465531
58.635415              80.685608              62.138436              75.888749             61.003944
87.455509              70.531662
```

Veri Temizleme

```
[30]: pd.set_option('display.width', 1000)
      df.head(5)
```

[30]:

	no.	sex	age	eeg.date	education	IQ	main.disorder	specific.disorder	AB.A.delta.a.FP1
0	1	M	57.0	2012.8.30	NaN	NaN	Addictive disorder	Alcohol use disorder	35.998557
1	2	M	37.0	2012.9.6	6.0	120.0	Addictive disorder	Alcohol use disorder	13.425118
2	3	M	32.0	2012.9.10	16.0	113.0	Addictive disorder	Alcohol use disorder	29.941780
3	4	M	35.0	2012.10.8	18.0	126.0	Addictive disorder	Alcohol use disorder	21.496226
4	5	M	36.0	2012.10.18	16.0	112.0	Addictive disorder	Alcohol use disorder	37.775667

5 rows × 1149 columns

```
[36]: print(df.isna().sum())

no.                        0
sex                        0
age                        0
eeg.date                   0
education                 15
                          ..
COH.F.gamma.p.P4.r.O1      0
COH.F.gamma.p.P4.s.O2      0
COH.F.gamma.q.T6.r.O1      0
COH.F.gamma.q.T6.s.O2      0
COH.F.gamma.r.O1.s.O2      0
Length: 1149, dtype: int64
```

```
[53]: print(df['IQ'].isna().sum())

13
```

Veri Temizleme

```
[119]: df.dropna(subset=['main.disorder','specific.disorder'],inplace=True)

[121]: df[['main.disorder','specific.disorder']].isna().sum()

[121]: main.disorder        0
       specific.disorder    0
       dtype: int64
```

Veri Temizleme

[112]:
```
print(df['IQ'].mode()[0])
df["IQ"].fillna(df["IQ"].mode()[0],inplace=True)
```

```
99.0
```

[100]:
```
print(df["IQ"].isna().sum())
```

```
0
```

[125]:
```
doldur = df.drop(columns=['IQ', 'education', 'sex', 'main.disorder','specific.disorder'])

for sutun in doldur:
    df[sutun] = df.groupby('main.disorder')[sutun].transform(
        lambda x: x.fillna(x.mode().iloc[0]) if not x.mode().empty else x
    )
```

[129]:
```
print(df[doldur.columns].isna().sum())
```

```
no.                        0
age                        0
eeg.date                   0
AB.A.delta.a.FP1           0
AB.A.delta.b.FP2           0
                          ..
COH.F.gamma.p.P4.r.O1      0
COH.F.gamma.p.P4.s.O2      0
COH.F.gamma.q.T6.r.O1      0
COH.F.gamma.q.T6.s.O2      0
COH.F.gamma.r.O1.s.O2      0
Length: 1144, dtype: int64
```

```
[131]: print(df.isna().sum())
```

```
no.                     0
sex                     0
age                     0
eeg.date                0
education               0
                       ..
COH.F.gamma.p.P4.r.O1   0
COH.F.gamma.p.P4.s.O2   0
COH.F.gamma.q.T6.r.O1   0
COH.F.gamma.q.T6.s.O2   0
COH.F.gamma.r.O1.s.O2   0
Length: 1149, dtype: int64
```

```
[133]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 930 entries, 1 to 944
Columns: 1149 entries, no. to COH.F.gamma.r.O1.s.O2
dtypes: float64(1144), int64(1), object(4)
memory usage: 8.2+ MB
```

age
57.00
37.00
32.00
35.00
36.00
24.00
26.00
23.00
24.00
30.00
34.00
32.00
20.00
28.00
23.00

```
[144]: df['age']=df['age'].astype(int)
```

age
37
32
35
36
24
26
23
24
30
34
32
20
28
23
20

Jamovi üstünden elde edilen bulgular

```python
import pandas as pd
import re
import os

file_path = os.path.expanduser("~/Desktop/EEG/EEG.csv")
df = pd.read_csv(file_path)

waves = ["delta", "theta", "alpha", "beta", "gamma"]
prefixes = ["AB", "COH"]
new_cols = {}

for prefix in prefixes:
    for wave in waves:
        cols = [col for col in df.columns
                if col.startswith(prefix) and re.search(rf'\b{wave}\b', col)]

        if cols:
            new_col_name = f"{prefix}_{wave}"
            new_cols[new_col_name] = df[cols].mean(axis=1)
        else:
            print(f"Uyarı: '{prefix}_{wave}' için sütun bulunamadı.")

before = df.iloc[:, :8]  # İlk 8 sütun
after = df.iloc[:, 8:]   # Geri kalanlar

inserted = pd.DataFrame(new_cols)
df_final = pd.concat([before, inserted, after], axis=1)
output_path = os.path.expanduser("~/Desktop/EEG/EEG_dalgalar.csv")
df_final.to_csv(output_path, index=False)
```

AB_delta	AB_theta	AB_alpha	AB_beta	AB_gam...	COH_delta	COH_theta	COH_alp...	COH_beta	COH_ga...
12.051	8.450	32.974	26.276	1.250	33.365	37.023	43.408	26.376	26.761
			8.848	1.446	28.263	30.404	30.869	25.512	58.296
			5.810	1.392	18.125	31.578	51.616	33.490	35.518
			17.775	1.714	25.833	37.786	47.248	42.235	69.036
			11.034	2.760	70.139	64.871	56.636	53.758	75.054
			13.836	3.671	21.528	30.088	32.107	38.602	35.437
			8.338	2.066	47.366	51.361	39.369	37.701	55.436
			11.929	1.695	28.757	34.152	41.713	31.835	29.551
			53.853	7.434	54.230	51.317	53.656	40.264	38.837
			24.720	1.724	35.739	36.658	44.839	31.111	30.545
			7.987	2.338	44.070	44.852	48.171	38.474	58.142
			10.310	1.339	37.677	40.756	51.852	51.999	64.393
			13.070	1.424	35.194	31.061	30.499	32.537	39.363
			15.123	1.395	43.226	70.342	65.947	69.808	84.151
			8.795	1.739	39.808	45.007	57.940	46.454	57.333
			17.887	4.980	45.158	49.949	55.541	49.946	53.090
			9.354	2.146	49.361	55.662	66.818	49.625	55.869
			20.634	7.129	47.554	40.111	39.657	32.274	28.369
			6.949	3.226	34.739	34.599	35.901	31.058	39.733
			10.002	1.163	51.153	44.890	34.022	45.916	58.588
			10.839	0.752	73.738	76.595	69.011	67.430	85.225
			30.810	3.463	34.161	38.404	42.903	34.942	59.196
			23.536	1.259	42.275	40.987	40.324	34.994	29.722
			11.808	1.722	41.765	50.039	57.443	51.780	55.216
			8.658	1.772	38.777	41.251	40.980	36.129	58.601
			9.560	1.466	68.767	63.772	78.836	65.397	71.182
			23.728	0.720	29.005	29.585	33.772	24.427	37.926
			8.077	2.494	20.452	27.066	36.523	26.618	21.652
			36.002	2.852	42.035	48.230	37.428	42.479	52.812
			12.770	1.672	61.038	54.358	49.661	52.236	57.942
			12.162	2.934	23.326	39.145	55.522	27.233	22.649
			13.300	0.608	37.870	47.639	35.165	36.924	33.577
			16.826	1.457	36.236	30.281	29.023	30.046	32.659
			37.439	6.047	82.203	78.177	75.987	66.993	88.875
			33.501	0.568	38.978	38.813	34.568	37.867	84.998

AB_delta	AB_theta	AB_alpha	AB_beta	AB_gam...	COH_delta	COH_theta	COH_alp...	COH_beta	COH_ga...
12.051	8.450	32.974	26.276	1.250	33.365	37.023	43.408	26.376	26.761
17.939	13.876	22.908	8.848	1.446	28.263	30.404	30.869	25.512	58.296
12.842	6.826	25.845	5.810	1.392	18.125	31.578	51.616	33.490	35.518
19.612	7.844	27.734	17.775	1.714	25.833	37.786	47.248	42.235	69.036
10.030	5.692	18.272	11.034	2.760	70.139	64.871	56.636	53.758	75.054
22.114	24.216	66.314	13.836	3.671	21.528	30.088	32.107	38.602	35.437
11.957	7.110	47.403	8.338	2.066	47.366	51.361	39.369	37.701	55.436
6.623	5.807	35.580	11.929	1.695	28.757	34.152	41.713	31.835	29.551
19.677	13.125	29.974	53.853	7.434	54.230	51.317	53.656	40.264	38.837
31.335	18.871	21.142	24.720	1.724	35.739	36.658	44.839	31.111	30.545
21.590	11.898	23.333	7.987	2.338	44.070	44.852	48.171	38.474	58.142
19.085	16.495	36.700	10.310	1.339	37.677	40.756	51.852	51.999	64.393
8.165	5.477	61.600	13.070	1.424	35.194	31.061	30.499	32.537	39.363
16.982	10.732	19.388	15.123	1.395	43.226	70.342	65.947	69.808	84.151
17.946	14.989	7.656	8.795	1.739	39.808	45.007	57.940	46.454	57.333
9.961	13.431	64.219	17.887	4.980	45.158	49.949	55.541	49.946	53.090
9.937	5.009	28.397	9.354	2.146	49.361	55.662	66.818	49.625	55.869
16.290	9.336	34.445	20.634	7.129	47.554	40.111	39.657	32.274	28.369
12.991	6.424	4.977	6.949	3.226	34.739	34.599	35.901	31.058	39.733
18.990	19.406	51.376	10.002	1.163	51.153	44.890	34.022	45.916	58.588
14.292	21.112	45.952	10.839	0.752	73.738	76.595	69.011	67.430	85.225
18.003	19.243	130.036	30.810	3.463	34.161	38.404	42.903	34.942	59.196
23.661	22.174	51.678	23.536	1.259	42.275	40.987	40.324	34.994	29.722
25.976	17.569	69.415	11.808	1.722	41.765	50.039	57.443	51.780	55.216

Elektrodlardan elde edilen dalgalar için yeni değişkenler oluşturuldu.

Sağlık analizi için is_healthy sütunu

```
if "is_healthy" in df_final.columns:
    df_final = df_final.drop(columns=["is_healthy"])

df_final.insert(4, "is_healthy", df_final["main.disorder"].apply(
    lambda x: 1 if str(x).strip().lower() == "Healthy control" else 0
))
```

Elektrodlarda gözlemlenen delta ve theta dalgalarıyla bireylerin sağlık durumları arasında **anlamlı** bir farklılık vardır.

Independent Samples T-Test

Independent Samples T-Test

		Statistic	p
AB_alpha	Mann-Whitney U	10486	0.348
AB_beta	Mann-Whitney U	9420	0.084
AB_gamma	Mann-Whitney U	11274	0.723
COH_delta	Mann-Whitney U	10884	0.521
COH_theta	Mann-Whitney U	10900	0.528
COH_alpha	Mann-Whitney U	9719	0.132
COH_beta	Mann-Whitney U	11496	0.850
COH_gamma	Mann-Whitney U	11312	0.745
AB_delta	Mann-Whitney U	8200	0.009
AB_theta	Mann-Whitney U	8841	0.031

Note. H_a $\mu_2 \neq \mu_1$

Independent Samples T-Test

Independent Samples T-Test

		Statistic	p
AB.delta	Mann-Whitney U	8200	0.009
AB.theta	Mann-Whitney U	8841	0.031
AB.alpha	Mann-Whitney U	10486	0.348
AB.beta	Mann-Whitney U	9420	0.084
AB.highbeta	Mann-Whitney U	11634	0.930
AB.gamma	Mann-Whitney U	11274	0.723

Note. H_a $\mu_{Healthy} \neq \mu_{Unhealthy}$

Group Descriptives

	Group	N	Mean	Median	SD	SE
AB.delta	Healthy	26	12.91	11.88	5.70	1.118
	Unhealthy	904	16.44	14.63	7.79	0.2591
AB.theta	Healthy	26	9.81	8.33	6.17	1.209
	Unhealthy	904	12.12	10.14	8.35	0.2778
AB.alpha	Healthy	26	19.73	13.03	17.68	3.467
	Unhealthy	904	24.75	16.12	25.26	0.8401
AB.beta	Healthy	26	9.85	9.38	3.98	0.780
	Unhealthy	904	12.71	10.62	7.53	0.2503
AB.highbeta	Healthy	26	2.26	1.79	1.59	0.313
	Unhealthy	904	2.28	1.81	1.62	0.0538
AB.gamma	Healthy	26	2.55	2.20	1.94	0.381
	Unhealthy	904	2.89	2.20	2.61	0.0866

Mann-Whitney U testi sonuçlarına göre, sağlıklı ve hastalıklı bireyler arasındaki AB.delta değerlerinde anlamlı bir fark bulunmaktadır (p = 0.009). Hastalıklı bireylerin AB.delta değerleri, sağlıklı bireylere kıyasla daha yüksek bulunmuştur.

Theta dalgalarında da iki grup arasında anlamlı bir fark vardır (p = 0.031). Hastalıklı bireylerde theta dalga aktivitesi daha yüksektir.

Dwass-Steel-Critchlow-Fligner pairwise comparisons

Pairwise comparisons - AB.delta

		W	p
Addictive disorder	Anxiety disorder	2.0831	0.761
Addictive disorder	Healthy control	−1.8303	0.855
Addictive disorder	Mood disorder	3.5518	0.155
Addictive disorder	Obsessive compulsive disorder	−0.0606	1.000
Addictive disorder	Schizophrenia	5.1810	0.005
Addictive disorder	Trauma and stress related disorder	4.6339	0.018
Anxiety disorder	Healthy control	−3.0180	0.333
Anxiety disorder	Mood disorder	0.7222	0.999
Anxiety disorder	Obsessive compulsive disorder	−1.1423	0.984
Anxiety disorder	Schizophrenia	2.5424	0.550
Anxiety disorder	Trauma and stress related disorder	1.9846	0.800
Healthy control	Mood disorder	4.7259	0.015
Healthy control	Obsessive compulsive disorder	1.3213	0.967
Healthy control	Schizophrenia	6.2578	<.001
Healthy control	Trauma and stress related disorder	5.4323	0.002
Mood disorder	Obsessive compulsive disorder	−2.0485	0.776
Mood disorder	Schizophrenia	2.3775	0.629
Mood disorder	Trauma and stress related disorder	1.6378	0.910
Obsessive compulsive disorder	Schizophrenia	3.4021	0.196
Obsessive compulsive disorder	Trauma and stress related disorder	2.9167	0.375
Schizophrenia	Trauma and stress related disorder	−0.5482	1.000

Elde edilen **delta** dalgalarının en çok fark yarattığı hastalıklar: **Şizofreni** ve **travmaya/strese bağlı bozukluklar**

	χ^2	df	p	ε^2
AB.delta	31.1	6	<.001	0.0335

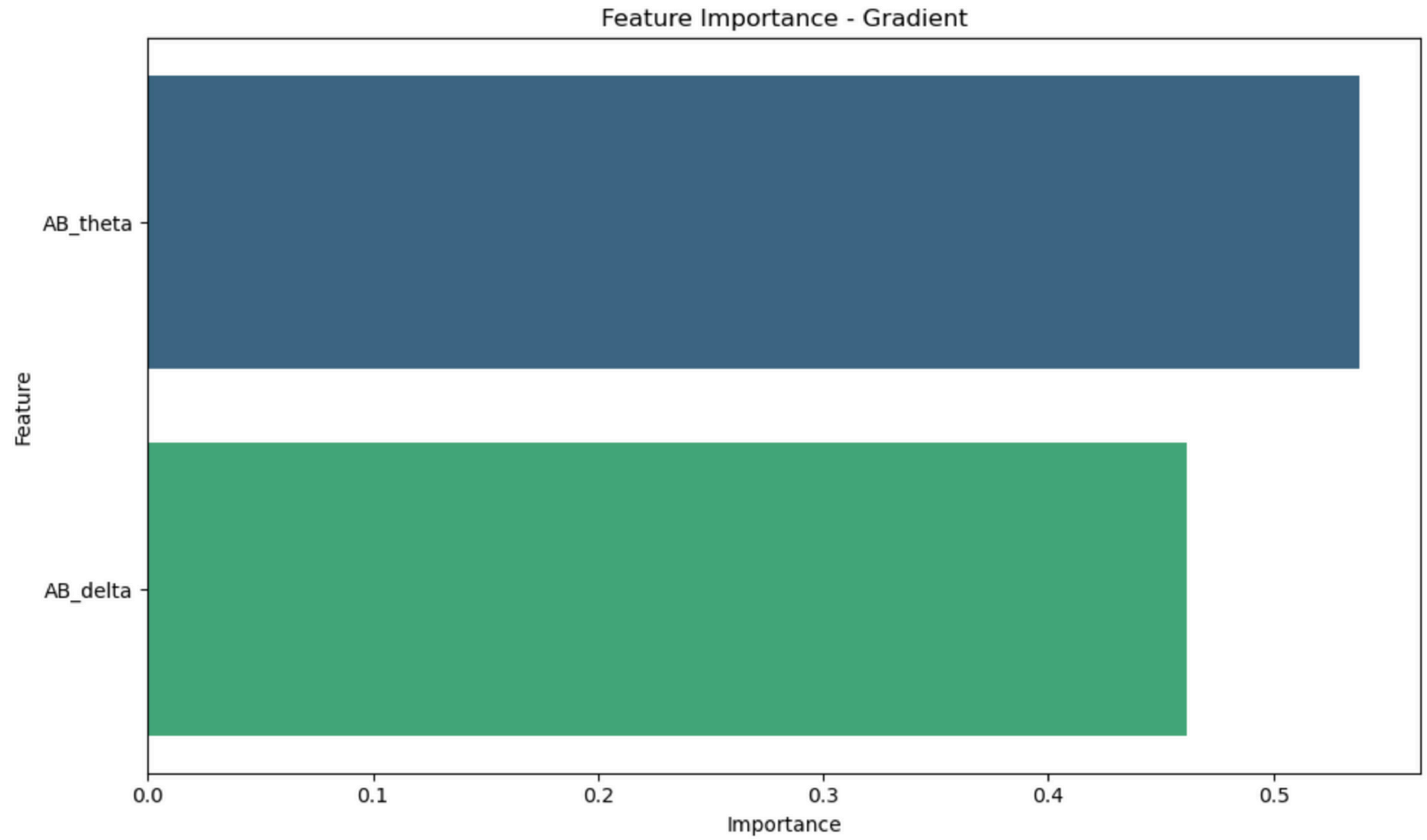
Peki EEG sinyallerinden hastalık tahmini yapabilir miyiz?

```python
file_path = os.path.expanduser("~/Desktop/EEG/EEG_dalgalarr.csv")
df = pd.read_csv(file_path)


keep_cols = [
    "AB_delta", "AB_theta", "AB_alpha", "AB_beta", "AB_gamma",
    "COH_delta", "COH_theta", "COH_alpha", "COH_beta", "COH_gamma", "main.disorder", "specific.disorder", "IQ",
    "education", "is_healthy", "age", "sex"
]

df = df[keep_cols].copy()
output_path = os.path.expanduser("~/Desktop/EEG/EEG_dalgalar.csv")
df.to_csv(output_path, index=False)
```

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 930 entries, 0 to 929
Data columns (total 17 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   AB_delta           930 non-null    float64
 1   AB_theta           930 non-null    float64
 2   AB_alpha           930 non-null    float64
 3   AB_beta            930 non-null    float64
 4   AB_gamma           930 non-null    float64
 5   COH_delta          930 non-null    float64
 6   COH_theta          930 non-null    float64
 7   COH_alpha          930 non-null    float64
 8   COH_beta           930 non-null    float64
 9   COH_gamma          930 non-null    float64
 10  main.disorder      930 non-null    object 
 11  specific.disorder  930 non-null    object 
 12  IQ                 930 non-null    float64
 13  education          930 non-null    float64
 14  is_healthy         930 non-null    int64  
 15  age                930 non-null    int64  
 16  sex                930 non-null    object 
dtypes: float64(12), int64(2), object(3)
memory usage: 123.6+ KB
```

```python
importances = model.feature_importances_
feat_imp_df = pd.DataFrame({
    'Feature': features,
    'Importance': importances
}).sort_values(by='Importance', ascending=False)

plt.figure(figsize=(10,6))
sns.barplot(data=feat_imp_df, x='Importance', y='Feature', palette='viridis')
plt.title("Feature Importance - Gradient")
plt.tight_layout()
plt.show()
```

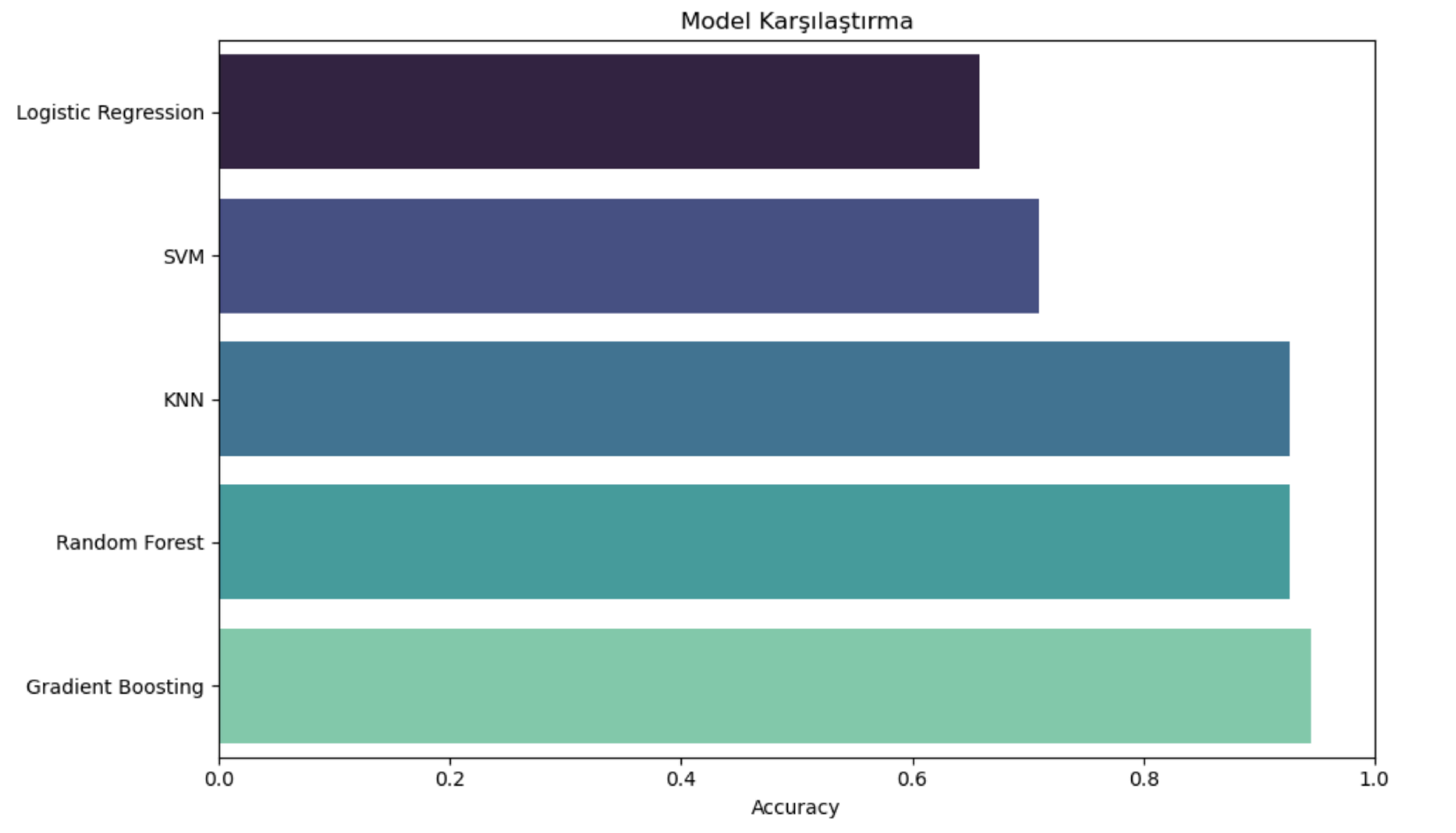


```
features = ['AB_theta', 'AB_delta']
target = 'is_healthy'
df = df[features + [target]].dropna()

healthy = df[df['is_healthy'] == 1]
disorder = df[df['is_healthy'] == 2]
n_disorder = len(disorder)

healthy_augmented = healthy.copy()
for col in features:
    noise = np.random.normal(0, 0.03 * healthy[col].std(), size=len(healthy))
    healthy_augmented[col] += noise

healthy_upsampled = pd.concat([healthy, healthy_augmented]).sample(n=int(n_disorder * 0.5), replace=True, random_state=42)

df_balanced = pd.concat([healthy_upsampled, disorder]).sample(frac=1, random_state=42).reset_index(drop=True)
```

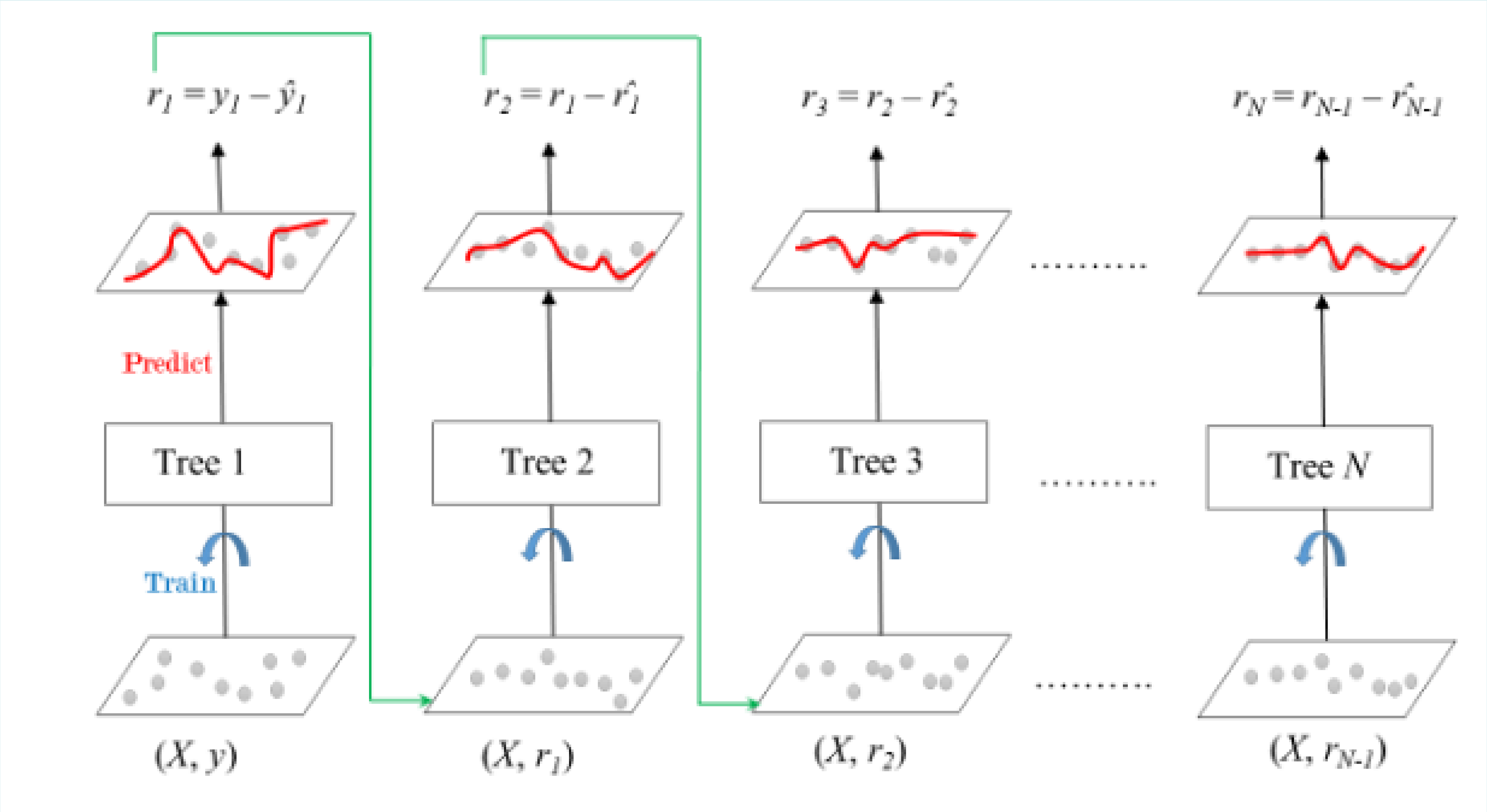
```python
X = df_balanced[features]
y = df_balanced[target]
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.2, random_state=42)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

models = {
    "Logistic Regression": LogisticRegression(max_iter=1000),
    "SVM": SVC(kernel='rbf', probability=True),
    "KNN": KNeighborsClassifier(n_neighbors=5),
    "Random Forest": RandomForestClassifier(n_estimators=100, max_depth=6, min_samples_split=10, min_samples_leaf=5,
                                            class_weight='balanced', random_state=42),
    "Gradient Boosting": GradientBoostingClassifier(n_estimators=100, learning_rate=0.1, max_depth=3, random_state=42)
}

results = []

for name, model in models.items():
    model.fit(X_train_scaled, y_train)
    y_pred = model.predict(X_test_scaled)
    acc = accuracy_score(y_test, y_pred)
    results.append((name, acc))
    print(f"\n {name} Accuracy: {acc:.4f}")
    print(classification_report(y_test, y_pred, target_names=["Healthy", "Disorder"]))

best_model = max(results, key=lambda x: x[1])
print(f"\n En iyi model: {best_model[0]} ({best_model[1]:.4f} doğruluk)")
```

```
 Logistic Regression Accuracy: 0.6581
              precision    recall  f1-score   support

     Healthy       0.00      0.00      0.00        91
    Disorder       0.66      0.99      0.79       181

    accuracy                           0.66       272
   macro avg       0.33      0.49      0.40       272
weighted avg       0.44      0.66      0.53       272


 SVM Accuracy: 0.7096
              precision    recall  f1-score   support

     Healthy       0.68      0.25      0.37        91
    Disorder       0.71      0.94      0.81       181

    accuracy                           0.71       272
   macro avg       0.70      0.60      0.59       272
weighted avg       0.70      0.71      0.66       272


 KNN Accuracy: 0.9265
              precision    recall  f1-score   support

     Healthy       0.82      1.00      0.90        91
    Disorder       1.00      0.89      0.94       181

    accuracy                           0.93       272
   macro avg       0.91      0.94      0.92       272
weighted avg       0.94      0.93      0.93       272


 Random Forest Accuracy: 0.9265
              precision    recall  f1-score   support

     Healthy       0.82      1.00      0.90        91
    Disorder       1.00      0.89      0.94       181

    accuracy                           0.93       272
   macro avg       0.91      0.94      0.92       272
weighted avg       0.94      0.93      0.93       272


 Gradient Boosting Accuracy: 0.9449
              precision    recall  f1-score   support

     Healthy       0.93      0.90      0.92        91
    Disorder       0.95      0.97      0.96       181

    accuracy                           0.94       272
   macro avg       0.94      0.93      0.94       272
weighted avg       0.94      0.94      0.94       272


 En iyi model: Gradient Boosting (0.9449 doğruluk)
```

```
model_names, scores = zip(*results)
plt.figure(figsize=(10,6))
sns.barplot(x=scores, y=model_names, palette='mako')
plt.xlabel("Accuracy")
plt.title("Model Karşılaştırma")
plt.xlim(0, 1)
plt.tight_layout()
plt.show()
```



Gradient Boosting

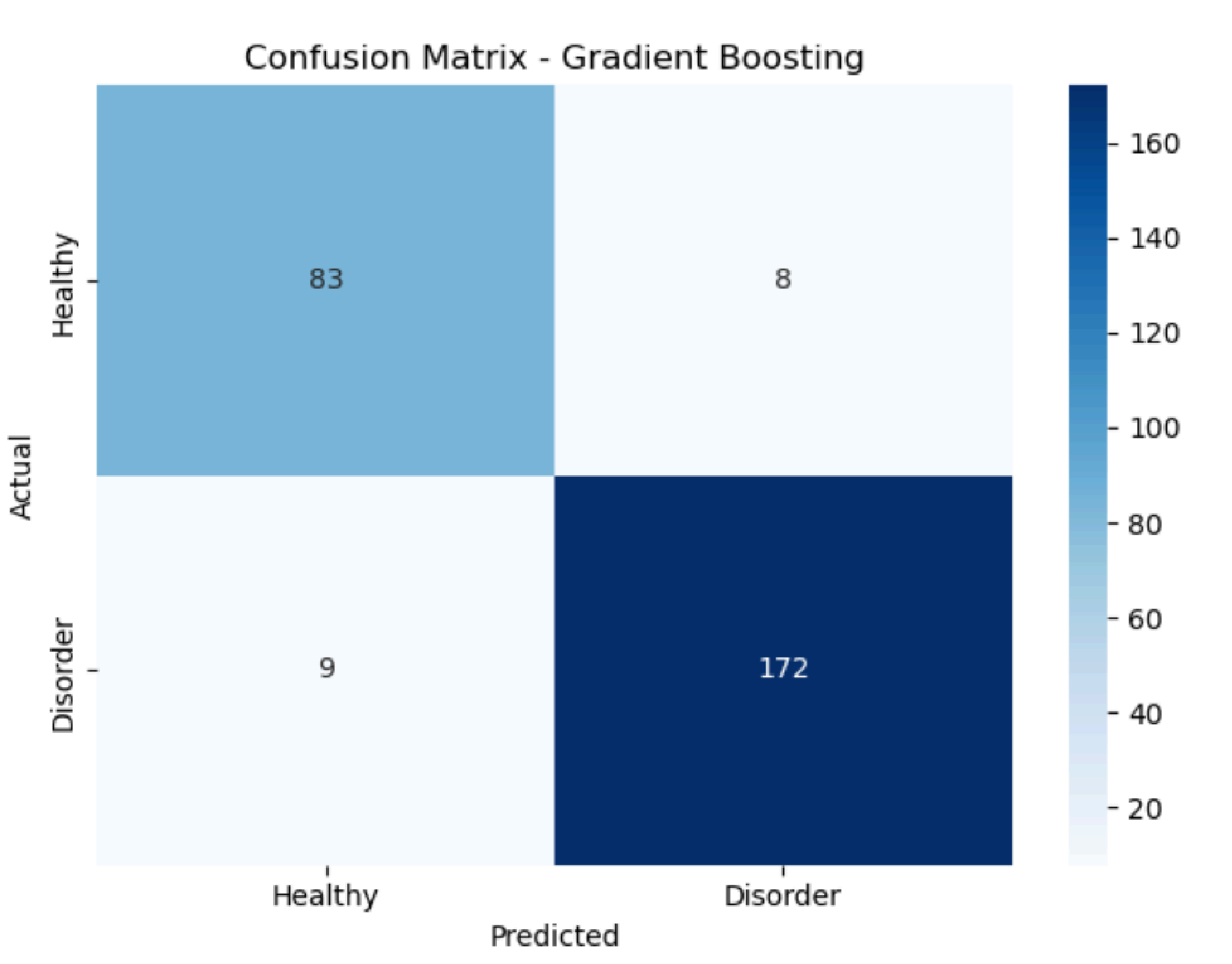


Gradient Boosting, zayıf öğrenicileri (weak learner) bir araya getirerek güçlü bir öğrenici (strong learner) oluşturmak için kullanılan bir makine öğrenme algoritmasıdır.

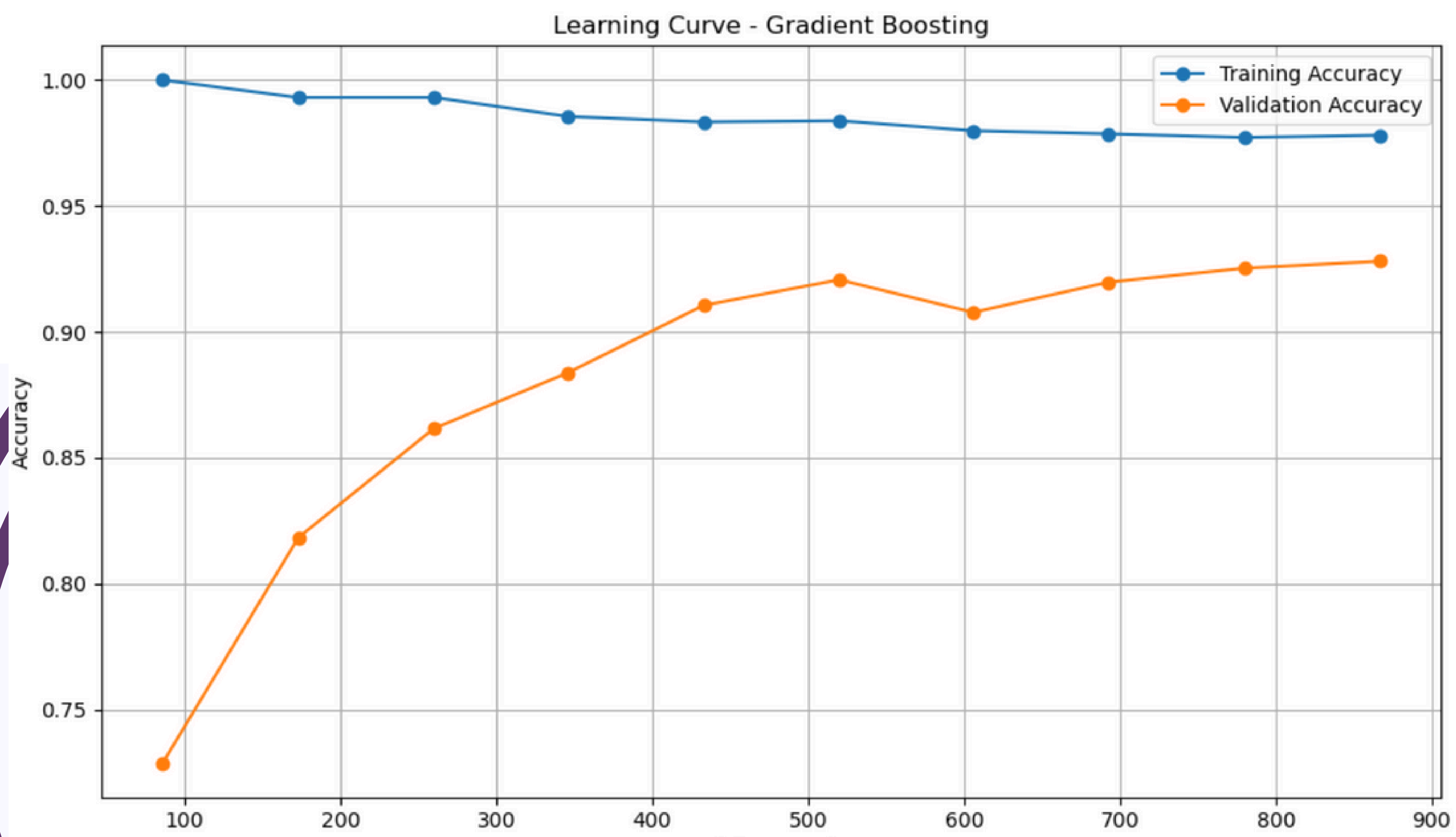
```python
model = GradientBoostingClassifier(n_estimators=100, learning_rate=0.1, max_depth=3, random_state=42)
model.fit(X_train_scaled, y_train)
y_pred = model.predict(X_test_scaled)

acc = accuracy_score(y_test, y_pred)
print(f" Accuracy: {acc:.4f}")
print(classification_report(y_test, y_pred, target_names=["Healthy", "Disorder"]))
```

```
Accuracy: 0.9375
              precision    recall  f1-score   support

     Healthy       0.90      0.91      0.91        91
    Disorder       0.96      0.95      0.95       181

    accuracy                           0.94       272
   macro avg       0.93      0.93      0.93       272
weighted avg       0.94      0.94      0.94       272
```

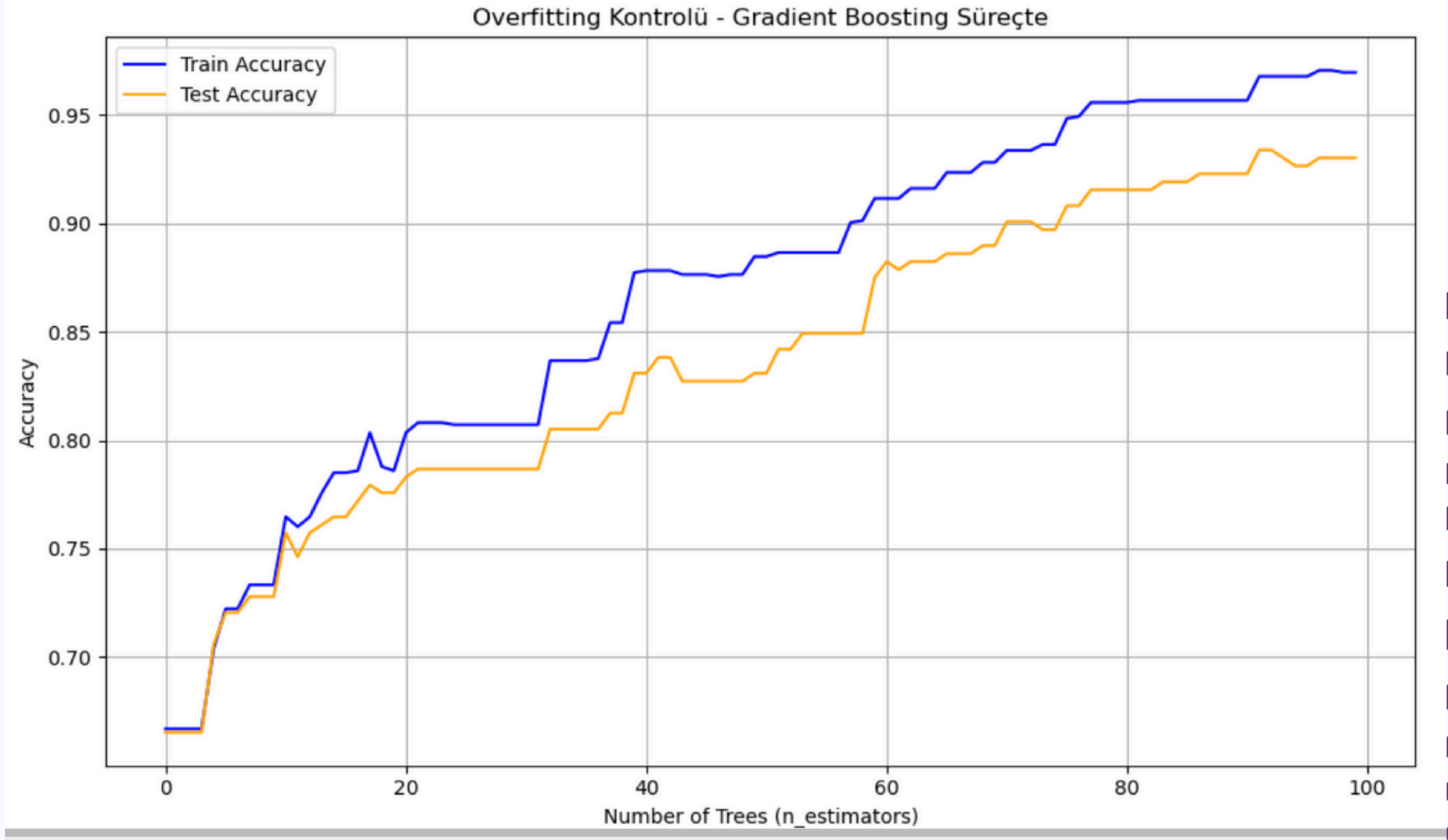
```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Healthy", "Disorder"], yticklabels=["Healthy", "Disorder"])
plt.title("Confusion Matrix - Gradient Boosting")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.tight_layout()
plt.show()
```



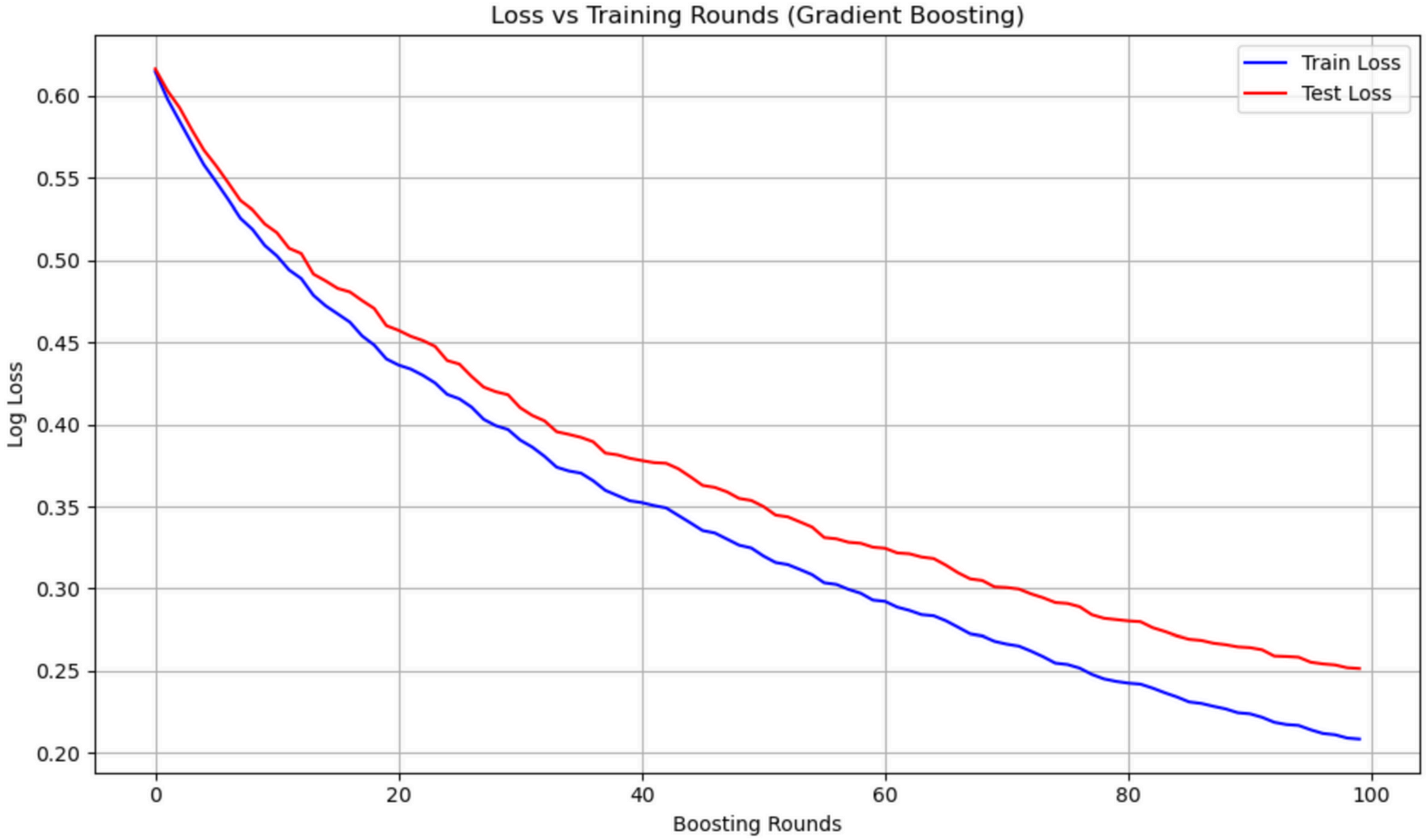
```python
import matplotlib.pyplot as plt
from sklearn.model_selection import learning_curve

train_sizes, train_scores, test_scores = learning_curve(
    model, X_train_scaled, y_train, train_sizes=np.linspace(0.1, 1.0, 10), cv=5, scoring='accuracy', n_jobs=-1
)

train_mean = train_scores.mean(axis=1)
test_mean = test_scores.mean(axis=1)
train_std = train_scores.std(axis=1)
test_std = test_scores.std(axis=1)

plt.figure(figsize=(10, 6))
plt.plot(train_sizes, train_mean, label="Train Accuracy", color='blue')
plt.plot(train_sizes, test_mean, label="Test Accuracy", color='red')

plt.fill_between(train_sizes, train_mean - train_std, train_mean + train_std, color='blue', alpha=0.2)
plt.fill_between(train_sizes, test_mean - test_std, test_mean + test_std, color='red', alpha=0.2)

plt.title('Learning Curve')
plt.xlabel('Training Set Size')
plt.ylabel('Accuracy')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



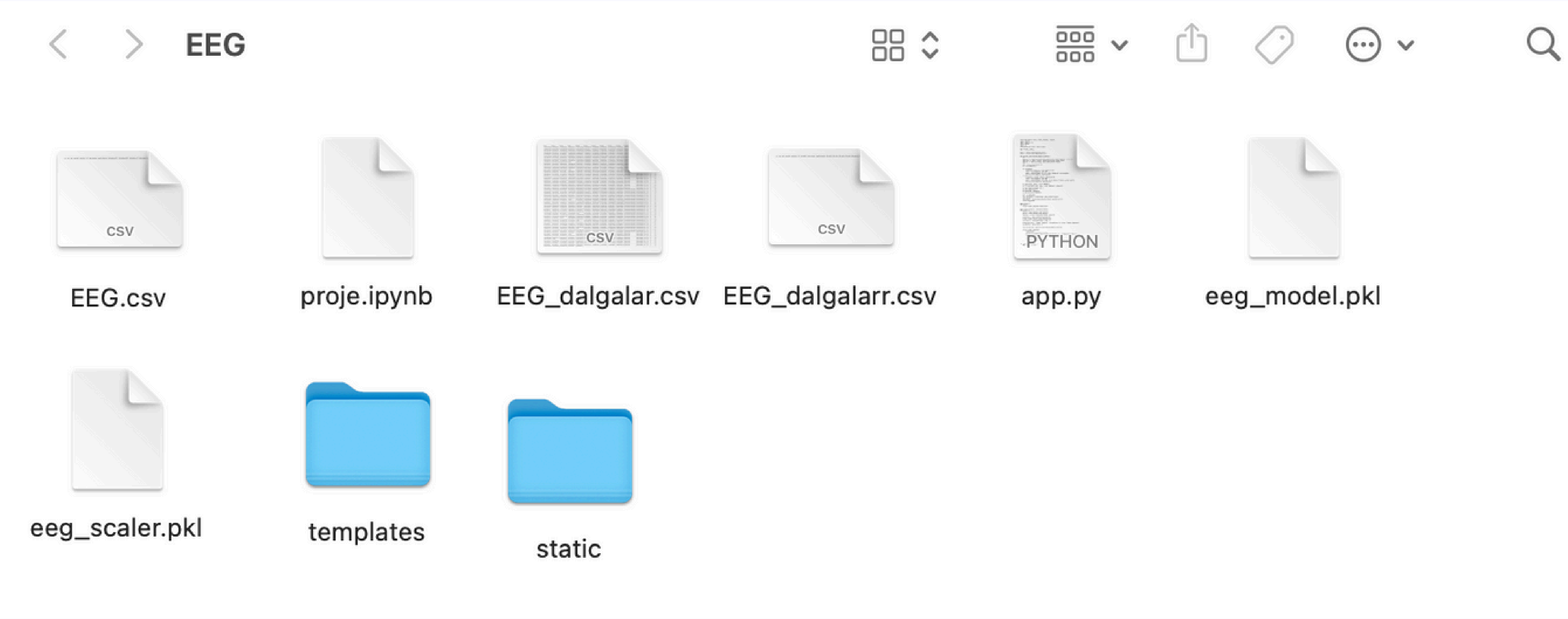
```python
train_acc_list = []
test_acc_list = []

for y_train_pred in model.staged_predict(X_train_scaled):
    train_acc_list.append(accuracy_score(y_train, y_train_pred))

for y_test_pred in model.staged_predict(X_test_scaled):
    test_acc_list.append(accuracy_score(y_test, y_test_pred))

plt.figure(figsize=(10, 6))
plt.plot(train_acc_list, label='Train Accuracy', color='blue')
plt.plot(test_acc_list, label='Test Accuracy', color='orange')
plt.xlabel("Number of Trees (n_estimators)")
plt.ylabel("Accuracy")
plt.title("Overfitting Kontrolü - Gradient Boosting Süreçte")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

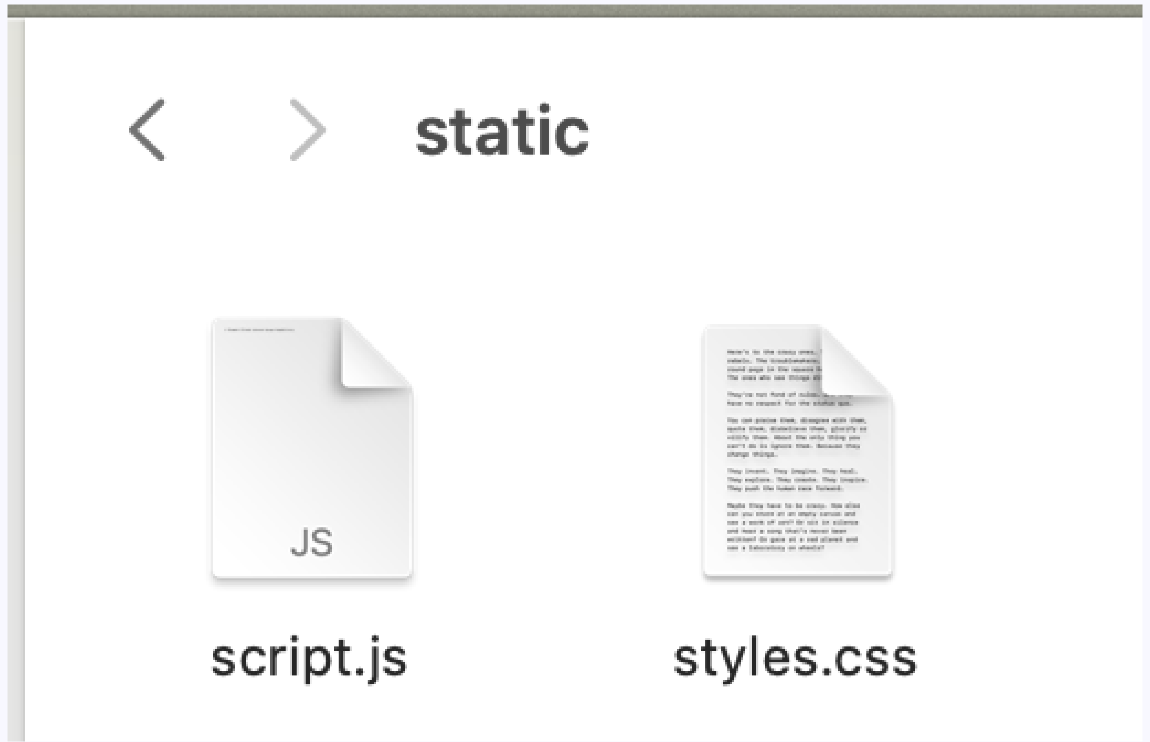


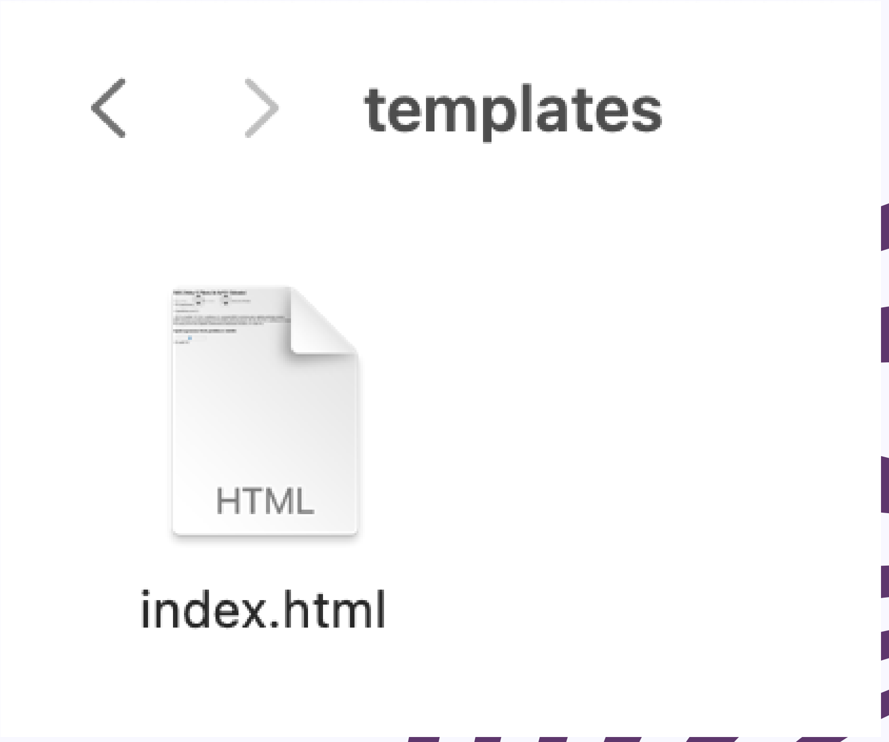
```python
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import log_loss

gb_model = GradientBoostingClassifier(n_estimators=100, learning_rate=0.1, max_depth=3, random_state=42)
gb_model.fit(X_train_scaled, y_train)

train_loss = []
test_loss = []

for y_train_pred_proba, y_test_pred_proba in zip(gb_model.staged_predict_proba(X_train_scaled),
                                                 gb_model.staged_predict_proba(X_test_scaled)):
    train_loss.append(log_loss(y_train, y_train_pred_proba))
    test_loss.append(log_loss(y_test, y_test_pred_proba))

plt.figure(figsize=(10, 6))
plt.plot(train_loss, label="Train Loss", color='blue')
plt.plot(test_loss, label="Test Loss", color='red')
plt.xlabel("Boosting Rounds")
plt.ylabel("Log Loss")
plt.title("Loss vs Training Rounds (Gradient Boosting)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```



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

from sklearn.model_selection import train_test_split, learning_curve
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, log_loss

from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC

from imblearn.over_sampling import SMOTE
from imblearn.under_sampling import RandomUnderSampler

!pip install xgboost
!pip install imbalanced-learn
```


EEG
EEG.csv
proje.ipynb
EEG_dalgalar.csv
EEG_dalgalarr.csv
app.py
eeg_model.pkl
eeg_scaler.pkl
templates
static


static
script.js
styles.css


templates
index.html

EEG Delta & Theta ile Sağlık Tahmini

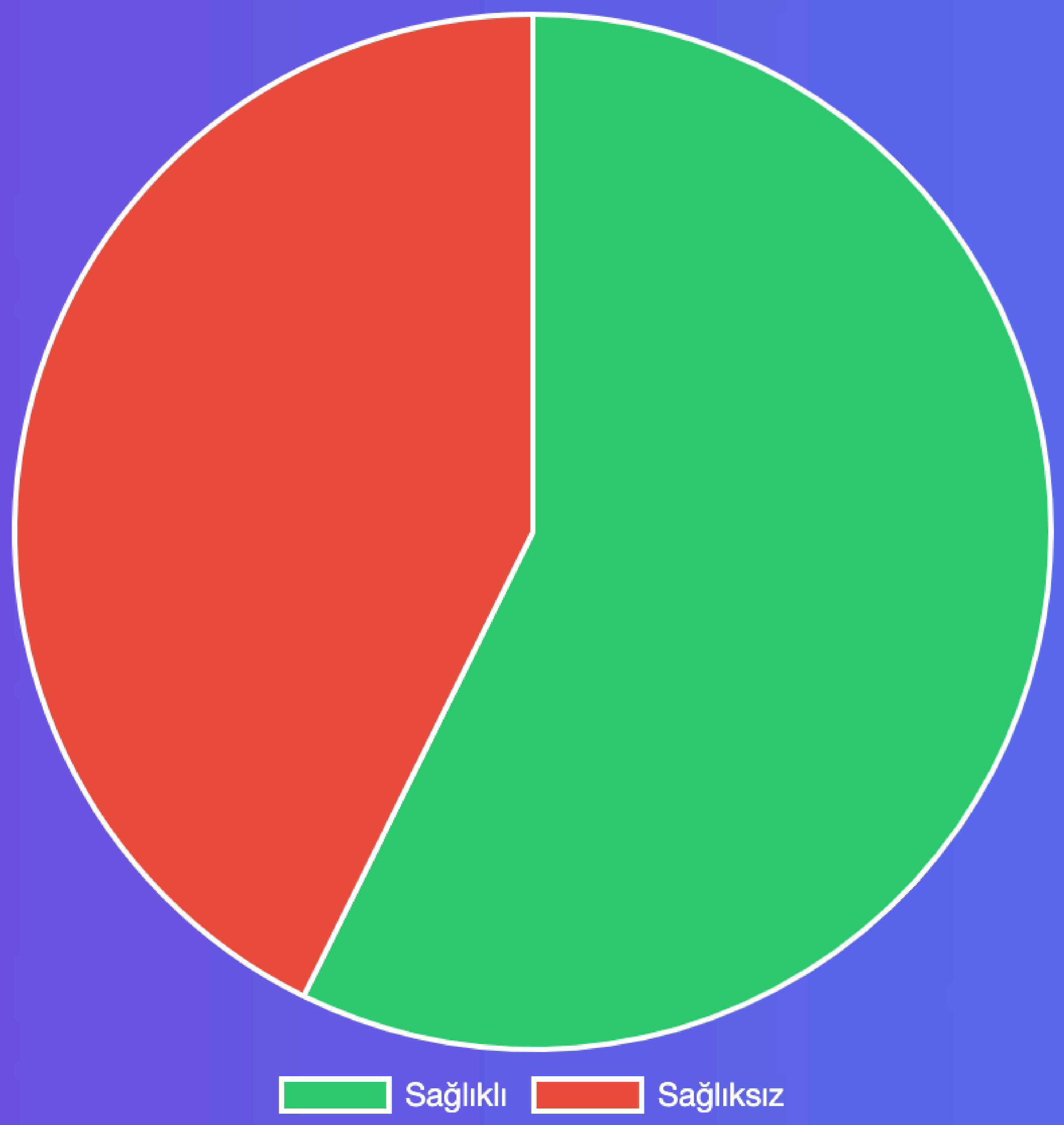
9.8

11.8

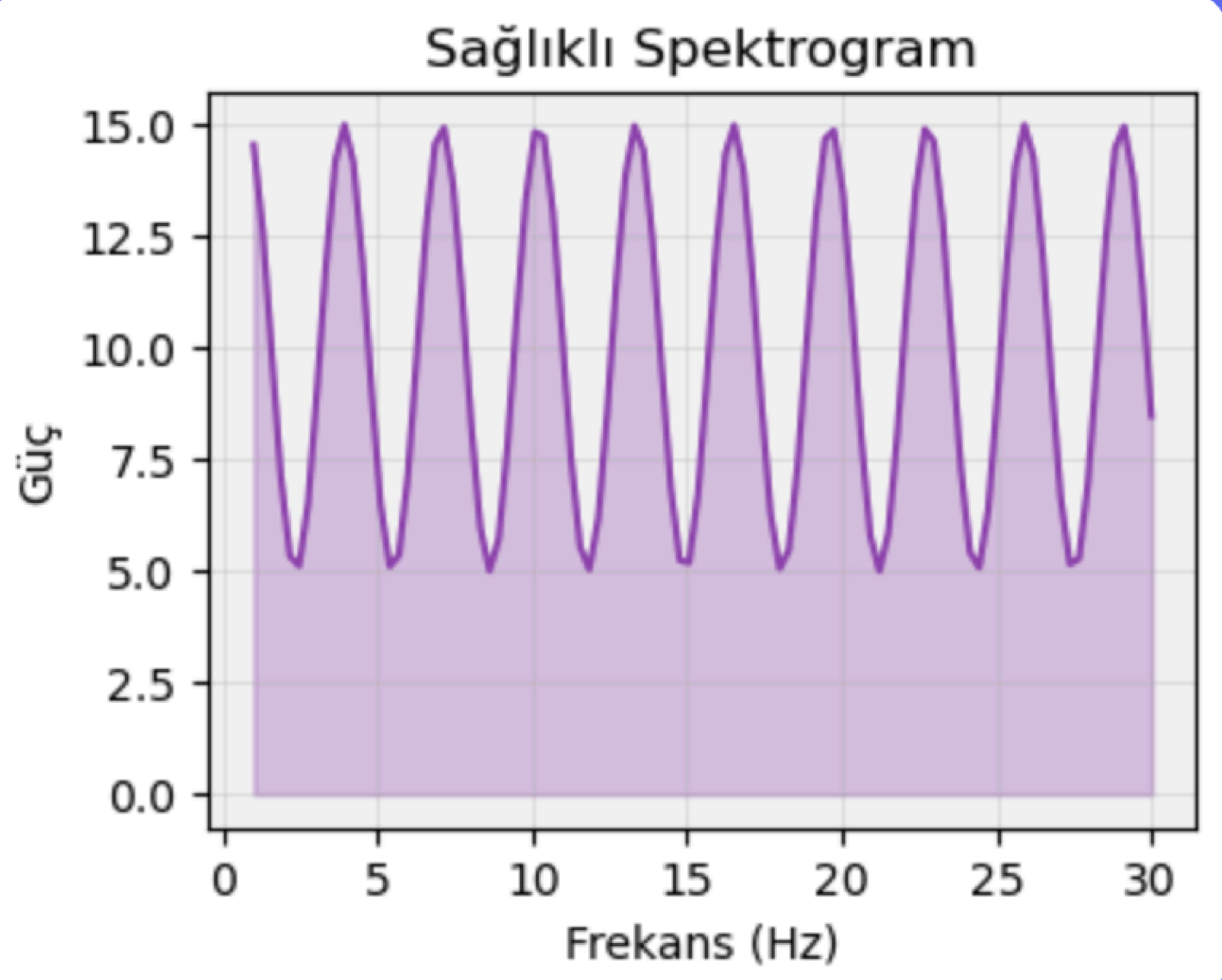
Sonucumu Hesapla

Tahmin: Sağlıklı

%57.29 oranında EEG verilerine göre sağlıklı görünüyorsunuz.
EEG sonuçları açısından belirgin bir bozukluk gözlemlenmedi.



Spektrogramınız böyle gözüküyor olabilir:



Teşekkürler

Kaynak

https://www.youtube.com/watch?v=O1hsYQY-t8E
https://www.kaggle.com/datasets/shashwatwork/eeg-psychiatric-disorders-dataset/data https://osf.io/8bsvr/
https://www.jamovi.org/cloud.html
https://www.w3schools.com/
https://mne.tools/stable/index.html
https://pubmed.ncbi.nlm.nih.gov/33693596/
https://pubmed.ncbi.nlm.nih.gov/31277844/
http://www.edumed.org.br/cursos/neurociencia/MethodsEEGMeasurement.pdf
https://www.kemalarikan.com/en/eeg-in-psychiatry.html