```
import pandas as pd
df = pd.read_csv('The_Cancer_data_1500_V2.csv')
df
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnosis
0	58	1	16.085313	0	1	8.146251	4.148219	1	1
1	71	0	30.828784	0	1	9.361630	3.519683	0	0
2	48	1	38.785084	0	2	5.135179	4.728368	0	1
3	34	0	30.040296	0	0	9.502792	2.044636	0	0
4	62	1	35.479721	0	0	5.356890	3.309849	0	1
...	...	...	...	...	...	...	...	...	...
1495	62	1	25.090025	0	0	9.892167	1.284158	0	1
1496	31	0	33.447125	0	1	1.668297	2.280636	1	1
1497	63	1	32.613861	1	1	0.466848	0.150101	0	1
1498	55	0	25.568216	0	0	7.795317	1.986138	1	1
1499	67	1	23.663104	0	0	2.525860	2.856600	1	0

1500 rows × 9 columns

```
from dataPrep import dataPreperation
df2 = dataPreperation(df)
df2
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnosis
1	71	0	30.828784	0	1	9.361630	3.519683	0	0
2	48	1	38.785084	0	2	5.135179	4.728368	0	1
3	34	0	30.040296	0	0	9.502792	2.044636	0	0
4	62	1	35.479721	0	0	5.356890	3.309849	0	1
5	27	0	37.105162	0	1	3.941905	2.324274	0	0
...	...	...	...	...	...	...	...	...	...
1492	66	1	19.484117	0	0	1.918732	0.726430	0	1
1493	59	1	39.266914	0	0	0.612167	1.581462	0	1
1494	79	1	17.832588	0	0	5.909161	4.880353	0	1
1495	62	1	25.090025	0	0	9.892167	1.284158	0	1
1497	63	1	32.613861	1	1	0.466848	0.150101	0	1

1284 rows × 9 columns

```
df3 = dataPreperation(df2)
df3
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnosis
1	71	0	30.828784	0	1	9.361630	3.519683	0	0
2	48	1	38.785084	0	2	5.135179	4.728368	0	1
3	34	0	30.040296	0	0	9.502792	2.044636	0	0
4	62	1	35.479721	0	0	5.356890	3.309849	0	1
5	27	0	37.105162	0	1	3.941905	2.324274	0	0
...	...	...	...	...	...	...	...	...	...
1492	66	1	19.484117	0	0	1.918732	0.726430	0	1
1493	59	1	39.266914	0	0	0.612167	1.581462	0	1
1494	79	1	17.832588	0	0	5.909161	4.880353	0	1
1495	62	1	25.090025	0	0	9.892167	1.284158	0	1
1497	63	1	32.613861	1	1	0.466848	0.150101	0	1

1284 rows × 9 columns

```
df4 = dataPreperation(df3)
df4
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnosis
1	71	0	30.828784	0	1	9.361630	3.519683	0	0
2	48	1	38.785084	0	2	5.135179	4.728368	0	1
3	34	0	30.040296	0	0	9.502792	2.044636	0	0
4	62	1	35.479721	0	0	5.356890	3.309849	0	1
5	27	0	37.105162	0	1	3.941905	2.324274	0	0
...	...	...	...	...	...	...	...	...	...
1492	66	1	19.484117	0	0	1.918732	0.726430	0	1
1493	59	1	39.266914	0	0	0.612167	1.581462	0	1
1494	79	1	17.832588	0	0	5.909161	4.880353	0	1
1495	62	1	25.090025	0	0	9.892167	1.284158	0	1
1497	63	1	32.613861	1	1	0.466848	0.150101	0	1

1284 rows × 9 columns

```
df5 = dataPreperation(df4)
df5
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnosis
1	71	0	30.828784	0	1	9.361630	3.519683	0	0
2	48	1	38.785084	0	2	5.135179	4.728368	0	1
3	34	0	30.040296	0	0	9.502792	2.044636	0	0
4	62	1	35.479721	0	0	5.356890	3.309849	0	1
5	27	0	37.105162	0	1	3.941905	2.324274	0	0
...	...	...	...	...	...	...	...	...	...
1492	66	1	19.484117	0	0	1.918732	0.726430	0	1
1493	59	1	39.266914	0	0	0.612167	1.581462	0	1
1494	79	1	17.832588	0	0	5.909161	4.880353	0	1
1495	62	1	25.090025	0	0	9.892167	1.284158	0	1
1497	63	1	32.613861	1	1	0.466848	0.150101	0	1

1284 rows × 9 columns

```
X = df5.drop('Diagnosis', axis=1)
y = df5['Diagnosis']


y.value_counts()
```

Diagnosis	count
0	907
1	377

dtype: int64

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, random_state=30)


def runModel(model, X_train, X_test, y_train, y_test):
  model.fit(X_train, y_train)

  y_pred = model.predict(X_test)
  y_train_pred = model.predict(X_train)

  acc_test = accuracy_score(y_test, y_pred)
  acc_train = accuracy_score(y_train, y_train_pred)

  y_pred_proba = model.predict_proba(X_test)

  cm = confusion_matrix(y_test, y_pred)
  # cr = classification_report(y_test, y_pred)
  print(f'Train Accuracy: {acc_train}')
  print(f'Test Accuracy: {acc_test}')
  print(cm)
  return acc_test, acc_train, cm, y_pred_proba, y_pred
```

```
from sklearn.linear_model import LogisticRegression as LR
from sklearn.tree import DecisionTreeClassifier as DT
from xgboost import XGBClassifier as XGB
from sklearn.neighbors import KNeighborsClassifier as KNN
from sklearn.ensemble import RandomForestClassifier as RF
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report


# model = LogisticRegression(max_iter= 10000)

models_arr = [LR(max_iter=10000), DT(), XGB(), KNN(), RF()]
for model in models_arr:
  print(model.__class__.__name__)
  runModel(model, X_train, X_test, y_train, y_test)
  acc_test, acc_train, cm, y_pred_proba, y_pred = runModel(model, X_train, X_test, y_train, y_test)
```

```
LogisticRegression
Train Accuracy: 0.844097995545657
Test Accuracy: 0.8523316062176166
[[246  25]
 [ 32  83]]
Train Accuracy: 0.844097995545657
Test Accuracy: 0.8523316062176166
[[246  25]
 [ 32  83]]
DecisionTreeClassifier
Train Accuracy: 1.0
Test Accuracy: 0.8652849740932642
[[240  31]
 [ 21  94]]
Train Accuracy: 1.0
Test Accuracy: 0.8730569948186528
[[241  30]
 [ 19  96]]
XGBClassifier
Train Accuracy: 1.0
Test Accuracy: 0.9248704663212435
[[256  15]
 [ 14 101]]
Train Accuracy: 1.0
Test Accuracy: 0.9248704663212435
[[256  15]
 [ 14 101]]
KNeighborsClassifier
Train Accuracy: 0.821826280623608
Test Accuracy: 0.7305699481865285
[[242  29]
 [ 75  40]]
Train Accuracy: 0.821826280623608
Test Accuracy: 0.7305699481865285
[[242  29]
 [ 75  40]]
RandomForestClassifier
Train Accuracy: 1.0
Test Accuracy: 0.917098445595855
[[265   6]
 [ 26  89]]
Train Accuracy: 1.0
Test Accuracy: 0.9196891191709845
[[265   6]
 [ 25  90]]
```

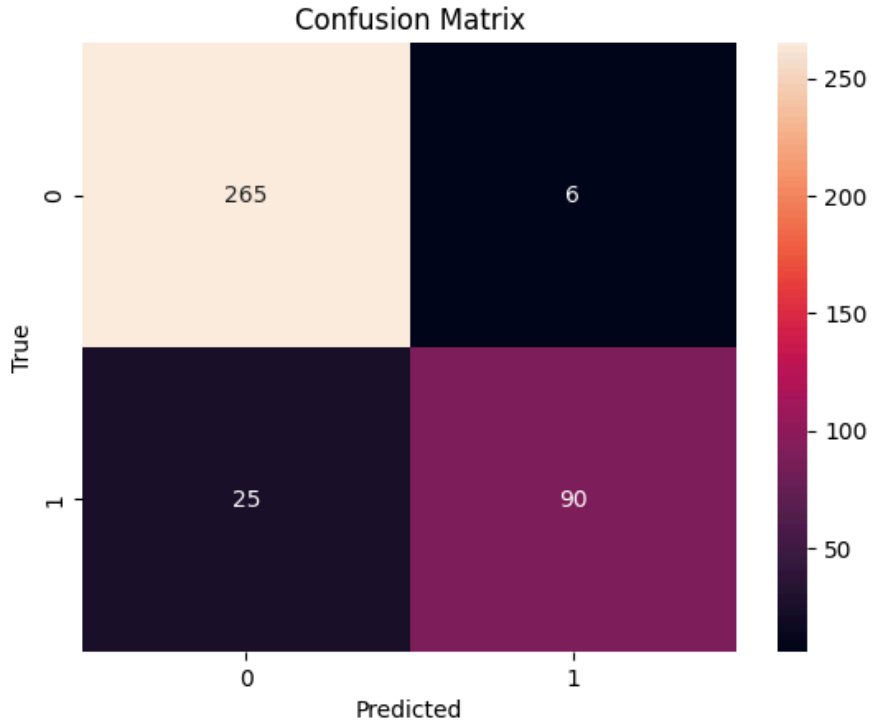
```
df5
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnosis
1	71	0	30.828784	0	1	9.361630	3.519683	0	0
2	48	1	38.785084	0	2	5.135179	4.728368	0	1
3	34	0	30.040296	0	0	9.502792	2.044636	0	0
4	62	1	35.479721	0	0	5.356890	3.309849	0	1
5	27	0	37.105162	0	1	3.941905	2.324274	0	0
...	...	...	...	...	...	...	...	...	...
1492	66	1	19.484117	0	0	1.918732	0.726430	0	1
1493	59	1	39.266914	0	0	0.612167	1.581462	0	1
1494	79	1	17.832588	0	0	5.909161	4.880353	0	1
1495	62	1	25.090025	0	0	9.892167	1.284158	0	1
1497	63	1	32.613861	1	1	0.466848	0.150101	0	1

1284 rows × 9 columns

```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming 'cm' is your confusion matrix from the previous code
sns.heatmap(cm, annot=True, fmt='d')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()
```




```
from sklearn.preprocessing import MinMaxScaler
minmaxscaller = MinMaxScaler()
```

```
df5_scaled = minmaxscaller.fit_transform(df5)
df5_scaled = pd.DataFrame(df5_scaled, columns=df5.columns)
df5_scaled
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnc
0	0.850000	0.0	0.634530	0.0	0.5	0.936891	0.706683	0.0	
1	0.466667	1.0	0.953481	0.0	1.0	0.513808	0.949447	0.0	
2	0.233333	0.0	0.602921	0.0	0.0	0.951021	0.410420	0.0	
3	0.700000	1.0	0.820976	0.0	0.0	0.536002	0.664538	0.0	
4	0.116667	0.0	0.886136	0.0	0.5	0.394357	0.466586	0.0	
...	...	...	...	...	...	...	...	...	
1279	0.766667	1.0	0.179747	0.0	0.0	0.191830	0.145659	0.0	
1280	0.650000	1.0	0.972796	0.0	0.0	0.061039	0.317392	0.0	
1281	0.983333	1.0	0.113541	0.0	0.0	0.591286	0.979973	0.0	
1282	0.700000	1.0	0.404475	0.0	0.0	0.989999	0.257679	0.0	
1283	0.716667	1.0	0.706090	1.0	0.5	0.046492	0.029904	0.0	

1284 rows × 9 columns

```
X_scaled = df5_scaled.drop('Diagnosis', axis=1)
y_scaled = df5_scaled['Diagnosis']

X_train_scaled, X_test_scaled, y_train_scaled, y_test_scaled = train_test_split(X_scaled, y_scaled, train_size=0.

X_train_scaled.shape, X_test_scaled.shape, y_train_scaled.shape, y_test_scaled.shape
```

```
((898, 8), (386, 8), (898,), (386,))
```

```
for model in models_arr:
  print(model.__class__.__name__)
  runModel(model, X_train_scaled, X_test_scaled, y_train_scaled, y_test_scaled)
```

```
LogisticRegression
Train Accuracy: 0.8385300668151447
Test Accuracy: 0.8549222797927462
[[249  22]
 [ 34  81]]
DecisionTreeClassifier
Train Accuracy: 1.0
Test Accuracy: 0.8549222797927462
[[237  34]
 [ 22  93]]
XGBClassifier
Train Accuracy: 1.0
Test Accuracy: 0.9248704663212435
[[256  15]
 [ 14 101]]
KNeighborsClassifier
Train Accuracy: 0.9198218262806236
Test Accuracy: 0.8497409326424871
[[253  18]
 [ 40  75]]
RandomForestClassifier
```

```
Train Accuracy: 1.0
Test Accuracy: 0.927461139896373
[[266   5]
 [ 23  92]]
```

```
y_train_scaled.value_counts()
```

Diagnosis	count
0.0	636
1.0	262

dtype: int64

```
from imblearn.over_sampling import SMOTE
smote = SMOTE()
X_train_smote, y_train_smote = smote.fit_resample(X_train_scaled, y_train_scaled)
```

```
y_train_smote.value_counts()
```

Diagnosis	count
0.0	636
1.0	636

dtype: int64

```
for model in models_arr:
  print(model.__class__.__name__)
  runModel(model, X_train_smote, X_test_scaled, y_train_smote, y_test_scaled)
```

```
LogisticRegression
Train Accuracy: 0.8301886792452831
Test Accuracy: 0.8290155440414507
[[217  54]
 [ 12 103]]
DecisionTreeClassifier
Train Accuracy: 1.0
Test Accuracy: 0.8367875647668394
[[227  44]
 [ 19  96]]
XGBClassifier
Train Accuracy: 1.0
Test Accuracy: 0.917098445595855
[[249  22]
 [ 10 105]]
KNeighborsClassifier
Train Accuracy: 0.9229559748427673
Test Accuracy: 0.8393782383419689
[[229  42]
 [ 20  95]]
RandomForestClassifier
Train Accuracy: 1.0
Test Accuracy: 0.9404145077720207
[[261  10]
 [ 13 102]]
```

```
df_smote = pd.concat([X_train_smote, y_train_smote], axis=1)
df_smote
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagno
0	0.366667	1.0	0.051236	0.0	0.000000	0.405179	0.466312	0.0	
1	1.000000	1.0	0.185725	1.0	0.500000	0.003593	0.795827	0.0	
2	0.733333	1.0	0.303569	1.0	0.500000	0.972818	0.727256	0.0	
3	0.366667	0.0	0.263679	1.0	0.000000	0.050134	0.073313	0.0	
4	0.233333	0.0	0.702698	0.0	0.500000	0.383690	0.664091	0.0	
...	...	...	...	...	...	...	...	...	
1267	0.939675	1.0	0.788314	0.0	0.000000	0.380175	0.664243	0.0	
1268	0.791359	1.0	0.873269	1.0	0.000000	0.417423	0.314809	0.0	
1269	0.723926	1.0	0.918265	1.0	0.000000	0.645312	0.390853	0.0	
1270	0.655971	1.0	0.849150	1.0	0.438262	0.480406	0.457323	0.0	
1271	0.588416	1.0	0.454688	1.0	0.500000	0.145078	0.522572	0.0	

1272 rows × 9 columns

```
df6 = dataPreperation(df_smote)
df6
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagno
0	0.366667	1.0	0.051236	0.0	0.000000	0.405179	0.466312	0.0	
1	1.000000	1.0	0.185725	1.0	0.500000	0.003593	0.795827	0.0	
2	0.733333	1.0	0.303569	1.0	0.500000	0.972818	0.727256	0.0	
3	0.366667	0.0	0.263679	1.0	0.000000	0.050134	0.073313	0.0	
4	0.233333	0.0	0.702698	0.0	0.500000	0.383690	0.664091	0.0	
...	...	...	...	...	...	...	...	...	
1267	0.939675	1.0	0.788314	0.0	0.000000	0.380175	0.664243	0.0	
1268	0.791359	1.0	0.873269	1.0	0.000000	0.417423	0.314809	0.0	
1269	0.723926	1.0	0.918265	1.0	0.000000	0.645312	0.390853	0.0	
1270	0.655971	1.0	0.849150	1.0	0.438262	0.480406	0.457323	0.0	
1271	0.588416	1.0	0.454688	1.0	0.500000	0.145078	0.522572	0.0	

1272 rows × 9 columns

```
df7 = dataPreperation(df6)
df7
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnc
0	0.366667	1.0	0.051236	0.0	0.000000	0.405179	0.466312	0.0	
1	1.000000	1.0	0.185725	1.0	0.500000	0.003593	0.795827	0.0	
2	0.733333	1.0	0.303569	1.0	0.500000	0.972818	0.727256	0.0	
3	0.366667	0.0	0.263679	1.0	0.000000	0.050134	0.073313	0.0	
4	0.233333	0.0	0.702698	0.0	0.500000	0.383690	0.664091	0.0	
...	...	...	...	...	...	...	...	...	
1267	0.939675	1.0	0.788314	0.0	0.000000	0.380175	0.664243	0.0	
1268	0.791359	1.0	0.873269	1.0	0.000000	0.417423	0.314809	0.0	
1269	0.723926	1.0	0.918265	1.0	0.000000	0.645312	0.390853	0.0	
1270	0.655971	1.0	0.849150	1.0	0.438262	0.480406	0.457323	0.0	
1271	0.588416	1.0	0.454688	1.0	0.500000	0.145078	0.522572	0.0	

1272 rows × 9 columns

```
df8 = dataPreperation(df7)
df8
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnc
0	0.366667	1.0	0.051236	0.0	0.000000	0.405179	0.466312	0.0	
1	1.000000	1.0	0.185725	1.0	0.500000	0.003593	0.795827	0.0	
2	0.733333	1.0	0.303569	1.0	0.500000	0.972818	0.727256	0.0	
3	0.366667	0.0	0.263679	1.0	0.000000	0.050134	0.073313	0.0	
4	0.233333	0.0	0.702698	0.0	0.500000	0.383690	0.664091	0.0	
...	...	...	...	...	...	...	...	...	
1267	0.939675	1.0	0.788314	0.0	0.000000	0.380175	0.664243	0.0	
1268	0.791359	1.0	0.873269	1.0	0.000000	0.417423	0.314809	0.0	
1269	0.723926	1.0	0.918265	1.0	0.000000	0.645312	0.390853	0.0	
1270	0.655971	1.0	0.849150	1.0	0.438262	0.480406	0.457323	0.0	
1271	0.588416	1.0	0.454688	1.0	0.500000	0.145078	0.522572	0.0	

1272 rows × 9 columns

```
df9 = dataPreperation(df8)
df9
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnc
0	0.366667	1.0	0.051236	0.0	0.000000	0.405179	0.466312	0.0	
1	1.000000	1.0	0.185725	1.0	0.500000	0.003593	0.795827	0.0	
2	0.733333	1.0	0.303569	1.0	0.500000	0.972818	0.727256	0.0	
3	0.366667	0.0	0.263679	1.0	0.000000	0.050134	0.073313	0.0	
4	0.233333	0.0	0.702698	0.0	0.500000	0.383690	0.664091	0.0	
...	...	...	...	...	...	...	...	...	
1267	0.939675	1.0	0.788314	0.0	0.000000	0.380175	0.664243	0.0	
1268	0.791359	1.0	0.873269	1.0	0.000000	0.417423	0.314809	0.0	
1269	0.723926	1.0	0.918265	1.0	0.000000	0.645312	0.390853	0.0	
1270	0.655971	1.0	0.849150	1.0	0.438262	0.480406	0.457323	0.0	
1271	0.588416	1.0	0.454688	1.0	0.500000	0.145078	0.522572	0.0	

1272 rows × 9 columns

```
df10 = dataPreperation(df9)
df10
```

	Age	Gender	BMI	Smoking	GeneticRisk	PhysicalActivity	AlcoholIntake	CancerHistory	Diagnc
0	0.366667	1.0	0.051236	0.0	0.000000	0.405179	0.466312	0.0	
1	1.000000	1.0	0.185725	1.0	0.500000	0.003593	0.795827	0.0	
2	0.733333	1.0	0.303569	1.0	0.500000	0.972818	0.727256	0.0	
3	0.366667	0.0	0.263679	1.0	0.000000	0.050134	0.073313	0.0	
4	0.233333	0.0	0.702698	0.0	0.500000	0.383690	0.664091	0.0	
...	...	...	...	...	...	...	...	...	
1267	0.939675	1.0	0.788314	0.0	0.000000	0.380175	0.664243	0.0	
1268	0.791359	1.0	0.873269	1.0	0.000000	0.417423	0.314809	0.0	
1269	0.723926	1.0	0.918265	1.0	0.000000	0.645312	0.390853	0.0	
1270	0.655971	1.0	0.849150	1.0	0.438262	0.480406	0.457323	0.0	
1271	0.588416	1.0	0.454688	1.0	0.500000	0.145078	0.522572	0.0	

1272 rows × 9 columns

```
X = df10.drop('Diagnosis', axis=1)
y = df10['Diagnosis']


X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, random_state=30)


models_output = []
for model in models_arr:
  print(model.__class__.__name__)
  acc_test, acc_train, cm, y_pred_proba, y_pred=  runModel(model, X_train, X_test, y_train, y_test)
  models_output.append([model.__class__.__name__, acc_test, acc_train, cm, y_pred_proba,y_pred])
```

```
LogisticRegression
Train Accuracy: 0.8382022471910112
Test Accuracy: 0.7984293193717278
[[154  42]
 [ 35 151]]
DecisionTreeClassifier
Train Accuracy: 1.0
Test Accuracy: 0.8429319371727748
[[163  33]
 [ 27 159]]
XGBClassifier
Train Accuracy: 1.0
Test Accuracy: 0.9162303664921466
[[177  19]
 [ 13 173]]
KNeighborsClassifier
Train Accuracy: 0.9325842696629213
Test Accuracy: 0.8455497382198953
[[156  40]
 [ 19 167]]
RandomForestClassifier
Train Accuracy: 1.0
Test Accuracy: 0.9162303664921466
[[178  18]
 [ 14 172]]
```

```
models_output[0][4]
```

```
array([[0.08250795, 0.91749205],
       [0.5532784 , 0.4467216 ],
       [0.15906324, 0.84093676],
       [0.80382817, 0.19617183],
       [0.63745919, 0.36254081],
       [0.03626903, 0.96373097],
       [0.26460946, 0.73539054],
       [0.12934668, 0.87065332],
       [0.43757034, 0.56242966],
       [0.74710264, 0.25289736],
       [0.76903772, 0.23096228],
       [0.00993796, 0.99006204],
       [0.04173702, 0.95826298],
       [0.95385968, 0.04614032],
       [0.23040789, 0.76959211],
       [0.85200692, 0.14799308],
       [0.34019043, 0.65980957],
       [0.00505133, 0.99494867],
       [0.85735645, 0.14264355],
       [0.99244984, 0.00755016],
       [0.25227534, 0.74772466],
       [0.01982659, 0.98017341],
       [0.62329832, 0.37670168],
       [0.8917382 , 0.1082618 ],
       [0.75075133, 0.24924867],
       [0.98026511, 0.01973489],
       [0.21613548, 0.78386452],
       [0.49521463, 0.50478537],
       [0.66517268, 0.33482732],
       [0.16282448, 0.83717552],
       [0.26597458, 0.73402542],
       [0.59648065, 0.40351935],
       [0.88804344, 0.11195656],
       [0.62778155, 0.37221845],
       [0.62586105, 0.37413895],
       [0.07314287, 0.92685713],
       [0.48634561, 0.51365439],
       [0.13640746, 0.86359254],
       [0.24378076, 0.75621924],
```

```
            [0.38982758, 0.61017242],
            [0.01246576, 0.98753424],
            [0.33990659, 0.66009341],
            [0.63590211, 0.36409789],
            [0.22664939, 0.77335061],
            [0.56968853, 0.43031147],
            [0.23253301, 0.76746699],
            [0.11332895, 0.88667105],
            [0.46032053, 0.53967947],
            [0.38209659, 0.61790341],
            [0.57061796, 0.42938204],
            [0.22767756, 0.77232244],
            [0.96174666, 0.03825334],
            [0.88858149, 0.11141851],
            [0.66089528, 0.33910472],
            [0.19917478, 0.80082522],
            [0.20908759, 0.79091241],
            [0.59812361, 0.40187639],
            [0.07480476, 0.92519524],
```

```
models_output[0][5]
```

```
array([1., 0., 1., 0., 0., 1., 1., 1., 1., 0., 0., 1., 1., 0., 1., 0., 1.,
       1., 0., 0., 1., 1., 0., 0., 0., 0., 1., 1., 0., 1., 1., 0., 0., 0.,
       0., 1., 1., 1., 1., 1., 1., 1., 0., 1., 0., 1., 1., 1., 1., 0., 1.,
       0., 0., 0., 1., 1., 0., 1., 0., 0., 1., 0., 0., 1., 1., 0., 1., 0.,
       0., 1., 1., 1., 1., 0., 0., 1., 0., 0., 1., 1., 1., 0., 0., 0., 0.,
       1., 0., 0., 0., 1., 0., 0., 0., 1., 1., 1., 0., 0., 0., 1., 1., 0.,
       1., 0., 1., 0., 0., 1., 0., 1., 1., 1., 0., 1., 0., 1., 0., 1., 0.,
       0., 0., 1., 0., 1., 0., 0., 0., 0., 0., 1., 1., 0., 0., 1., 0., 0.,
       1., 1., 1., 0., 0., 0., 0., 0., 0., 0., 0., 0., 1., 1., 0., 1., 0.,
       0., 0., 0., 1., 0., 1., 1., 1., 0., 0., 0., 0., 1., 0., 0., 1., 1.,
       0., 0., 0., 0., 1., 0., 1., 1., 0., 1., 0., 1., 0., 1., 1., 1., 0.,
       1., 0., 0., 0., 1., 1., 1., 0., 1., 1., 0., 1., 1., 0., 1., 0., 1.,
       1., 1., 1., 1., 0., 0., 0., 0., 0., 1., 1., 1., 1., 1., 1., 0., 0.,
       0., 1., 0., 0., 0., 0., 1., 0., 0., 0., 1., 1., 1., 0., 1., 0., 1.,
       0., 0., 0., 1., 1., 1., 0., 1., 1., 1., 1., 0., 1., 0., 1., 0., 0.,
       1., 0., 1., 0., 0., 1., 0., 0., 1., 1., 0., 1., 0., 0., 1., 1., 0.,
       0., 0., 0., 0., 1., 0., 1., 0., 1., 0., 1., 1., 1., 0., 0., 0., 1.,
       0., 1., 0., 0., 1., 0., 0., 1., 1., 0., 0., 0., 1., 1., 1., 0., 1.,
       0., 0., 1., 1., 1., 1., 0., 0., 0., 1., 0., 1., 1., 0., 0., 1., 1.,
       1., 1., 1., 0., 0., 1., 1., 1., 0., 0., 0., 1., 1., 1., 1., 1., 0.,
       0., 1., 1., 0., 1., 1., 0., 0., 1., 0., 0., 1., 0., 0., 1., 1., 1.,
       0., 1., 1., 1., 1., 1., 0., 0., 1., 0., 0., 0., 1., 1., 1., 1., 1.,
       1., 1., 1., 1., 1., 0., 1., 1.])
```

```
df_y_pred = pd.DataFrame(models_output[2][5], columns=['Prediction'])
df_y_pred
```

	Prediction
0	1
1	1

```
df_pred_proba = pd.DataFrame(models_output[2][4], columns=['0', '1'])
df_pred_proba
```

	0	1
0	0.000096	0.999904
1	0.016027	0.983973
2	0.042034	0.957966
3	0.977587	0.022413
4	0.078924	0.921076
...	...	...
377	0.018323	0.981677
378	0.500966	0.499034
379	0.987401	0.012599
380	0.003845	0.996155
381	0.054930	0.945070

382 rows × 2 columns

```
df_both = pd.concat([df_y_pred, df_pred_proba], axis=1)
df_both
```

	Prediction	0	1
0	1	0.000096	0.999904
1	1	0.016027	0.983973