Heart Disease Classification Model Reports

1. Özet

Bu çalışma da Makine Öğrenmesi altında Eğiticili Öğrenme (Supervised Learning) algoritmaları içerisinde yer alan K Nearest Neighbor (KNN), Decision Tree(DT) ve Multi Layer Perceptron(MLP) sınıflandırma algoritmaları ile ikili sınıflandırma (Binary Classification) yapılmıştır. Bu algoritmalar da yer alan hiper parametrelerin optimizasyonu için hız ve performans gerekçesi ile RandomSearch algoritması kullanılmıştır. Model Performans için K Fold Cross Validation algoritması (k=5) ile veri seti 5 fold'a bölünerek (over ve under fitting önüne geçilmesi) performanslar ölçülmüştür. Bağımlı ***output*** değişkenin sınıf dağılımına baktığımızda dengeli sınıflara (0:Hasta Olmama Durumu=138, 1:Hasta Olması Durumu=165) sahip bir veri seti ile modelleme yaptığımızı söyleyebilirim. KNN, DT ve MLP algoritmaları RandomSearch parametre optimizasyonları sağlanarak k=5 k fold cross validation ile modellerin eğitimleri yapılmıştır.

Model Performansları

	Validation Set Perf.	Test Set Perf. (N=61)			
	Accuracy	Accuracy	Precision	Recall	F1 Score
K Nearest Neighbor(KNN)	0.933588	0.93442	0.91176	0.96875	0.93939
Decision Tree(DT)	1	1	1	1	1
Multi Layer Perceptron(MLP)	0.958418	0,98360	0,96969	1	0,98461

2. Giriş

Bu çalışmada 303 adet gözlem ve gözlemlere ait değişkenler ile Kalp Krizi(Heart Attack) geçirme durumunu Makine Öğrenmesi sınıflandırma algoritmalarından Decision Tree, K Nearest Neighbor ve Multi Layer Perceptron ile modellenmesi sağladık. Bağımlı değişkenimiz kategorik olduğu için bu çalışma da İkili Sınıflandırma (Binary Classification) problemi modelledik. Sağlık alanın da yapılan çalışmalar/modellemeler doğrudan insan hayatını etkilediği için diğer gerçek hayat problemlerinin yanında kritik bir öneme sahiptir. Kalp Krizi geçirme durumun da bulunan birine tersi yönde beyanatta bulunulması (False Negative) can kayıplarına sebebiyet vermektedir. Sağlık alanında yer alan problemlerde False Negative minimize edilmesi önem arz etmektedir. Makine Öğrenmesi, veri içerisinde yer alan desenlerin (pattern) algoritmalar vasıtasıyla çıkarılarak (insan beynin kavrayabileceğinden daha fazla karmaşıklığa sahip) veriden (sürekli) öğrenebilen algoritmalardır. İnsan beyni 2 boyuttan sonra genelleme yada örüntüleri çıkarma noktasında işlevselliğini kaybedebiliyor. Bu noktada boyut karmaşıklığı artsada Makine öğrenmesi algoritmaları ile bu tipte problemleri çözebilmekteyiz. Sadece burada bu tip modellerin veriden öğrenme (desenlerin ortaya çıkarılması) yaptığı için modele girdi sağlayan verinin kalitesi (Garbage in, Garbe out) oldukça önemlidir.

3. Veri Kümesi

Heart Disease veri seti 303 gözlemden oluşan 13 adet bağımsız ve 1 adet bağımlı olmak üzere 14 tane değişkenden oluşmaktadır.

Output yani tahmin etmek istediğimiz değişkenin dağılımına göz attığımızda Hasta Olmama Durumu ifade eden 0 değeri 138 tane gözlem ve Hasta Olma Durumu ifade eden 1 değeri 165 tane gözlemden oluşmaktadır.

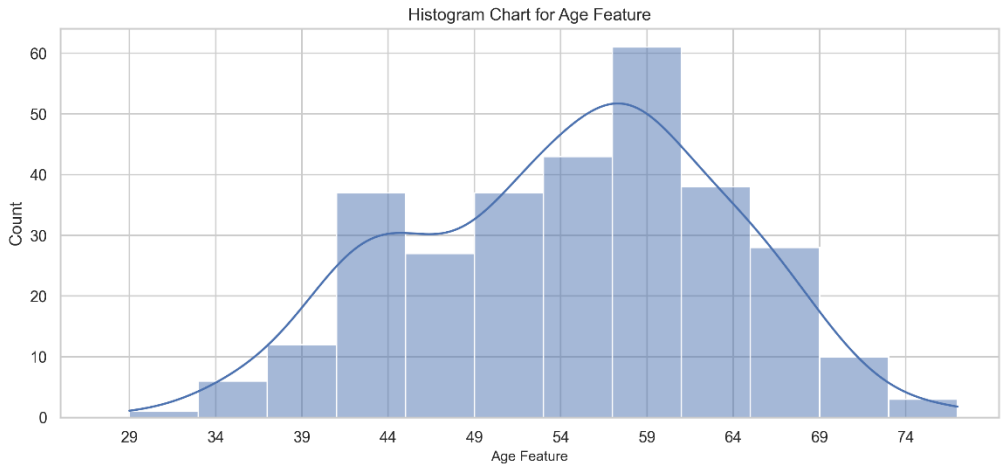
Özellik Sayısı (Features) 13 tane bağımsız ve 1 tane bağımlı(output) olmak üzere 14 tanedir.

Veri seti içerisinde yer alan özelliklerin(değişkenlerin) kullanım amacı ve tipleri ve bazılarının (açıklamalarını) aşağıdaki tabloda görebilirsiniz.

Id	Features	Input/Output	Data Types	Description	Values
1	age	Input (Independent)	int	Hastanın Yaşı (Yıl Cinsinden)	
2	sex	Input (Independent)	cat	Hastanın Cinsiyeti	0 ve 1
3	cp	Input (Independent)	cat	Hastanın Göğüs Ağrısı Tipi	0,1,2 ve 3
4	trtbps	Input (Independent)	int	Hastanın (Dinlenirken) Kan Basıncı Değeri	
5	chol	Input (Independent)	int	Hastanın Kolesterol (BMI sensörü ile) Değeri	
6	fbs	Input (Independent)	cat	Hastanın Açlık Kan Şekeri	0(False) ve 1(True)
7	restecg	Input (Independent)	cat	Hastanın Dinlenme Halindeki Elektrokardiyografik Sonuçları	0,1 ve2
8	thalachh	Input (Independent)	int	Bilgi Elde Edilememiştir!	
9	exng	Input (Independent)	cat	Egzersizin Neden Olduğu Angina	0(No) ve 1(Yes)
10	oldpeak	Input (Independent)	int	Bilgi Elde Edilememiştir!	
11	slp	Input (Independent)	cat	Bilgi Elde Edilememiştir!	0,1 ve 2
12	caa	Input (Independent)	cat	Bilgi Elde Edilememiştir!	0,1,2,3 ve 4
13	thall	Input (Independent)	cat	Bilgi Elde Edilememiştir!	0,1,2 ve 3
14	output	Output (Dependent)	cat	Hastanın Kalp Krizi Geçirme Durumu	0 ve 1

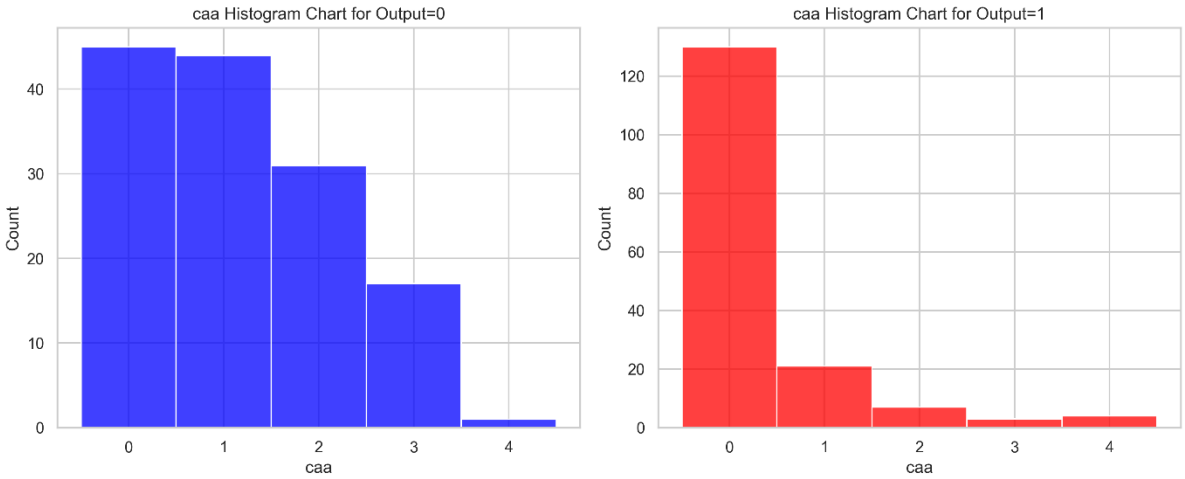
Not: Burada yorumlarıma ek olarak proje dizini içerisinde reports altında yer alan "**HeartDiseaseReports.html**" Data Profiling raporuna göz atmanızı öneririm.

Hastaların Yaş Dağılım Histogramı



Age özelliği için Ortalama değerin 54,36 ve Std. Dev. 9,08 olduğu, Mix ve Max Range genişliğinin 29 ile 77 arasında olduğunu, Median değerinin 55 olduğunu ve histogram üzerinde yoğunluğunun 41 yaş ile 69 yaş arasında bulunduğu ve Normal Dağılımı benzediğini şeklen söyleyebiliriz. Ayrıca, Normal Dağılıma göre veri setinin %95'nin; **μ ± 2 x Std. Dev.** = 54,36 ± 2*9,08 ile 36,2 ile 77,52 arasında bulunduğunu söylebiliriz.

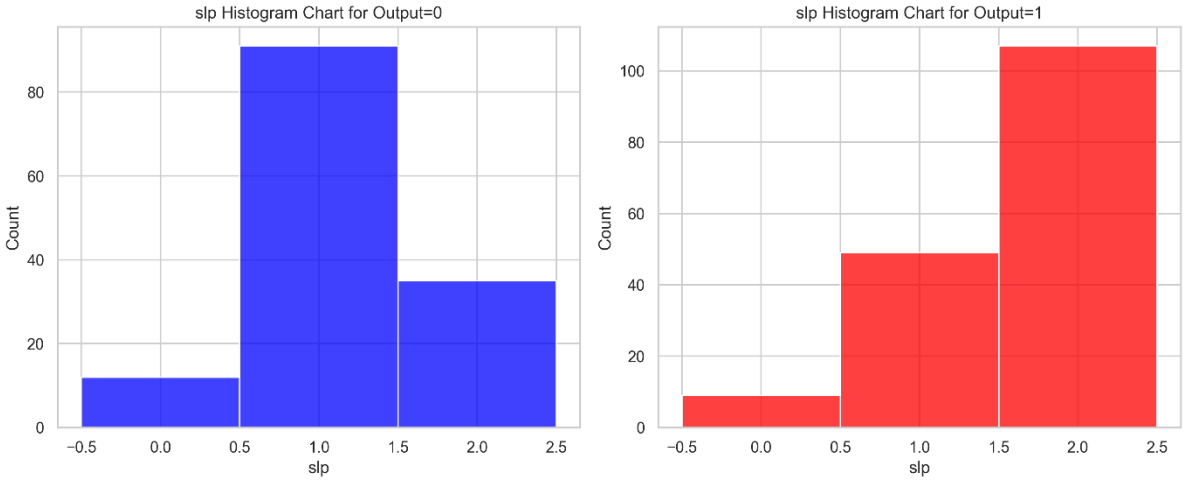
Hasta ve Sağlam Kişiler için Özellik Histogramları



caa Kategorik değişkeni 5 tane distinct değeri bulunmaktadır. Bunlardan özellikle 0 değeri hem output=0 hemde output=1 için dağılımda yoğunluk göstermektedir.

Output=0(Hasta Olmayan) için değerlerin daha çok 0 ve 1 de yer aldığını (nispeten 2 ve 3 de bulunduğunu) söyleyebiliriz.

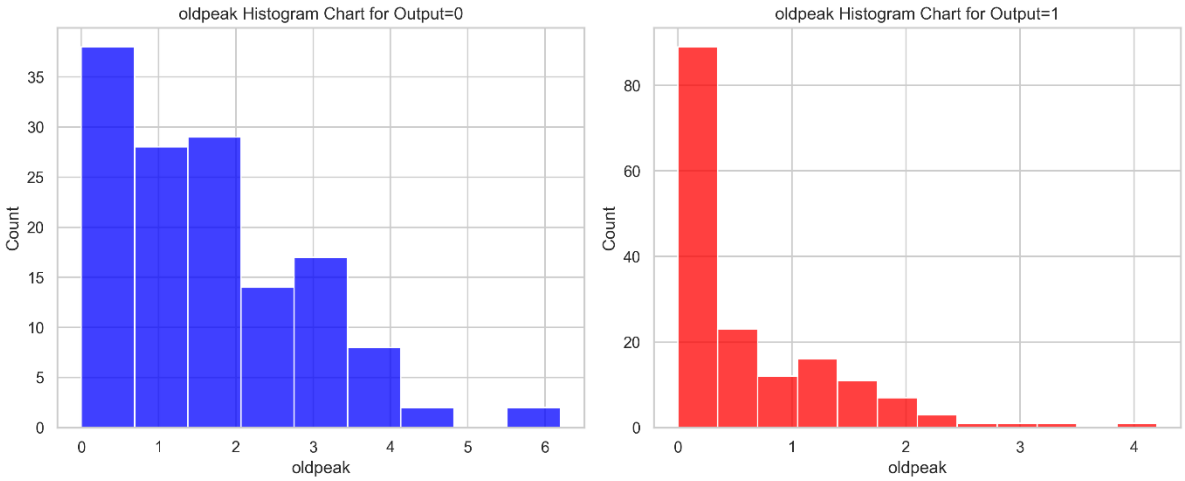
Output=1(Hasta Olan) için değerlerin en çok 0'da biraz 1'de ve çok az da 2 de bulunduğunu söyleyebiliriz.



slp Kategorik değişkeni 3 tane distinct değer almaktadır. Bu değerlerin ciddi kısmı 1 ve 2 de yer almaktadır.

Output=0(Hasta Olmayan) için değerlerin daha çok 1 ve biraz 2 de yer aldığını söylebiliriz.

Output=1(Hasta Olan) için değerlerin daha çok 2'de bulunduğunu ve biraz da 1 bulunduğunu söyleyebiliriz.

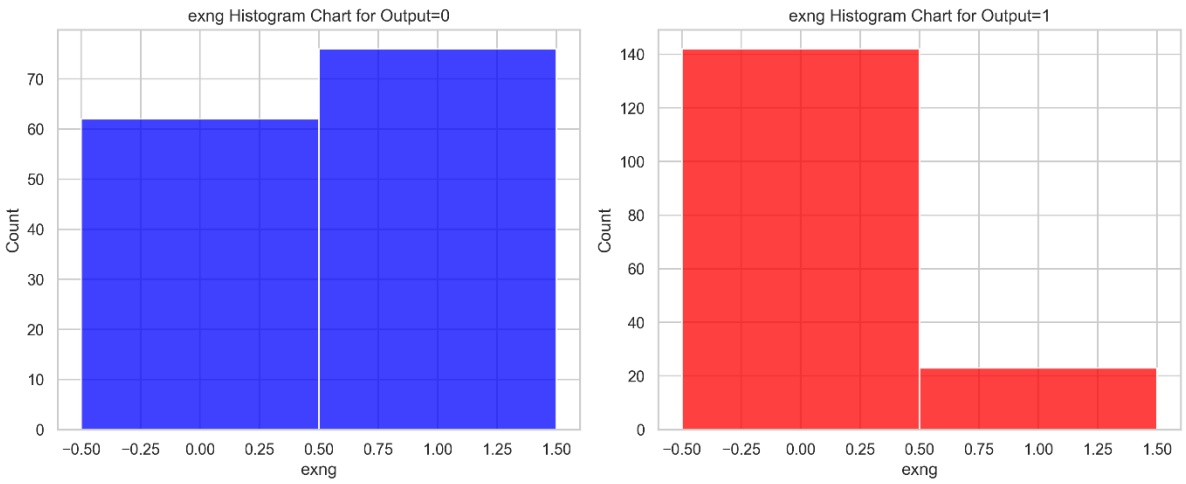


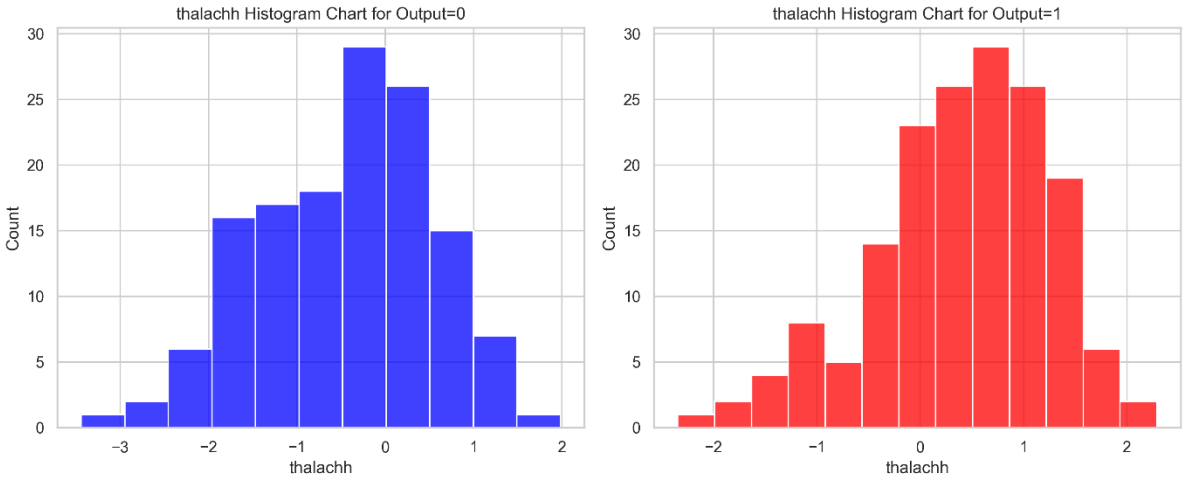
Oldpeak değişkeni

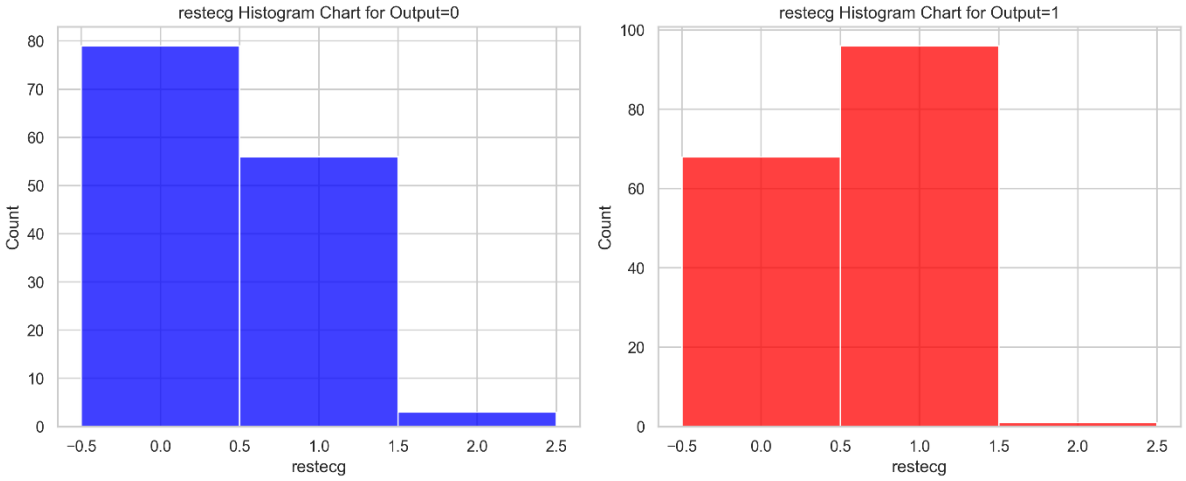
Mean: 1.04 & Std. Dev.: 1.16 Veri setine baktığımızda ciddi oranda 0 rakamını görmekteyiz.

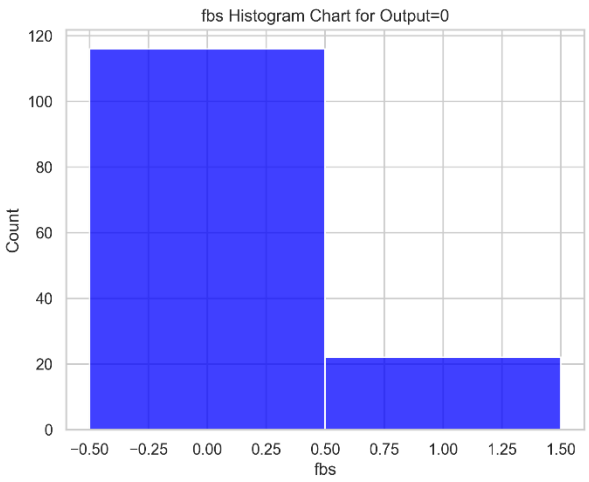
Output=0(Hasta Olmayan) için daha çok 0,1,2,3, değerleri görmekteyiz.

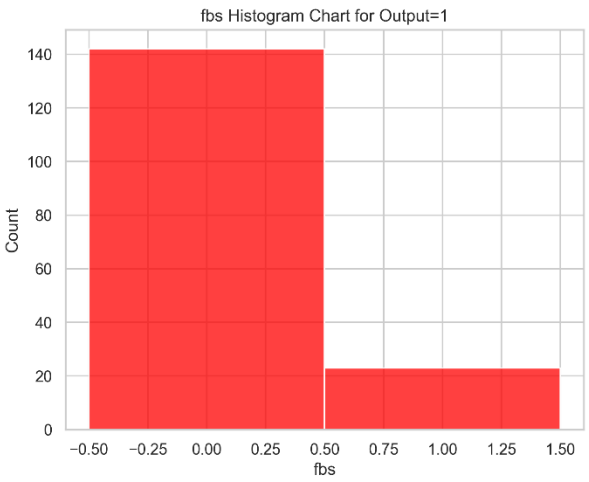
Output=1(Hasta olan) için en çok 0 ve biraz da 1 ve 2 değerlerini görmekteyiz.

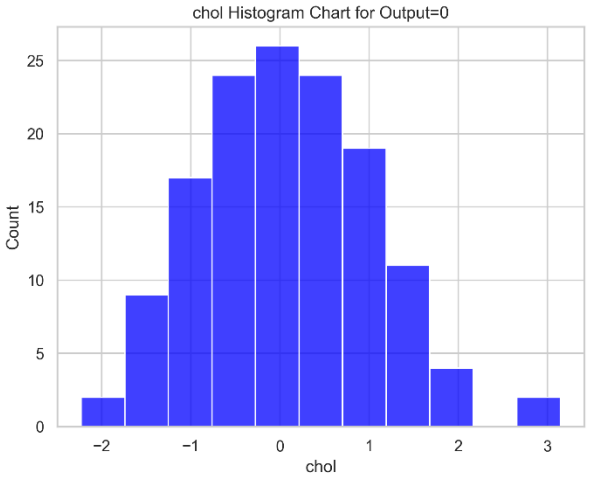

exng Histogram Chart for Output=0
Count
exng
exng Histogram Chart for Output=1
Count
exng

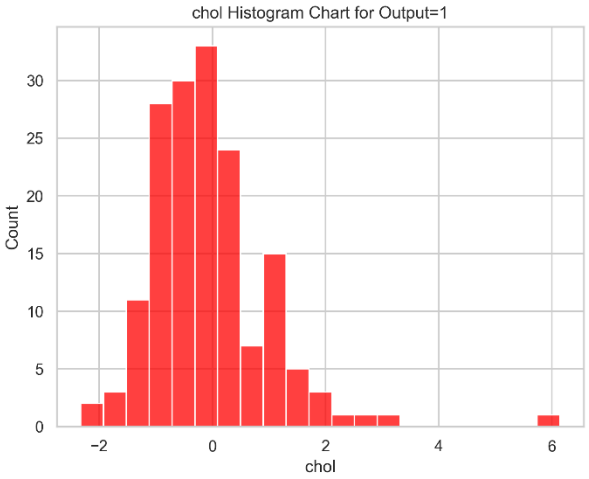

thalachh Histogram Chart for Output=0
Count
thalachh
thalachh Histogram Chart for Output=1
Count
thalachh

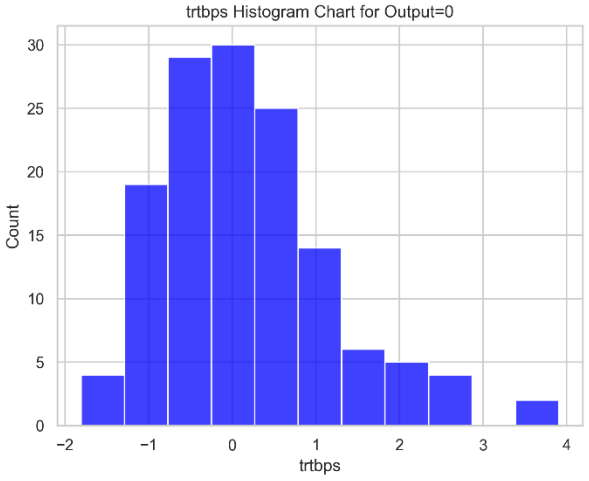

restecg Histogram Chart for Output=0
Count
restecg
restecg Histogram Chart for Output=1
Count
restecg

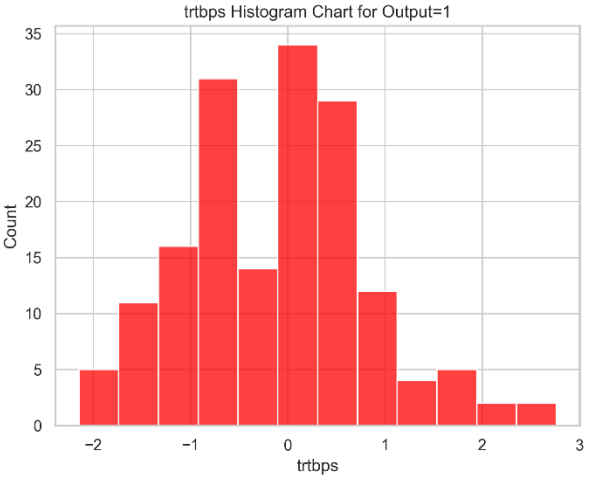

fbs Histogram Chart for Output=0
Count
fbs

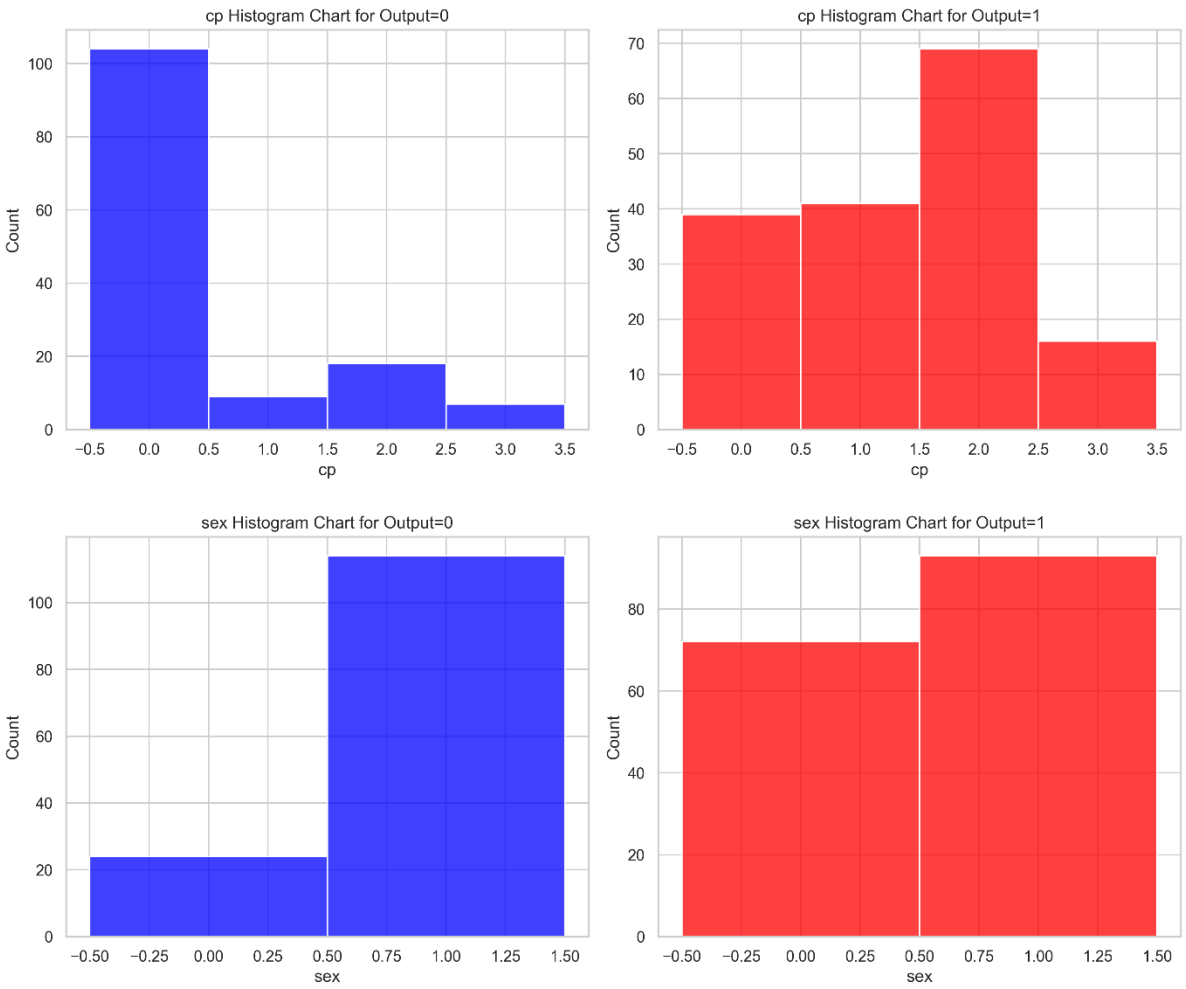

fbs Histogram Chart for Output=1
Count
fbs

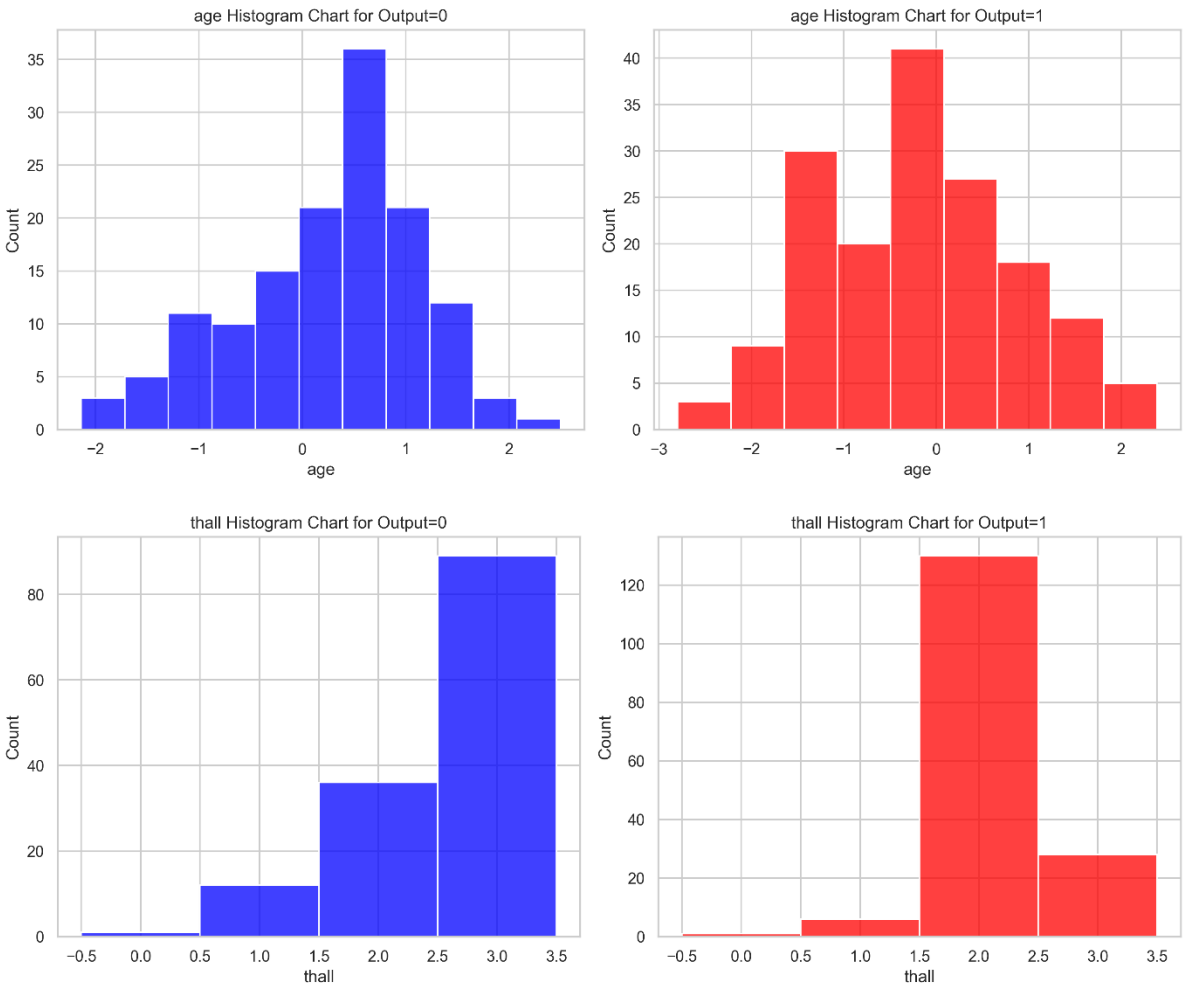

chol Histogram Chart for Output=0
Count
chol


chol Histogram Chart for Output=1
Count
chol


trtbps Histogram Chart for Output=0
Count
trtbps


trtbps Histogram Chart for Output=1
Count
trtbps


cp Histogram Chart for Output=0
Count
cp
cp Histogram Chart for Output=1
Count
cp
sex Histogram Chart for Output=0
Count
sex
sex Histogram Chart for Output=1
Count
sex



4. Deneysel Analiz

4.1. Veri setinde Ön İşlemler

Veri seti içerisindeki özellikler için Missing Value değerlerine bakılarak verinin ve özelliklerin modelleme açısından kalitesi ölçülmüştür.

Show Missing Value Table for Each Features

```
[14]: missing_values(df_data, threshold=0)
```

[14]:

	Count	Percent
age	0	0.0
sex	0	0.0
cp	0	0.0
trtbps	0	0.0
chol	0	0.0
fbs	0	0.0
restecg	0	0.0
thalachh	0	0.0
exng	0	0.0
oldpeak	0	0.0
slp	0	0.0
caa	0	0.0
thall	0	0.0
output	0	0.0

Veri seti içerisinde özelliklerin benzersiz değerlerine (low/high cardinality) bakılarak (n_unique) özelliklerin tipleri belirlenmiş ve dönüşümleri yapılmıştır.

Show High Cardinality Table for Each Features ¶

```
[15]: detect_cardinality(df=df_data)
```

[15]:

	Count
age	41
sex	2
cp	4
trtbps	49
chol	152
fbs	2
restecg	3
thalachh	91
exng	2
oldpeak	40
slp	3
caa	5
thall	4
output	2

```
[20]: feature_list = df_data.columns.to_list()
# feature_list.remove("output")
print("Features[{}]: {}".format(len(feature_list), feature_list))
print()

num_feature_list = df_data.select_dtypes(include=['number']).columns.to_list()
print("Numerical Features[{}]: {}".format(len(num_feature_list), num_feature_list))
print()

cat_feature_list = df_data.select_dtypes(include=['category']).columns.to_list()
print("Categorical Features[{}]: {}".format(len(cat_feature_list), cat_feature_list))

Features[14]: ['age', 'sex', 'cp', 'trtbps', 'chol', 'fbs', 'restecg', 'thalachh', 'exng', 'oldpeak', 'slp', 'caa', 'thall', 'output']

Numerical Features[5]: ['age', 'trtbps', 'chol', 'thalachh', 'oldpeak']

Categorical Features[9]: ['sex', 'cp', 'fbs', 'restecg', 'exng', 'slp', 'caa', 'thall', 'output']
```

Nümerik değişkenlerin Z-Score Normalization ile scale edilerek özellikler arasındaki ölçek(scaling) farklılıkları (Distance hesaplamalarının olumsuz etkilenmemesi için) ortadan kaldırılmıştır.

Veri seti içerisinde 1 tane duplicate gözlem tespit edilmiş ve ardından drop edilerek veri seti içerisinden çıkarılmıştır.

Duplicated Data

```
[20]: df_data.duplicated().sum()

[20]: 1

[21]: df_data[df_data.duplicated()]
```

[21]:

	age	sex	cp	trtbps	chol	fbs	restecg	thalachh	exng	oldpeak	slp	caa	thall	output
164	38	1	2	138	175	0	1	173	0	0.0	2	4	2	1

```
[22]: df_data.drop_duplicates(inplace=True)

[23]: df_data.reset_index(drop=True, inplace=True)
```

Not: proje dizini içerisinde src/ altında yer alan utils.py dosyası içerisinden yukarıda yapılan çalışmaların fonksiyonlarına ulaşabilirsiniz

4.2. Her Model için (Val. Set üzerinde) En Uygun Parametre Konfigürasyonların Paylaşılması

4.2.1. Validation Parametreleri

K Nearest Neighbor (KNN) Parameters

'**n_neighbors**': np.arange(start=3, stop=11, step=2),

'**weights**': ['uniform', 'distance'],

'metric': ['euclidean', 'manhattan', 'minkowski']}

Best Parameters: 0.933588 using {'weights': 'distance', 'n_neighbors': 5, 'metric': 'manhattan'}

Decision Tree (DT) Parameters

'criterion': ['gini', 'entropy'],

'max_depth': [3, 5, 7],

'**min_samples_split**': [2, 5, 10],

'**min_samples_leaf**': [3,5],

Best Parameters: 1.000000 using {'min_samples_split': 10, 'min_samples_leaf': 3, 'max_depth': 7, 'criterion': 'gini'}

Multi Layer Perceptron (MLP) Parameters

'**hidden_layer_sizes**': [(5,), (10,), (15,)],

'**activation**': ['logistic', 'tanh', 'relu'],

'**solver**': ['sgd',],

'**learning_rate**': ['constant', 'invscaling', 'adaptive'],

'**max_iter**': [150, 200, 250],

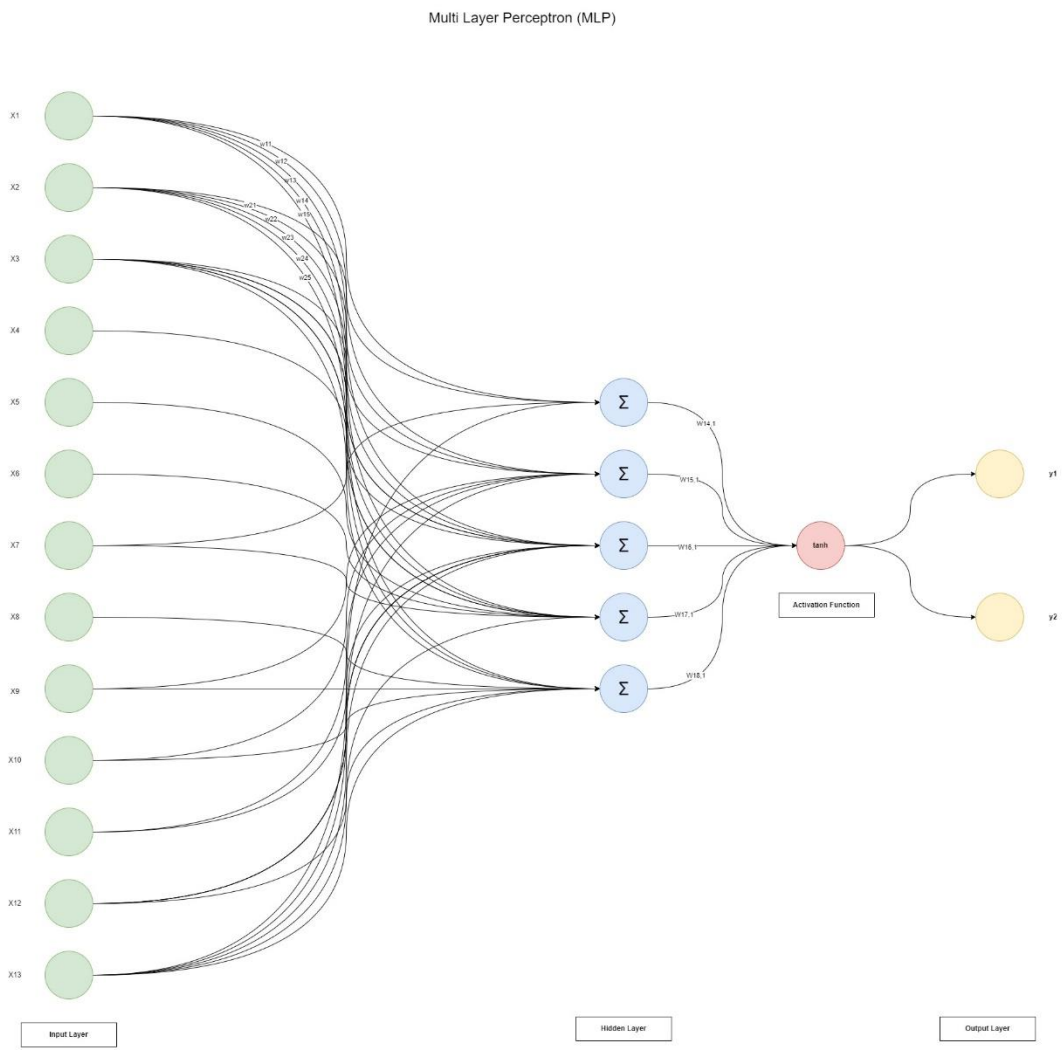
'**batch_size**': [16, 32, 64, X_train.shape[0]],

'**alpha**': [1e-4, 1e-3, 1e-2]

Best Parameters: 0.987585 using {'solver': 'sgd', 'max_iter': 200, 'learning_rate': 'adaptive', 'hidden_layer_sizes': (10,), 'batch_size': 16, 'alpha': 0.001, 'activation': 'tanh'}

4.2.2. MLP Sinir Ağı Mimarisi

4.2.2.1. MLP Mimarisi



4.2.2.2. Stochastic Gradient Descent ve Batch Gradient Descent

Stochastic Gradient Descent için Max_Iter:150

Batch Gradient Descent için BatchSize:16 iken Max_Iter:150

4.3. Test Set üzerinde Her Model İçin Confusion Matrix ve Performans Metrikleri

K Nearest Neighbor (KNN) Parameters

{'Accuracy': 0.9344262295081968,

'Precision': 0.9117647058823529,

'Recall': 0.96875,

'F1 Score': 0.9393939393939394,

'Confusion Matrix': array([[26, 3],

[1, 31]], dtype=int64)}

Decision Tree (DT) Parameters

{'Accuracy': 1.0,

'Precision': 1.0,

'Recall': 1.0,

'F1 Score': 1.0,

'Confusion Matrix': array([[29, 0],

[0, 32]], dtype=int64)}

Multi Layer Perceptron (MLP) Parameters

{'Accuracy': 0.9836065573770492,

'Precision': 0.9696969696969697,

'Recall': 1.0,

'F1 Score': 0.9846153846153847,

'Confusion Matrix': array([[28, 1],

[0, 32]], dtype=int64)}

4.4. Performans Metrikleri Hesaplanması ve Yorumlanması

Yukarıda Accuracy, Precision, Recall ve F1 Score performans metrikleri yer almaktadır.

4.5. En Başarılı Öğrenme Modeli

En başarılı Öğrenme Modeli aşağıdaki tabloda da görüldüğü üzere Decision Tree algoritmasıdır. Kolay okunulabilirliği ve hem kategorik hemde nümerik değişkenler ile çalışabilmesi de bunlara ilave olarak belirtilebilir.

	Validation Set Perf.	Test Set Perf. (N=61)			
	Accuracy	Accuracy	Precision	Recall	F1 Score
K Nearest Neighbor(KNN)	0.933588	0.93442	0.91176	0.96875	0.93939
Decision Tree(DT)	1	1	1	1	1
Multi Layer Perceptron(MLP)	0.958418	0,98360	0,96969	1	0,98461

Decision Tree Confusion Matrix

'Confusion Matrix': array([[29, 0],

[0, 32]], dtype=int64)}

5. Sonuç

Bu çalışma 3 ayrı algoritma denenmiştir. Değişkenler arasında Distance Base hesaplamalı KNN, klasik Makine Öğrenmesi yöntemlerinden Decision Tree ve Sinir Ağları temelli MultiLayer Perceptron alghoritmaları kullanılmıştır. Bütün algoritmalar kabul edilebilir seviye Validation ve Test set üzerinde

de başarım göstermişlerdir. Her bir algoritmayı o algoritmanın hiper parametreleri üzerinden optimize edilmesi ve K Fold Cross Validation ile modellemenin daha tutarlı olması sağlanmıştır. Problemin karmaşıklığının çok olmadığını da söylebilirim.