İSTİNYE ÜNİVERSİTESİ

FEMİLDA JOSEPHİN JOSEPH SHOBANA BAI

ENS305 MACHINE LEARNING PROJECT

POTABİLİTY OF WATER

ZEYNEP ÖZIŞIL(180722023) & ELİF BAYIR(180722020)

OBJECTİVE OF THE WORK

The deterioration of the quality of drinking water can lead to various diseases. Therefore, drinking water plays an important role in our lives. Our aim in this project is to explain the relationship between the factors that determine the potability rate of water.

MOTİVATİON OF THE WORK

Water is a substance that has a very important place in human life. We wanted to examine the potability of water, as we see it as one of the cornerstones of life.

SOCİETAL IMPACT

The quality of drinking water is important for the life of all living things. Living things can survive for weeks without food, but can withstand thirst for a few days.

Living things can survive for weeks without food, but can withstand thirst for a few days. Water has many vital effects such as protecting heart health and balancing body temperature.

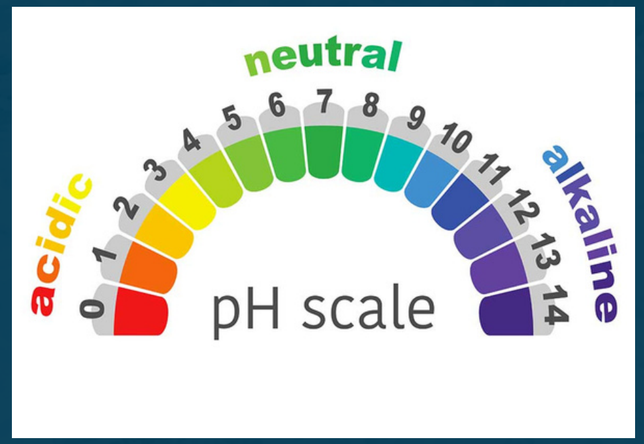
EXİSTİNG WORK

Factors affecting Water Potability

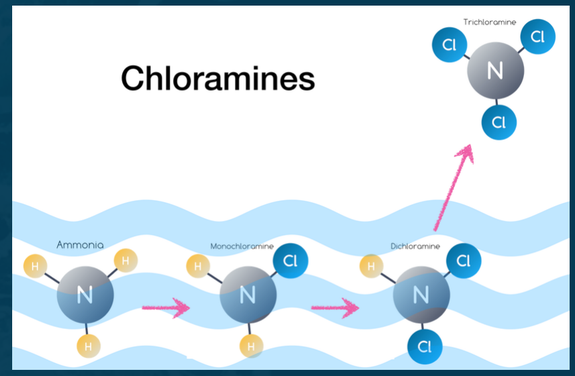
- PH
- HARDNESS
- SOLİDS
- CHLORAMİNE
- SULFATE
- CONDUCTİVİTY
- ORGANİC CARBON
- TRİHALOMETHANES
- TURBİDTY
- POTABİLİTY

PH

(0 to 14)

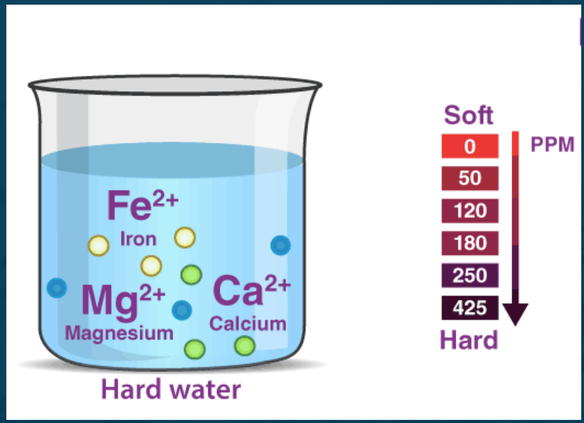


CHLORAMİNE (ppm)



HARDNESS

(mg/L)

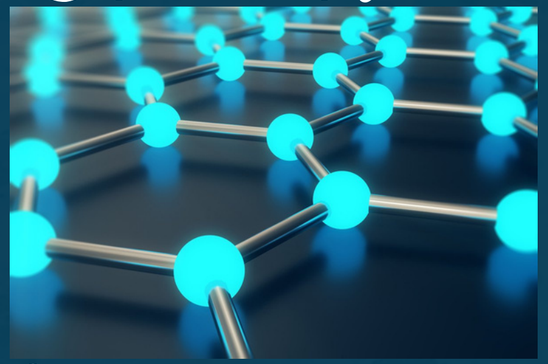


SULFATE (mg/L)

SOLİDS (ppm)

CONDUCTİVİTY

(µS/cm)

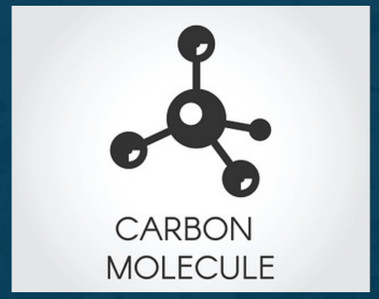


TURBİDTY

(Measure of light emiting property of water in NTU)

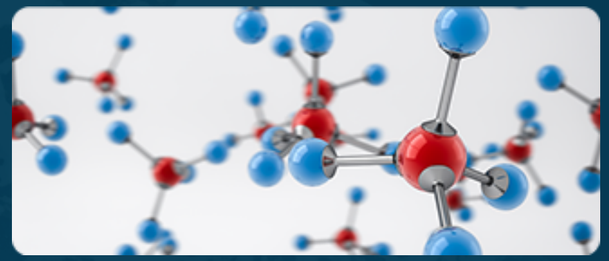
ORGANİC CARBON

(ppm)



POTABİLİTY

(Potable - 1 and Not potable - 0)



TRİHALOMETHANES (µg/L)

SYSTEM ARCHİTECTURE OF THE PROPOSED WORK

Visualizations

Data Preprocessing

Standardizing The Data

Importing The Dataset

Modelling

Libraries

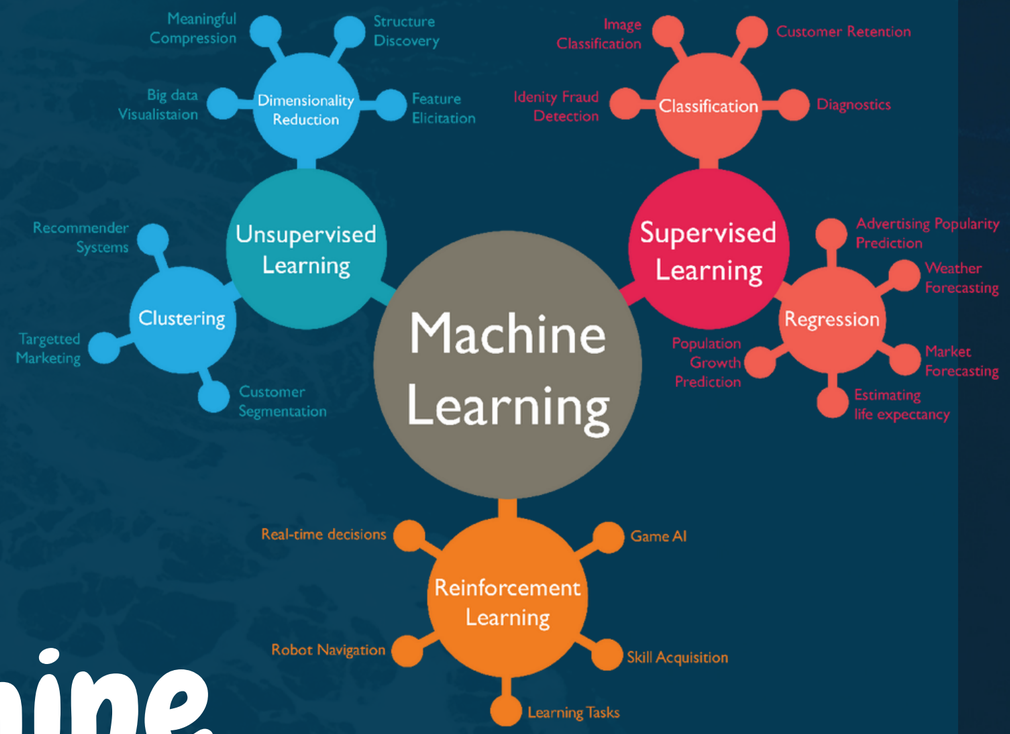
Final Model

DATASET DESCRİPTİON

A	B	C	D	E	F	G	H	I	J
ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
	204.8904555	20791.31898	7.300211873	368.5164413	564.3086542	10.37978308	86.99097046	2.963135381	0
3.716080075	129.4229205	18630.05786	6.635245884		592.8853591	15.18001312	56.32907628	4.500656275	0
8.099124189	224.2362594	19909.54173	9.275883603		418.6062131	16.86863693	66.42009251	3.05593375	0
8.316765884	214.3733941	22018.41744	8.059332377	356.8861356	363.2665162	18.4365245	100.3416744	4.628770537	0
9.092223456	181.1015092	17978.98634	6.546599974	310.1357375	398.4108134	11.55827944	31.99799273	4.075075425	0
5.584086638	188.3133238	28748.68774	7.544868789	326.6783629	280.4679159	8.39973464	54.91786184	2.559708228	0
10.22386216	248.0717353	28749.71654	7.513408466	393.6633955	283.6516335	13.78969532	84.60355617	2.672988737	0
8.635848719	203.3615226	13672.09176	4.563008686	303.3097712	474.6076449	12.3638167	62.79830896	4.401424715	0
	118.9885791	14285.58385	7.804173553	268.6469407	389.3755659	12.70604897	53.92884577	3.595017181	0

METHODOLOGİES USED/ALGORİTHM USED

- Logistic Regression
- Decision Tree
- Random Forest
- KNeighbours
- Support Vector Machine



Implementation / Analysis

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

```
df=pd.read_csv('water_potability.csv')
df
```

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0
...	...	...	...	...	...	...	...	...	...	...
3271	4.668102	193.681735	47580.991603	7.166639	359.948574	526.424171	13.894419	66.687695	4.435821	1
3272	7.808856	193.553212	17329.802160	8.061362	NaN	392.449580	19.903225	NaN	2.798243	1
3273	9.419510	175.762646	33155.578218	7.350233	NaN	432.044783	11.039070	69.845400	3.298875	1
3274	5.126763	230.603758	11983.869376	6.303357	NaN	402.883113	11.168946	77.488213	4.708658	1
3275	7.874671	195.102299	17404.177061	7.509306	NaN	327.459760	16.140368	78.698446	2.309149	1

3276 rows × 10 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3276 entries, 0 to 3275
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ph               2785 non-null   float64
 1   Hardness         3276 non-null   float64
 2   Solids           3276 non-null   float64
 3   Chloramines      3276 non-null   float64
 4   Sulfate          2495 non-null   float64
 5   Conductivity     3276 non-null   float64
 6   Organic_carbon   3276 non-null   float64
 7   Trihalomethanes  3114 non-null   float64
 8   Turbidity        3276 non-null   float64
 9   Potability       3276 non-null   int64  
dtypes: float64(9), int64(1)
memory usage: 256.1 KB
```

```
df.head()
```

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
0	NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.963135	0
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013	56.329076	4.500656	0
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637	66.420093	3.055934	0
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.628771	0
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.075075	0

0
sn.

```
df.nunique()
```

```
ph                 2785
Hardness           3276
Solids             3276
Chloramines        3276
Sulfate            2495
Conductivity       3276
Organic_carbon     3276
Trihalomethanes    3114
Turbidity          3276
Potability            2
dtype: int64
```

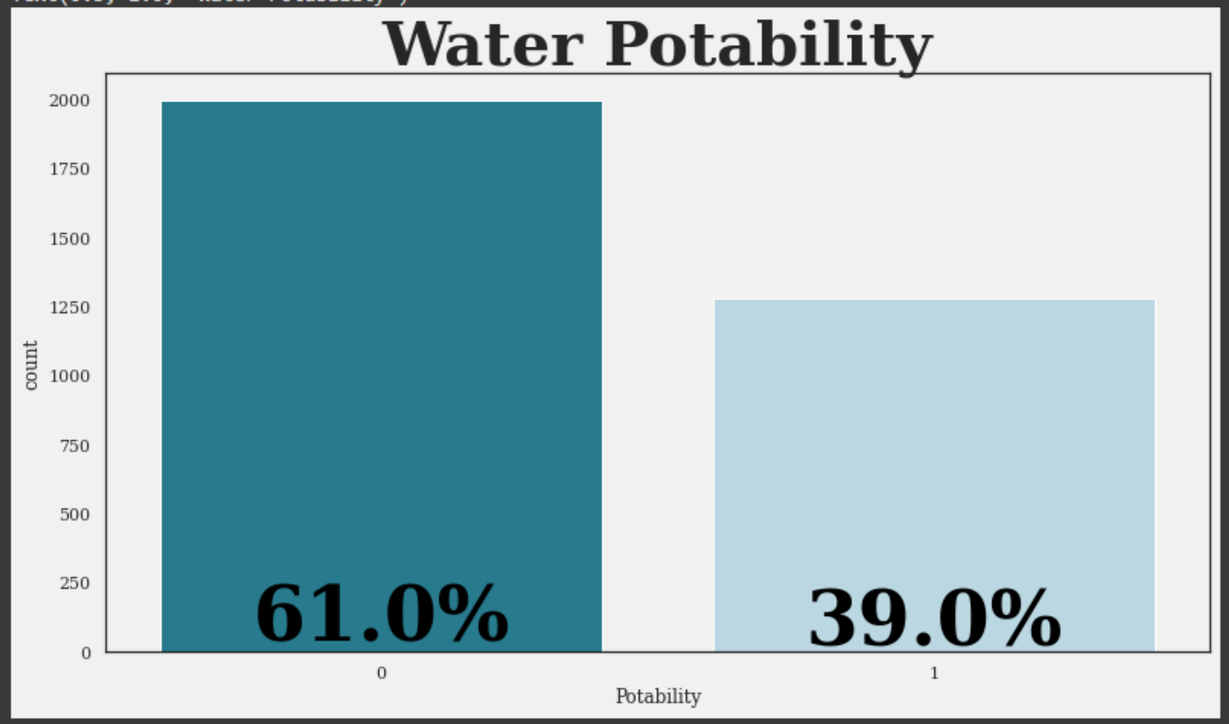

```
df.describe()
```

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turbidity	Potability
count	2785.000000	3276.000000	3276.000000	3276.000000	2495.000000	3276.000000	3276.000000	3114.000000	3276.000000	3276.000000
mean	7.080795	196.369496	22014.092526	7.122277	333.775777	426.205111	14.284970	66.396293	3.966786	0.390110
std	1.594320	32.879761	8768.570828	1.583085	41.416840	80.824064	3.308162	16.175008	0.780382	0.487849
min	0.000000	47.432000	320.942611	0.352000	129.000000	181.483754	2.200000	0.738000	1.450000	0.000000
25%	6.093092	176.850538	15666.690297	6.127421	307.699498	365.734414	12.065801	55.844536	3.439711	0.000000
50%	7.036752	196.967627	20927.833607	7.130299	333.073546	421.884968	14.218338	66.622485	3.955028	0.000000
75%	8.062066	216.667456	27332.762127	8.114887	359.950170	481.792304	16.557652	77.337473	4.500320	1.000000
max	14.000000	323.124000	61227.196008	13.127000	481.030642	753.342620	28.300000	124.000000	6.739000	1.000000

```
[124] colors = ['#17869E','#B4DBE9']
      sns.set(palette=colors, font='Serif', style='white', rc={'axes.facecolor':'#f1f1f1', 'figure.facecolor':'#f1f1f1'})
      sns.palplot(colors)
```

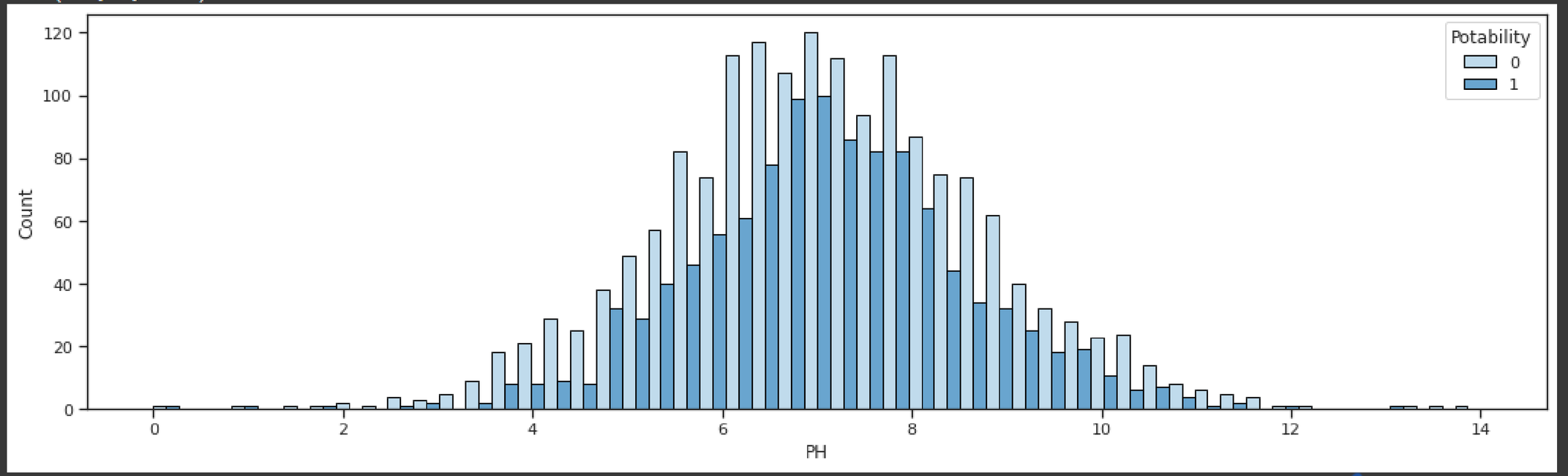
```
fig = plt.figure(figsize=(13,7))
ax=sns.countplot(data=df, x='Potability')
for i in ax.patches:
    ax.text(x=i.get_x()+i.get_width()/2, y=i.get_height()/50, s=f"{np.round(i.get_height()/len(df)*100,0)}%", ha='center', size=50, weight='bold', rotation=0, color='black')
plt.title("Water Potability", size=40, weight='bold')
```

```
Text(0.5, 1.0, 'Water Potability')
```



```
plt.subplots(figsize=(18,5))
sns.histplot(df,x="ph", hue="Potability",multiple="dodge",palette="Blues" ,edgecolor="black")
plt.xlabel('PH')
```

```
Text(0.5, 0, 'PH')
```



```
[23] df[df["Potability"]==1]["ph"].median()
```

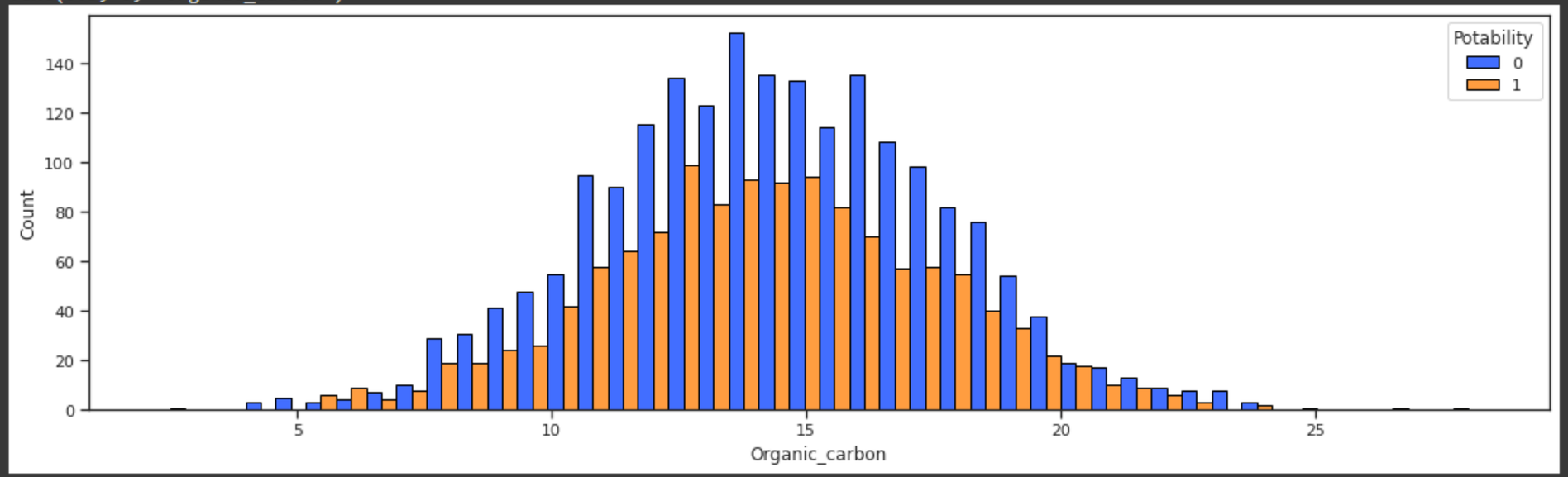
```
7.036752103833548
```

```
df[df["Potability"]==0]["ph"].median()
```

```
7.035455515887571
```

```
plt.subplots(figsize=(18,5))
sns.histplot(df,x="Organic_carbon", hue="Potability",multiple="dodge",palette="bright" ,edgecolor="black")
plt.xlabel('Organic_carbon')
```

```
Text(0.5, 0, 'Organic_carbon')
```



```
[26] df[df["Potability"]==1]["Organic_carbon"].median()
```

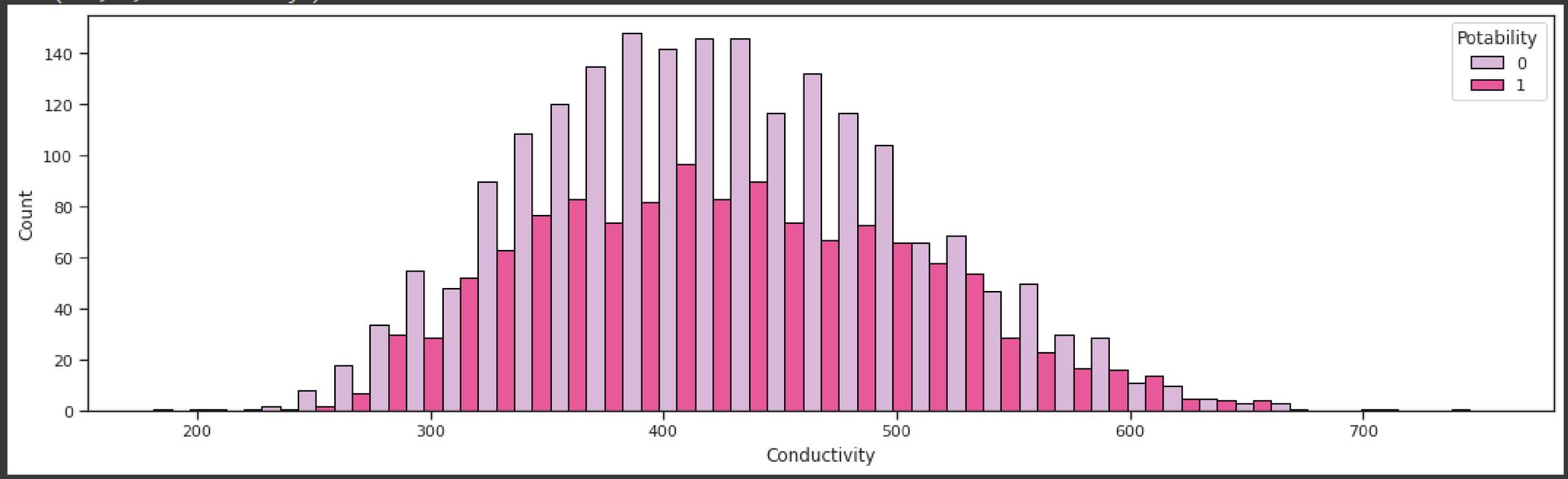
```
14.162808700391992
```

```
df[df["Potability"]==0]["Organic_carbon"].median()
```

```
14.293507768607894
```

```
plt.subplots(figsize=(18,5))
sns.histplot(df,x="Conductivity", hue="Potability",multiple="dodge",palette="PuRd" ,edgecolor="black")
plt.xlabel('Conductivity')
```

```
Text(0.5, 0, 'Conductivity')
```



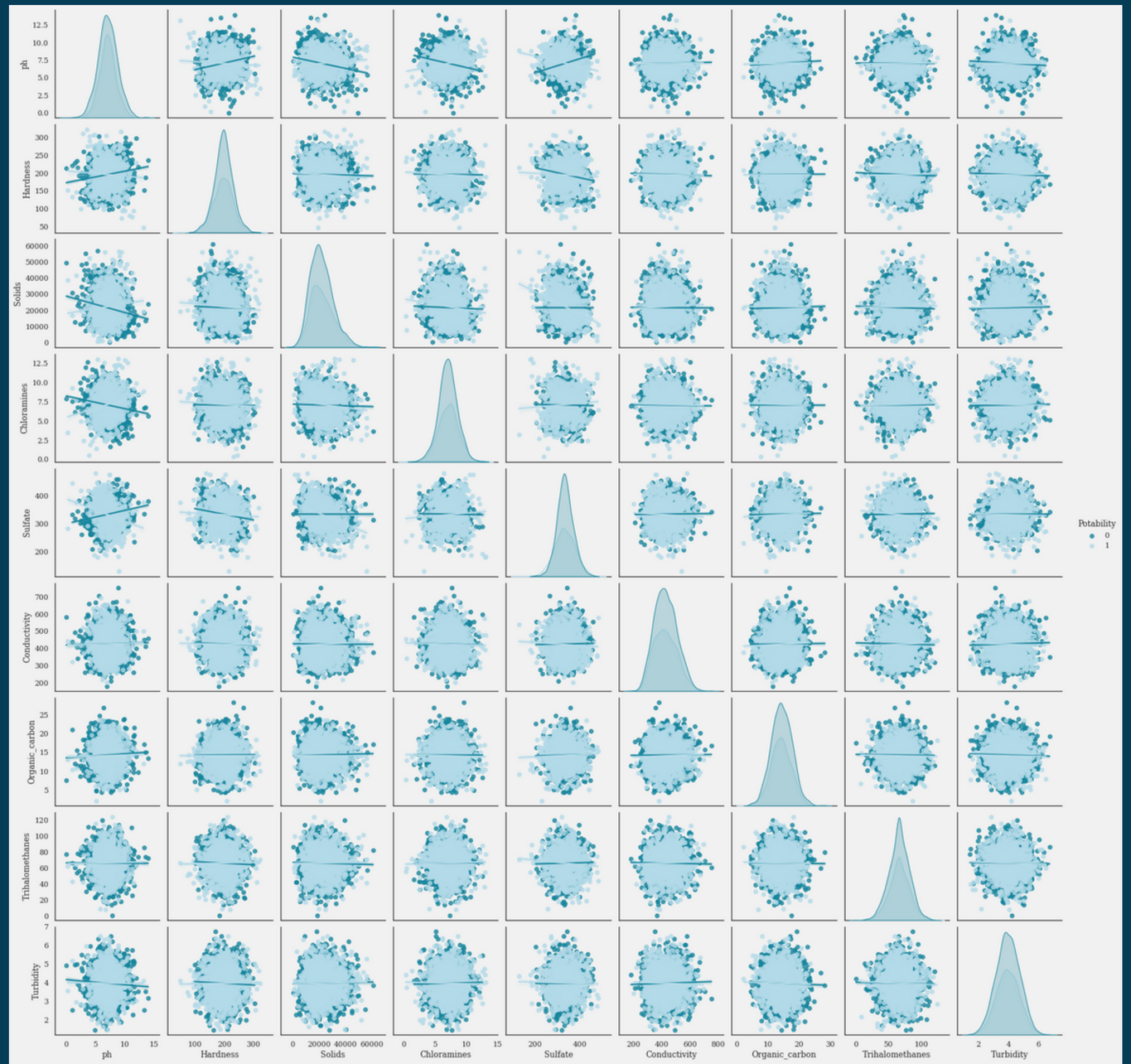

```
[29] df[df["Potability"]==1]["Conductivity"].median()
```

```
420.71272912166967
```

```
df[df["Potability"]==0]["Conductivity"].median()
```

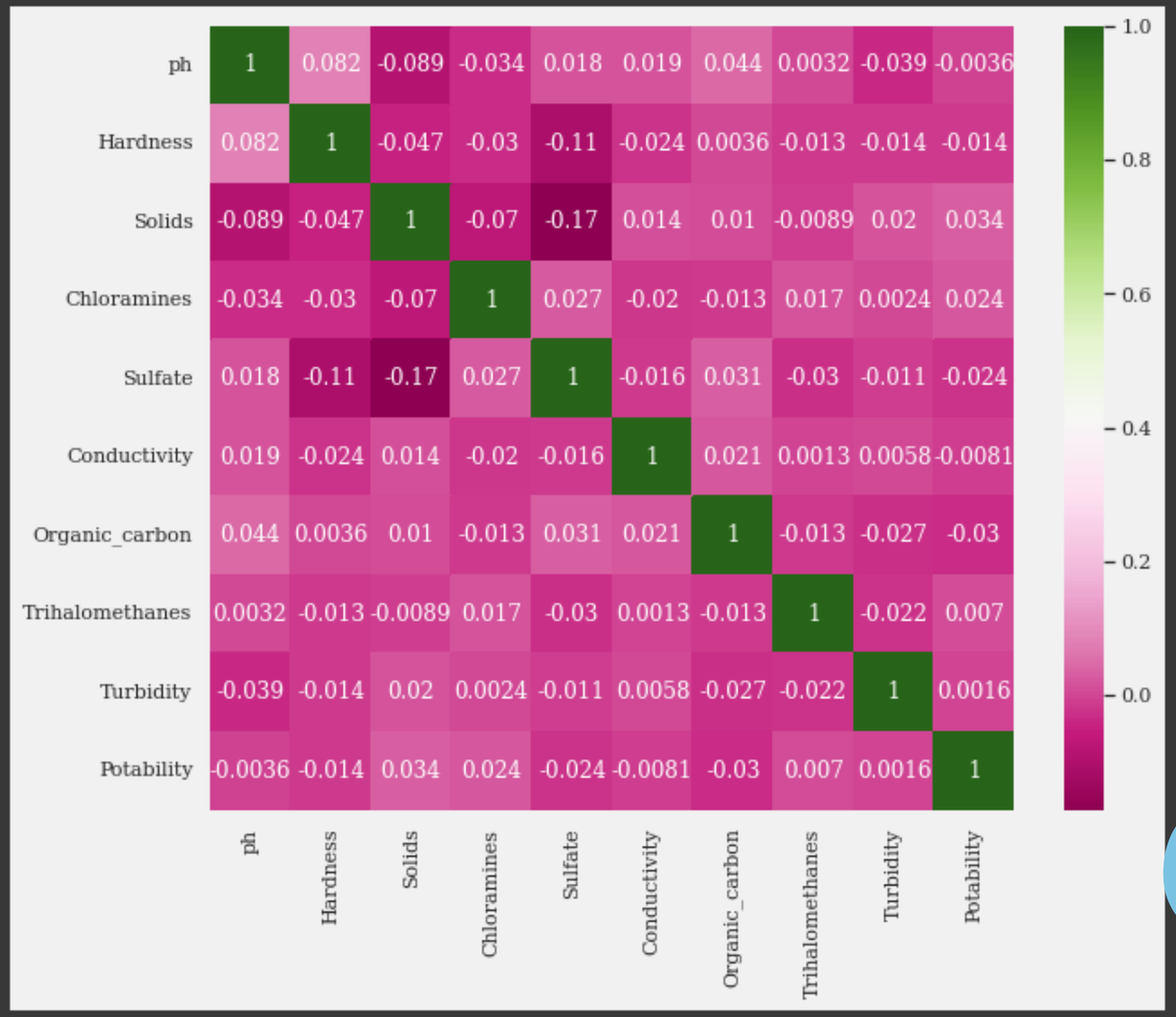
```
422.2293312022622
```

```
sns.pairplot(df, hue='Potability', kind='reg')
```



$$r = \frac{\sum XY - \frac{(\sum X)(\sum Y)}{n}}{\sqrt{\left(\sum X^2 - \frac{(\sum X)^2}{n}\right)\left(\sum Y^2 - \frac{(\sum Y)^2}{n}\right)}}$$

```
data=df.corr()
features=data.index
plt.figure(figsize=(10,8))
s=sns.heatmap(df[features].corr(),annot=True,cmap='PiYG')
```



```
[35] import missingno as msno
```

```
msno.matrix(df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30fc391e50>
```

```
[192] df.isnull().sum()
```

```
ph                 491
Hardness             0
Solids               0
Chloramines          0
Sulfate            781
Conductivity         0
Organic_carbon       0
Trihalomethanes    162
Turbidity            0
Potability           0
dtype: int64
```

```
[193] df.nunique()
```

```
ph                 2785
Hardness           3276
Solids             3276
Chloramines        3276
Sulfate            2495
Conductivity       3276
Organic_carbon     3276
Trihalomethanes    3114
Turbidity          3276
Potability            2
dtype: int64
```

```
[194] df['Sulfate'] = df['Sulfate'].replace(np.nan,0)
      df['ph'] = df['ph'].replace(np.nan,0)
      df['Trihalomethanes'] = df['Trihalomethanes'].replace(np.nan,0)
```

```
[195] df["ph"]=df["ph"].fillna(df['ph'].median)
      df["Sulfate"]=df["Sulfate"].fillna(df['Sulfate'].median)
      df["Trihalomethanes"]=df["Trihalomethanes"].fillna(df['Trihalomethanes'].median)
```

```
[ ] df.isnull().sum()
```

```
ph                0
Hardness          0
Solids            0
Chloramines       0
Sulfate           0
Conductivity      0
Organic_carbon    0
Trihalomethanes   0
Turbidity         0
Potability        0
dtype: int64
```

```
[197] x = df.drop('Potability',axis=1).values
      y = df['Potability'].values
```

```
[198] from sklearn.model_selection import train_test_split
      from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

```
[214] x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=0)
```

```
[229] from sklearn.preprocessing import StandardScaler
      st_x= StandardScaler()
      x_train= st_x.fit_transform(x_train)
      x_test= st_x.transform(x_test)
```

```
[408] from sklearn.linear_model import LogisticRegression
      classifier = LogisticRegression(random_state = 0)
      classifier.fit(x_train, y_train)
```

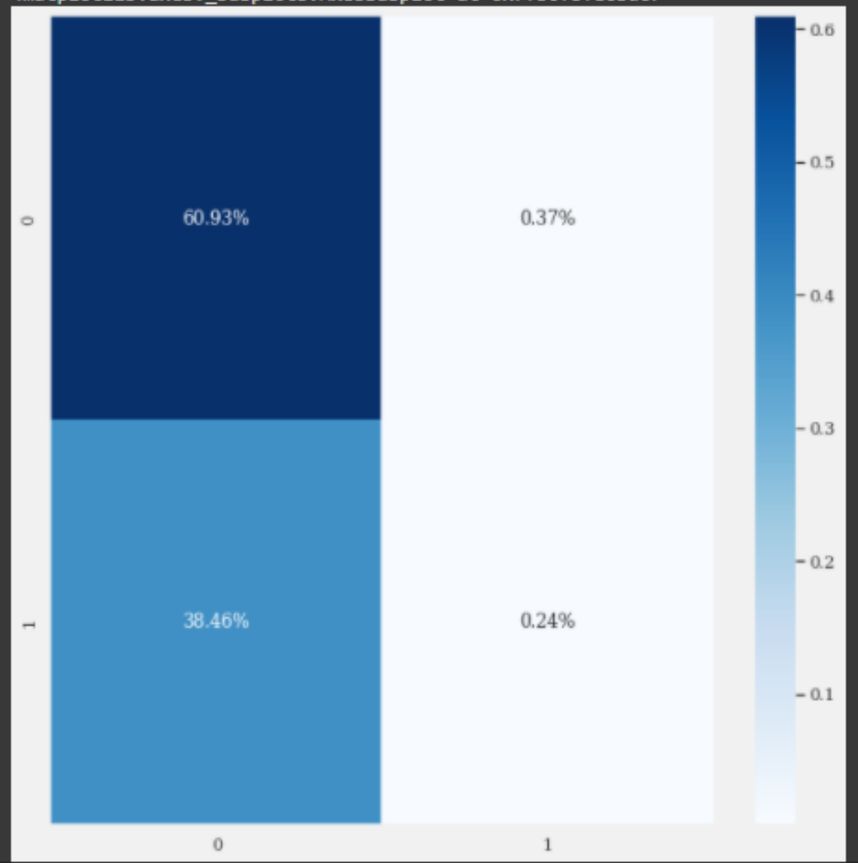
```
LogisticRegression(random_state=0)
```

```
[409] y_pred= classifier.predict(x_test)
      from sklearn.metrics import confusion_matrix
      cm= confusion_matrix(y_test, y_pred)
      print(cm)
      lg=accuracy_score(y_test, y_pred)
      print(lg)
```

```
[[499   3]
 [315   2]]
0.6117216117216118
```

```
sns.heatmap(cm/np.sum(cm), annot = True, fmt=  '0.2%', cmap = 'Blues')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30f5fe6bd0>
```



```
[411] from sklearn.neighbors import KNeighborsClassifier
      knn= KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
      knn.fit(x_train, y_train)
```

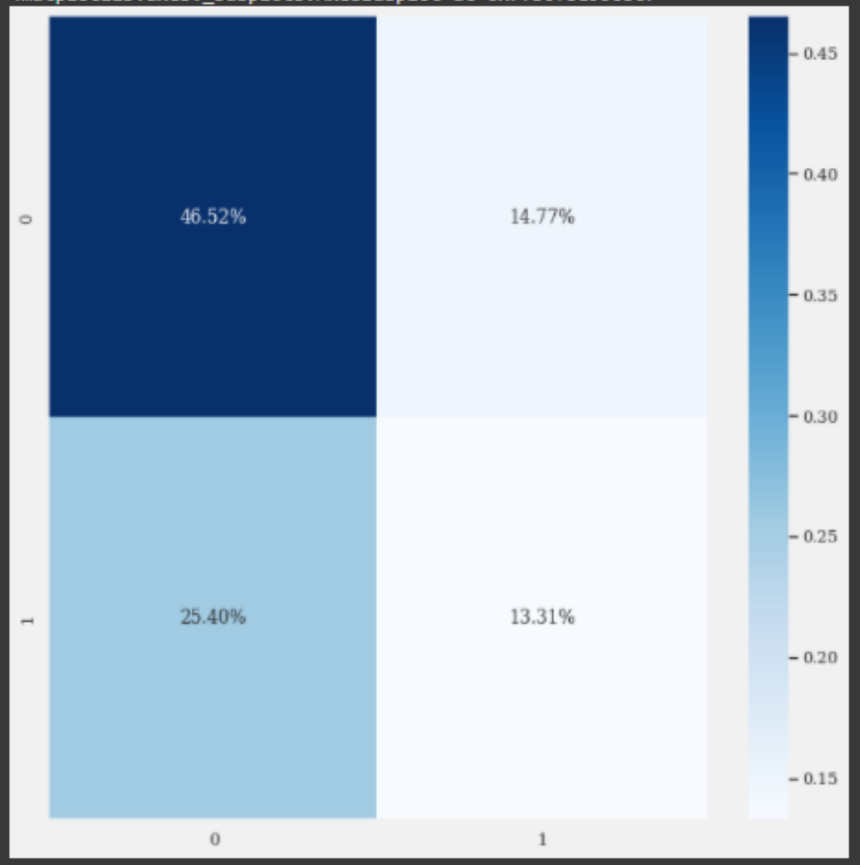
```
KNeighborsClassifier()
```

```
[412] y_pred= knn.predict(x_test)
      cm2= confusion_matrix(y_test, y_pred)
      print(cm2)
      kn=accuracy_score(y_test, y_pred)
      print(kn)
```

```
[[381 121]
 [208 109]]
0.5982905982905983
```

```
sns.heatmap(cm2/np.sum(cm2), annot = True, fmt=  '0.2%', cmap = 'Blues')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30f5e99650>
```



```
[414] from sklearn.svm import SVC, LinearSVC
      classifier = SVC()
      svc.fit(x_train, y_train)
```

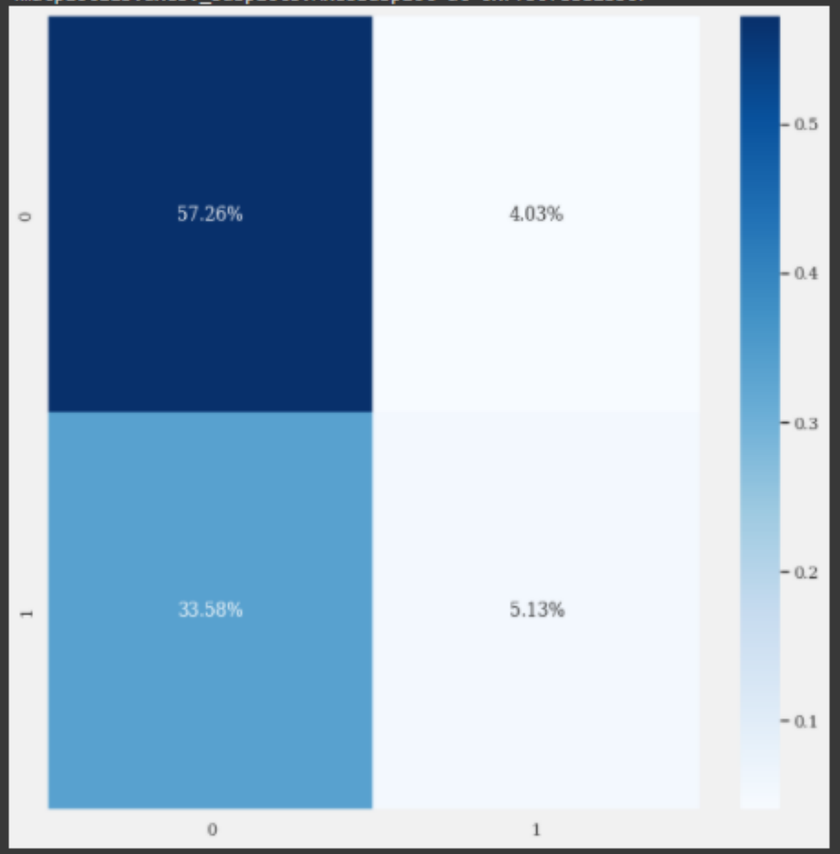
```
SVC()
```

```
[415] y_pred= svc.predict(x_test)
      cm3= confusion_matrix(y_test, y_pred)
      print(cm3)
      sv=accuracy_score(y_test, y_pred)
      print(sv)
```

```
[[469  33]
 [275  42]]
0.6239316239316239
```

```
sns.heatmap(cm3/np.sum(cm3), annot = True, fmt=  '0.2%', cmap = 'Blues')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30fe8e2850>
```



```
[417] from sklearn.tree import DecisionTreeClassifier
      decision_tree = DecisionTreeClassifier()
      decision_tree.fit(x_train, y_train)
```

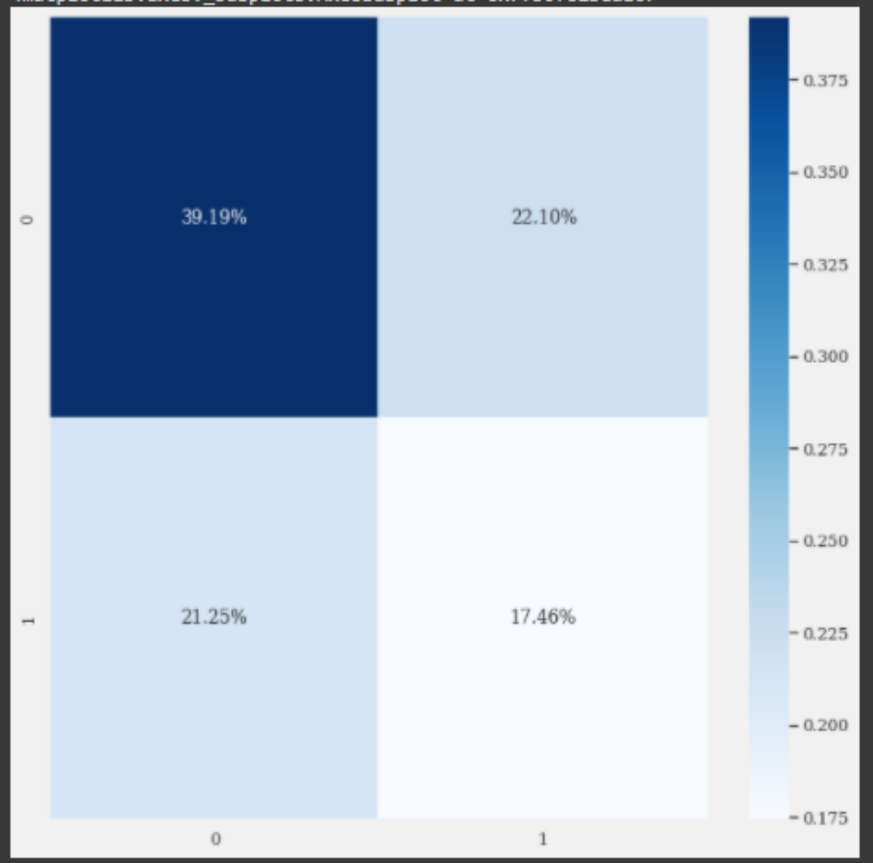
```
DecisionTreeClassifier()
```

```
[418] y_pred= decision_tree.predict(x_test)
      cm4= confusion_matrix(y_test, y_pred)
      print(cm4)
      dt=accuracy_score(y_test, y_pred)
      print(dt)
```

```
[[321 181]
 [174 143]]
0.5665445665445665
```

```
[ ] sns.heatmap(cm4/np.sum(cm4), annot = True, fmt=  '0.2%', cmap = 'Blues')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30f62bea10>
```



```
[420] from sklearn.ensemble import RandomForestClassifier
      random_forest = RandomForestClassifier(n_estimators=100)
      random_forest.fit(x_train, y_train)
```

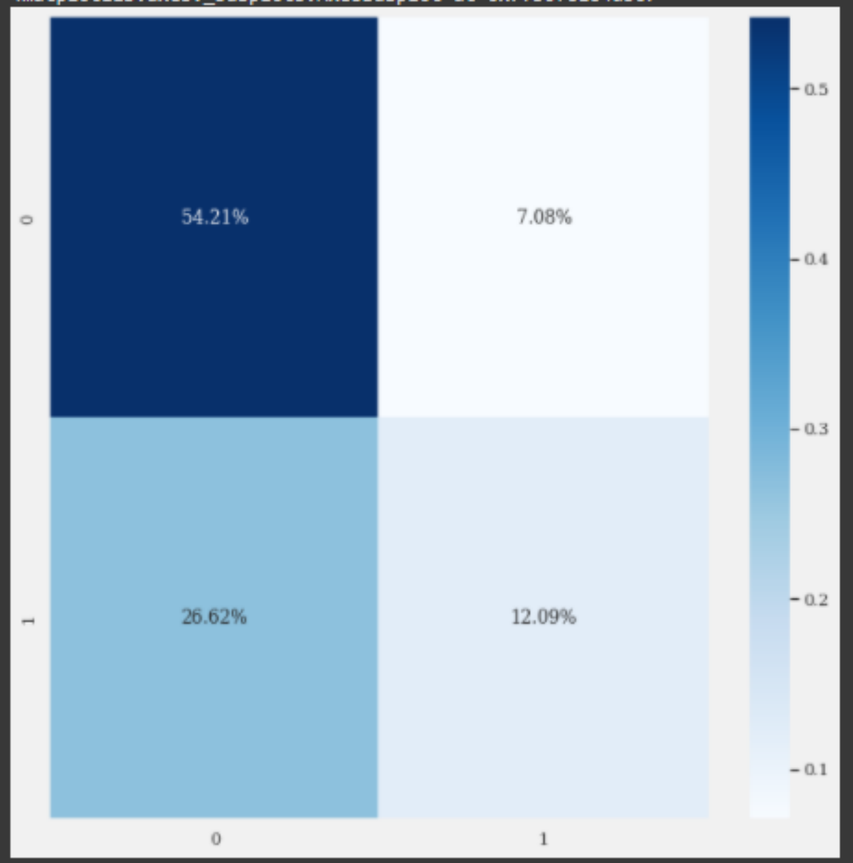
```
RandomForestClassifier()
```

```
[421] y_pred= random_forest.predict(x_test)
      cm5= confusion_matrix(y_test, y_pred)
      print(cm5)
      rm=accuracy_score(y_test, y_pred)
      print(rm)
```

```
[[444  58]
 [218  99]]
0.663003663003663
```

```
[422] sns.heatmap(cm5/np.sum(cm5), annot = True, fmt=  '0.2%', cmap = 'Blues')
```

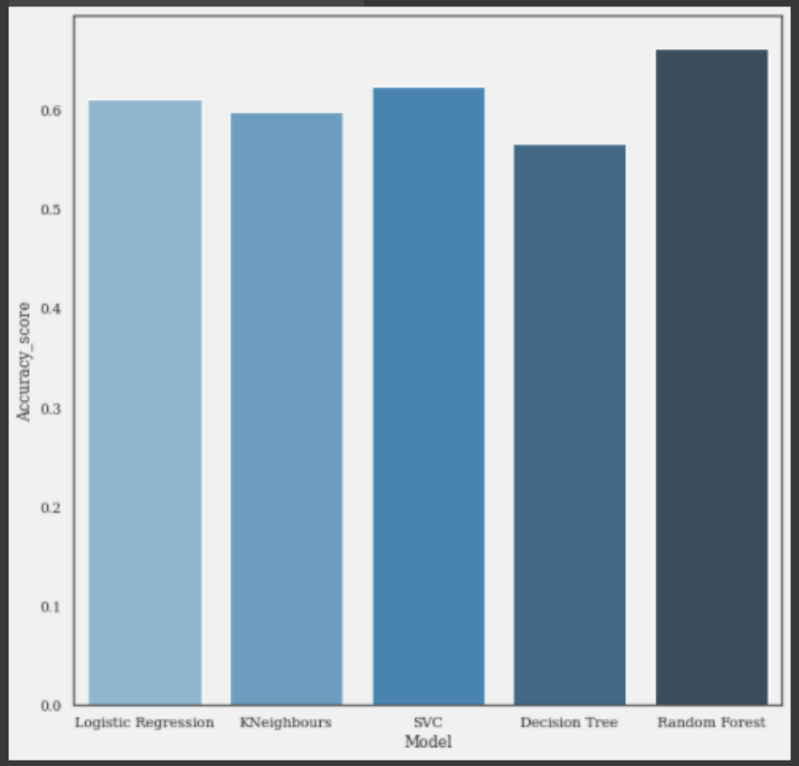
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f30f6164d50>
```



Conclusion

```
models = pd.DataFrame({'Model':['Logistic Regression', 'KNeighbours', 'SVC', 'Decision Tree', 'Random Forest'],'Accuracy_score' :[lg, kn, sv, dt, rm]})
models
sns.barplot(x='Model', y='Accuracy_score', data=models, palette="Blues_d")
models.sort_values(by='Accuracy_score', ascending=False)
```

	Model	Accuracy_score
4	Random Forest	0.663004
2	SVC	0.623932
0	Logistic Regression	0.611722
1	KNeighbours	0.598291
3	Decision Tree	0.566545



References

Data Sources: https://www.kaggle.com/kubrakurt/su-kalitesi/data?scriptVersionId=68655683
Visual Sources: https://www.canva.com/
Others:
https://abapsikoloji.com/suyun-onemi-insan-sagligi-uzerindeki-etkisi/
https://www.milliyet.com.tr/pembenar/galeri/hangi-su-daha-saglikli-iyi-ve-kaliteli-bir-suyun-ozellikleri-6539768/4
https://www.miato.com.tr/su-kalite-ozellikleri/

- The python file and slide presentation are loaded on the blackboard.
- Electronic copy of the code(COLAB) has been sent to your e-mail address via drive.

İSTİNYE ÜNİVERSİTESİ

Thank you!

ZEYNEP ÖZIŞIL 180722023

ELİF BAYIR 180722020