```
df.head()
```

```
   PassengerId  Survived  Pclass  \
0            1         0       3
1            2         1       1
2            3         1       3
3            4         1       1
4            5         0       3

                                                Name     Sex   Age
SibSp  \
0                            Braund, Mr. Owen Harris    male  22.0
1
1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0
1
2                             Heikkinen, Miss. Laina  female  26.0
0
3       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0
1
4                           Allen, Mr. William Henry    male  35.0
0

   Parch            Ticket     Fare Cabin Embarked
0      0         A/5 21171   7.2500   NaN        S
1      0          PC 17599  71.2833   C85        C
2      0  STON/O2. 3101282   7.9250   NaN        S
3      0            113803  53.1000  C123        S
4      0            373450   8.0500   NaN        S
```

```
df = pd.get_dummies(df, columns=['Pclass'], prefix='Pclass')
df = pd.get_dummies(df, columns=['Sex'], prefix='Sex')
df = pd.get_dummies(df, columns=['Embarked'], prefix='Embarked')

df.head()
```

```
   PassengerId  Survived
Name  \
0            1         0                            Braund, Mr. Owen
Harris
1            2         1  Cumings, Mrs. John Bradley (Florence Briggs
Th...
2            3         1                             Heikkinen, Miss.
Laina
3            4         1       Futrelle, Mrs. Jacques Heath (Lily May
Peel)
4            5         0                           Allen, Mr. William
Henry

    Age  SibSp  Parch            Ticket     Fare Cabin  Pclass_1
Pclass_2  \
```

```
0  22.0      1      0         A/5 21171   7.2500   NaN     False
False
1  38.0      1      0          PC 17599  71.2833   C85      True
False
2  26.0      0      0  STON/O2. 3101282   7.9250   NaN     False
False
3  35.0      1      0            113803  53.1000  C123      True
False
4  35.0      0      0            373450   8.0500   NaN     False
False
```

| | Pclass_3 | Sex_female | Sex_male | Embarked_C | Embarked_Q | Embarked_S |
|---|---|---|---|---|---|---|
| 0 | True | False | True | False | False | True |
| 1 | False | True | False | True | False | False |
| 2 | True | True | False | False | False | True |
| 3 | False | True | False | False | False | True |
| 4 | True | False | True | False | False | True |

```
df['Family']=df['SibSp']+df['Parch']|+1
df.drop(['SibSp','Parch'],axis=1,inplace=True)
df.head()
```

```
   PassengerId  Survived
Name  \
0            1         0                            Braund, Mr. Owen
Harris
1            2         1  Cumings, Mrs. John Bradley (Florence Briggs
Th...
2            3         1                             Heikkinen, Miss.
Laina
3            4         1       Futrelle, Mrs. Jacques Heath (Lily May
Peel)
4            5         0                           Allen, Mr. William
Henry

    Age            Ticket     Fare Cabin  Pclass_1  Pclass_2  Pclass_3
\
0  22.0         A/5 21171   7.2500   NaN     False     False      True

1  38.0          PC 17599  71.2833   C85      True     False     False

2  26.0  STON/O2. 3101282   7.9250   NaN     False     False      True

3  35.0            113803  53.1000  C123      True     False     False
```

```
4  35.0            373450   8.0500   NaN     False     False      True


   Sex_female  Sex_male  Embarked_C  Embarked_Q  Embarked_S  Family
0       False      True       False       False        True       1
1        True     False        True       False       False       1
2        True     False       False       False        True       1
3        True     False       False       False        True       1
4       False      True       False       False        True       1
```

```
df.drop(['Name','Ticket','Fare','Cabin'],axis=1,inplace=True)
df.head()
```

```
   PassengerId  Survived   Age  Pclass_1  Pclass_2  Pclass_3
Sex_female  \
0            1         0  22.0     False     False      True
False
1            2         1  38.0      True     False     False
True
2            3         1  26.0     False     False      True
True
3            4         1  35.0      True     False     False
True
4            5         0  35.0     False     False      True
False

   Sex_male  Embarked_C  Embarked_Q  Embarked_S  Family
0      True       False       False        True       1
1     False        True       False       False       1
2     False       False       False        True       1
3     False       False       False        True       1
4      True       False       False        True       1
```

```
df.isnull().sum()
```

```
PassengerId      0
Survived         0
Age            177
Pclass_1         0
Pclass_2         0
Pclass_3         0
Sex_female       0
Sex_male         0
Embarked_C       0
Embarked_Q       0
Embarked_S       0
Family           0
dtype: int64
```

```
df['Age'].fillna(df['Age'].median(),inplace=True)
```

```
df.isnull().sum()
```

```
PassengerId    0
Survived       0
Age            0
Pclass_1       0
Pclass_2       0
Pclass_3       0
Sex_female     0
Sex_male       0
Embarked_C     0
Embarked_Q     0
Embarked_S     0
Family         0
dtype: int64
```

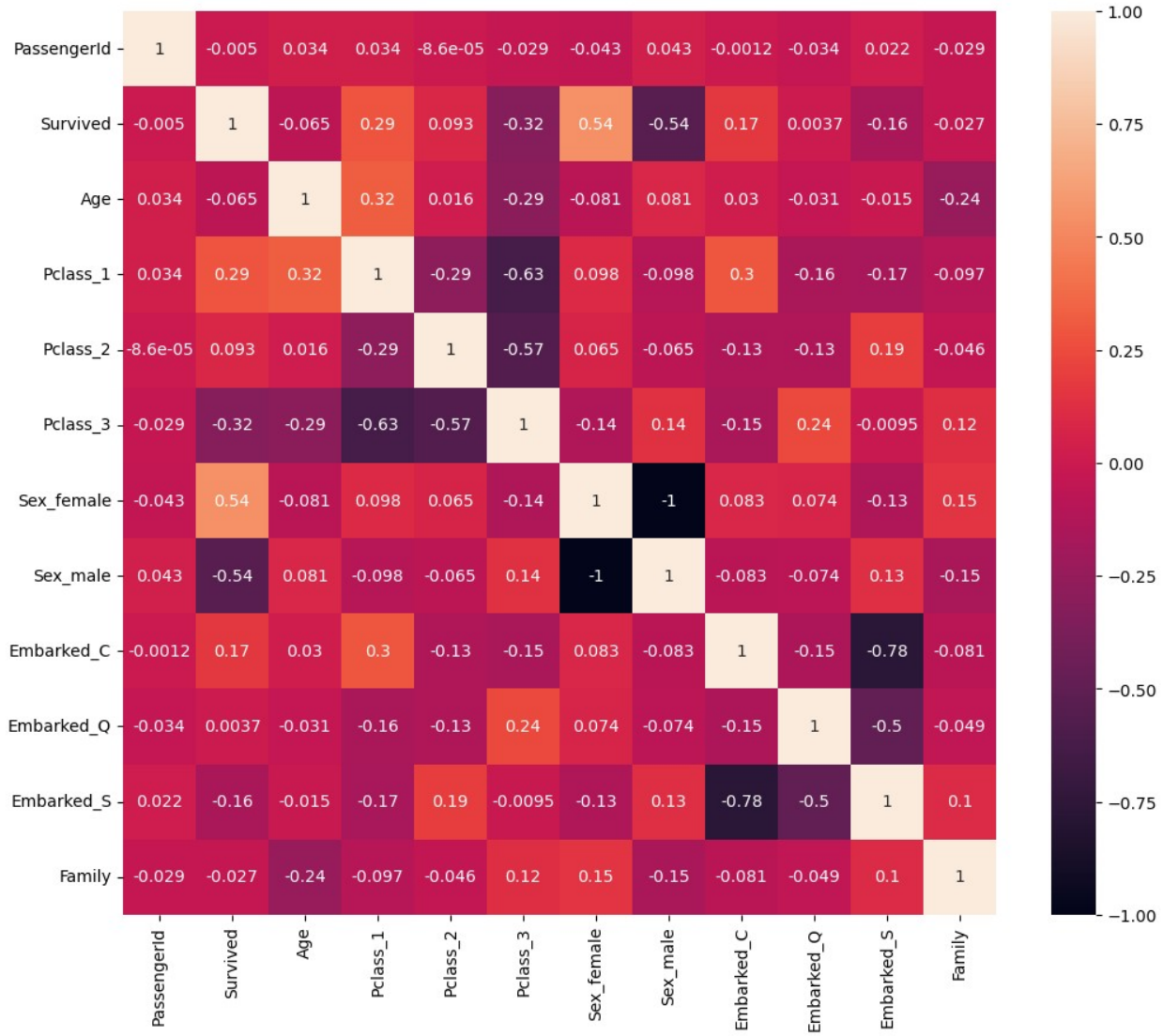
```
df.head()
```

```
   PassengerId  Survived   Age  Pclass_1  Pclass_2  Pclass_3
Sex_female  \
0            1         0  22.0     False     False      True
False
1            2         1  38.0      True     False     False
True
2            3         1  26.0     False     False      True
True
3            4         1  35.0      True     False     False
True
4            5         0  35.0     False     False      True
False

   Sex_male  Embarked_C  Embarked_Q  Embarked_S  Family
0      True       False       False        True       1
1     False        True       False       False       1
2     False       False       False        True       1
3     False       False       False        True       1
4      True       False       False        True       1
```

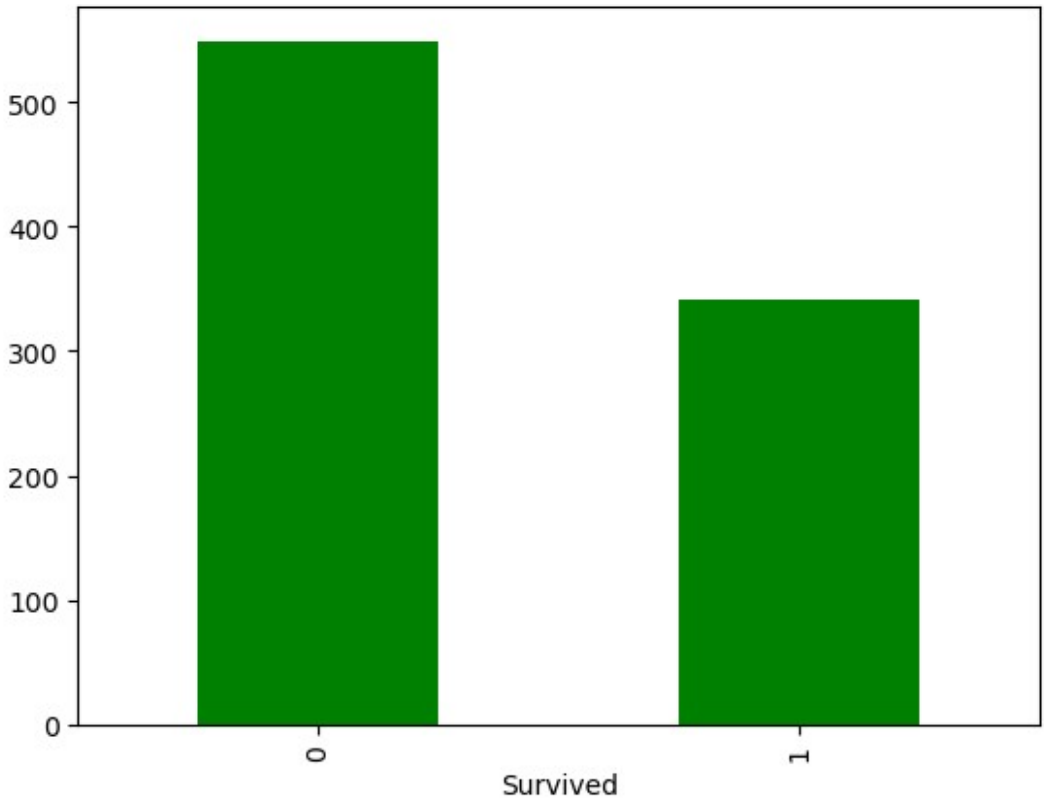
```
plt.figure(figsize=(12, 10))
sns.heatmap(df.corr(),annot=True)
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```




```
df['Survived'].value_counts().plot(kind='bar', color='green')
```

```
<Axes: xlabel='Survived'>
```




```
df.columns
```

```
Index(['PassengerId', 'Survived', 'Age', 'Pclass_1', 'Pclass_2',
'Pclass_3',
       'Sex_female', 'Sex_male', 'Embarked_C', 'Embarked_Q',
'Embarked_S',
       'Family'],
      dtype='object')
```

```
x=df[['PassengerId', 'Age', 'Pclass_1', 'Pclass_2', 'Pclass_3',
       'Sex_female', 'Sex_male', 'Embarked_C', 'Embarked_Q',
'Embarked_S',
       'Family']]
y=df['Survived']

#x=pd.DataFrame(x)
#x

from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,rando
m_state=42)

from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
```

```
x_train=scaler.fit_transform(x_train)
x_test=scaler.transform(x_test)
```

# KNN Algorithm

```
from sklearn.metrics import confusion_matrix, accuracy_score
from sklearn.neighbors import KNeighborsClassifier

best_accuracy = 0
best_k = 1
for k in range(1, 26):
    neigh = KNeighborsClassifier(n_neighbors=k)
    neigh.fit(x_train, y_train)
    y_pred = neigh.predict(x_test)
    accuracy = accuracy_score(y_test, y_pred)
    if accuracy > best_accuracy:
        best_accuracy = accuracy
        best_k = k

best_k
```

```
10
```

```
neigh=KNeighborsClassifier(n_neighbors=13)
neigh.fit(x_train,y_train)
```

```
KNeighborsClassifier(n_neighbors=13)
```

```
y_pred=neigh.predict(x_test)
y_pred=list(y_pred)
y_test=list(y_test)

cm=confusion_matrix(y_test,y_pred)
cm
```

```
array([[142,  15],
       [ 40,  71]], dtype=int64)
```

```
score=accuracy_score(y_test,y_pred)
score
```

```
0.7947761194029851
```

```
for i in range(len(y_test)):
    if y_pred[i]!=y_test[i]:
        print("prediction:",y_pred[i]," test: ",y_test[i])
```

```
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
```

```
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 1  test:  0
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 1  test:  0
prediction: 1  test:  0
prediction: 1  test:  0
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 1  test:  0
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
prediction: 0  test:  1
```

```
prediction: 1  test:  0
prediction: 0  test:  1
prediction: 0  test:  1
```

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score

for i in range(1, 26):
    neigh = KNeighborsClassifier(n_neighbors=i, weights='uniform')
    neigh.fit(x_train, y_train)

    # Use the previously trained model to make predictions on the
training set
    y_pred_train = neigh.predict(x_train)
    y_pred_test = neigh.predict(x_test)

    score_test = accuracy_score(y_pred_test, y_test)
    print('for', i, 'test accuracy is ', score_test)

    score_train = accuracy_score(y_pred_train, y_train)
    print('train accuracy is ', score_train)
```

```
for 1 test accuracy is  0.7574626865671642
train accuracy is  1.0
for 2 test accuracy is  0.7835820895522388
train accuracy is  0.8635634028892456
for 3 test accuracy is  0.8022388059701493
train accuracy is  0.8747993579454254
for 4 test accuracy is  0.7985074626865671
train accuracy is  0.8378812199036918
for 5 test accuracy is  0.7835820895522388
train accuracy is  0.8459069020866774
for 6 test accuracy is  0.7798507462686567
train accuracy is  0.8459069020866774
for 7 test accuracy is  0.7873134328358209
train accuracy is  0.8394863563402889
for 8 test accuracy is  0.7985074626865671
train accuracy is  0.8314606741573034
for 9 test accuracy is  0.7985074626865671
train accuracy is  0.8330658105939005
for 10 test accuracy is  0.8059701492537313
train accuracy is  0.8314606741573034
for 11 test accuracy is  0.7910447761194029
train accuracy is  0.8314606741573034
for 12 test accuracy is  0.7947761194029851
train accuracy is  0.8218298555377207
for 13 test accuracy is  0.7947761194029851
train accuracy is  0.826645264847512
for 14 test accuracy is  0.8022388059701493
train accuracy is  0.8250401284109149
```

```
for 15 test accuracy is  0.8059701492537313
train accuracy is  0.8250401284109149
for 16 test accuracy is  0.7985074626865671
train accuracy is  0.8138041733547352
for 17 test accuracy is  0.7910447761194029
train accuracy is  0.8186195826645265
for 18 test accuracy is  0.7910447761194029
train accuracy is  0.8170144462279294
for 19 test accuracy is  0.7985074626865671
train accuracy is  0.8202247191011236
for 20 test accuracy is  0.7985074626865671
train accuracy is  0.8154093097913323
for 21 test accuracy is  0.7947761194029851
train accuracy is  0.8202247191011236
for 22 test accuracy is  0.7873134328358209
train accuracy is  0.812199036918138
for 23 test accuracy is  0.7910447761194029
train accuracy is  0.8154093097913323
for 24 test accuracy is  0.7985074626865671
train accuracy is  0.8138041733547352
for 25 test accuracy is  0.7873134328358209
train accuracy is  0.812199036918138
```

```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import GridSearchCV
parameters = {'n_neighbors': [5,10,15,20,26], 'weights': ['uniform',
'distance'], 'p': [1, 2, 3, 4, 5]}

classifier = KNeighborsClassifier()
clf = GridSearchCV(classifier, parameters)

clf.fit(x_train, y_train)
```

```
GridSearchCV(estimator=KNeighborsClassifier(),
             param_grid={'n_neighbors': [5, 10, 15, 20, 26],
                         'p': [1, 2, 3, 4, 5],
                         'weights': ['uniform', 'distance']})
```

```
clf.best_params_
```

```
{'n_neighbors': 20, 'p': 1, 'weights': 'uniform'}
```

```
new=list(x_test)
new
```

```
[array([ 1.01391396, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([-0.02533086,  0.15305586, -0.53590119,  1.92865159, -
```

```
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.51814016, -0.69140812, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.05625357, -1.76618045, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.56495282, -1.15202483, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.59884004, -0.2307914 ,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.56034949, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-0.43333068, -0.99848592, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.91446254, -0.99848592, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-1.19159449, -0.76817757,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-1.18774544,  0.61367258,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
```

```
 array([ 0.96387625,  1.15105874, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.15172567, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([-0.77589656,  0.07628641, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.39099107,  0.53690312, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.56817787, -0.99848592,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.67519713,  0.99751984,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.79836689, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([ 1.69519668, -0.15402195, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.29166992,  1.3813671 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.58344382, -0.38433031, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
```

```
        -0.43379723]),
 array([ 0.00546158,  0.38336422,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.97604742, -0.76817757, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.90998948, -0.69140812, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.35572558, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.43731   , -1.45910264, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([-1.59959431,  0.84398093,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.87534799,  0.15305586, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.4725755 , -1.91971935, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([-0.19084022,  0.15305586, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.66055519, -0.76817757, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
```

```
0.60269272,
        -0.43379723]),
 array([-1.17234922, -0.53786921, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.33723457, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.81053806, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.92985876, -0.84494702, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.67134808, -0.15402195, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-1.6265377 , -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.95295309, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([ 1.52583826,  0.07628641,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 1.12938561, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.25317937, -0.61463866, -0.53590119,  1.92865159, -
1.13968987,
```

```
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 1.20933227e-01, -4.83043745e-04, -5.35901187e-01, -
5.18496966e-01,
         8.77431686e-01, -7.20771944e-01,  7.20771944e-01, -
4.63060419e-01,
        -3.11176776e-01,  6.02692718e-01, -4.33797232e-01]),
 array([ 1.50659299, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.32568741, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-0.32170809,  1.22782819, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-1.38404724, -0.99848592, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([-1.45717928, -0.76817757, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.36789674, -0.46109976, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.90676443, -0.38433031, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.82146122,  2.22583108,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([ 0.00931064, -1.8429499 , -0.53590119, -0.51849697,
```

```
0.87743169,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([ 1.44115906,  1.76521436,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.3294062 ,  0.84398093, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([ 1.37187607, -1.38233319,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.63733059,  2.76321724, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([ 0.53663116, -0.84494702,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-1.44563212,  0.22982531, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.56419854,  1.61167546,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.90291538,  0.46013367, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.54570754, -0.76817757, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
```

```
 array([-0.14465156, -0.61463866, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.61252541e-03, -1.22879428e+00, -5.35901187e-01,
1.92865159e+00,
        -1.13968987e+00,  1.38740139e+00, -1.38740139e+00, -
4.63060419e-01,
        -3.11176776e-01,  6.02692718e-01, -4.33797232e-01]),
 array([-0.07536857, -0.0772525 , -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.06842474, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.9606512 , -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([ 1.25640442,  2.14906162, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.43793401, -0.61463866, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-9.07647921e-02, -4.83043745e-04, -5.35901187e-01, -
5.18496966e-01,
         8.77431686e-01, -7.20771944e-01,  7.20771944e-01, -
4.63060419e-01,
        -3.11176776e-01,  6.02692718e-01, -4.33797232e-01]),
 array([ 0.33263125, -0.92171647, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([ 0.55972549,  1.76521436,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
```

```
        -0.43379723]),
 array([-0.51416083, -0.2307914 , -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.51800989, -0.84494702,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
         2.01195549]),
 array([ 1.70674385, -0.2307914 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 1.49889488,  0.76721148,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([ 1.49119677, -0.46109976, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.61361225,  0.53690312, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.23715915,  0.11467113, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([ 0.35957463, -0.53786921,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([-0.34095336,  0.69044203, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.73740601,  0.92075038, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
```

```
0.60269272,
         0.78907913]),
 array([-0.69121736, -0.99848592, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([-1.46102834e+00, -4.83043745e-04, -5.35901187e-01,
1.92865159e+00,
        -1.13968987e+00,  1.38740139e+00, -1.38740139e+00, -
4.63060419e-01,
        -3.11176776e-01,  6.02692718e-01, -4.33797232e-01]),
 array([-0.48721745,  0.84398093,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([ 1.27949875, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-1.04533041,  2.07229217,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.18251811,  3.22383396,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.88751916,  0.15305586,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.525708  ,  0.07628641,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 1.44885717,  0.69044203,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.75280223, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
```

```
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.0799719 , -1.53587209, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 0.36727274, -1.53587209, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         3.23483185]),
 array([ 0.69059335,  2.45613943,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.22163266, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-1.42253779, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.63348153, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.02546113,  1.45813655,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.4140854 ,  1.22782819, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.45642501,  0.53690312, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.47951934, -0.15402195, -0.53590119, -0.51849697,
0.87743169,
```

```
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.38266896,  2.68644779,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.55587643,  0.46013367, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.59499098, -0.76817757,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.07474457,  0.69044203, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.41806473,  0.11467113, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.01378369, -0.30756085, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.86380082,  0.84398093, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 0.0631974 ,  1.3813671 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.61423626,  0.07628641, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.91061349,  0.84398093,  1.86601565, -0.51849697, -
```

```
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.85995177, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.8413305 , -0.65302339, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.55278165, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.25565015,  0.53690312,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.31338597, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.65670613, -0.69140812, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.34555669,  3.22383396,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.59114193,  0.53690312, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.47182123, -0.53786921, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
```

```
 array([ 1.11783845,  1.45813655, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([ 0.32108408, -0.46109976, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.87597199,  0.22982531,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.65657586,  2.60967834,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.41483968, -2.16307852, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 1.30644214, -0.30756085, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.6097632 ,  0.46013367,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 1.34878174,  1.534906  ,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.18004733, -0.38433031,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.50968777, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
```

```
        -0.43379723]),
 array([ 0.18636716, -0.61463866, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.33325525, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-1.46487739, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([ 0.13632945, -0.0772525 , -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.2756497 , -1.8429499 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.59051792,  1.534906  ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 1.47964961, -1.07525538, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.53725516, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-8.25934278e-01, -4.83043745e-04, -5.35901187e-01, -
5.18496966e-01,
         8.77431686e-01, -7.20771944e-01,  7.20771944e-01, -
4.63060419e-01,
        -3.11176776e-01,  6.02692718e-01, -4.33797232e-01]),
 array([ 1.53738543,  0.99751984, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
```

```
0.60269272,
        -0.43379723]),
 array([-1.52646227, -0.84494702, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.79451783, -0.61463866, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.57202692,  1.15105874, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.39868918, -0.0772525 , -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.57127265,  0.61367258, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.69134763,  0.76721148, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
         2.01195549]),
 array([-0.11385912,  0.38336422, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.89521727,  0.07628641, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.58804715,  2.83998669, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.59574526, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
```

```
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.85610271,  1.07428929, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.5327821 , -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.26334826, -0.38433031, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.93308381, -0.84494702, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.29551898, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-1.41868873, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.63670658, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.75987634, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.39944346, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([ 0.27874448,  0.53690312, -0.53590119,  1.92865159, -
```

```
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.81376311,  0.22982531, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
         0.78907913]),
 array([-0.57574571, -0.42271503, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.23640487, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.3949704 ,  0.76721148, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.27167037, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-1.69582068, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-1.50721699,  2.76321724,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.18314211, -0.46109976, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.04395213, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
```

```
 array([ 0.29029164, -0.53786921, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.31785903,  0.46013367, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.34170763, -0.46109976,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.11001007, -0.84494702, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 0.48659344,  0.53690312,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.56432881, -1.53587209, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([-0.04842519, -0.92171647, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.26025348, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-1.61883959,  0.69044203, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         3.23483185]),
 array([-1.39174535, -0.92171647, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
```

```
        -0.43379723]),
 array([-1.67657541, -1.91971935, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.44102879,  1.26621292,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.10629128, -0.46109976, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.24472699,  0.22982531, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.99466869, -0.2307914 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.41421567, -1.76618045, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         3.23483185]),
 array([-1.26087748, -0.38433031,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.34802747,  1.22782819,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-3.25557141e-01, -4.83043745e-04, -5.35901187e-01,
1.92865159e+00,
        -1.13968987e+00, -7.20771944e-01,  7.20771944e-01,
2.15954541e+00,
        -3.11176776e-01, -1.65922031e+00, -4.33797232e-01]),
 array([-1.60344337, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
```

```
0.60269272,
        -0.43379723]),
 array([-1.50336794, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.61436653,  0.99751984, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.57512171,  0.53690312, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.79514184,  0.30659477, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.06010262, -0.92171647, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.40329251e+00, -4.83043745e-04, -5.35901187e-01, -
5.18496966e-01,
         8.77431686e-01, -7.20771944e-01,  7.20771944e-01, -
4.63060419e-01,
        -3.11176776e-01,  6.02692718e-01, -4.33797232e-01]),
 array([-0.71816074,  1.61167546, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.15557472,  0.46013367,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.04085735,  0.69044203,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 1.36417796,  0.38336422, -0.53590119,  1.92865159, -
1.13968987,
```

```
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.29104592, -0.92171647,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.3711218 , -1.38233319, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         3.23483185]),
 array([-0.46027406,  2.45613943, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.34032936,  0.07628641, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.34417841, -1.689411  , -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 0.14402756,  2.60967834, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.21791387, -0.69140812, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.3364803 , -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([ 1.02931018e+00, -4.83043745e-04, -5.35901187e-01, -
5.18496966e-01,
         8.77431686e-01, -7.20771944e-01,  7.20771944e-01, -
4.63060419e-01,
        -3.11176776e-01,  6.02692718e-01, -4.33797232e-01]),
 array([-0.21393455,  0.53690312,  1.86601565, -0.51849697, -
1.13968987,
```

```
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 0.19021622, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.1401785 ,  1.61167546, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.26397226, -0.15402195,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913]),
 array([-0.31016092,  0.07628641, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.12925534,  0.30659477,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([ 0.08629173, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-0.14080251, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([-1.68812257, -2.07325826, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         2.01195549]),
 array([ 0.51738588, -0.30756085, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-1.11846245,  1.68844491,  1.86601565, -0.51849697, -
```

```
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 1.67210235, -0.30756085, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.07921763, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.32631141, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.94910403, -0.38433031, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.08306668, -0.84494702, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.5680476 , -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.68749857, -0.30756085, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.7643494 , -0.38433031, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 0.12478228, -0.53786921, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
```

```
 array([-0.54110422, -2.15616927,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.49876461, -0.38433031, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.70736785, -0.2307914 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.15619873,  0.38336422, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.14093278, -0.61463866,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
         2.01195549]),
 array([-0.71431169, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-0.30246281, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-1.10691529, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.41731045, -0.53786921, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.47580055,  2.53290888,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678, -
1.65922031,
```

```
        -0.43379723]),
 array([ 0.89074421, -0.84494702, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.2524251 ,  0.99751984,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.69444241,  2.14906162, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-0.55265138, -0.76817757, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.05227424,  0.99751984, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.2941407 , -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.16004778, -0.69140812, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.52968732,  0.42174894, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.34865147, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.07089551, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
```

```
0.60269272,
        -0.43379723]),
 array([ 1.18327238, -0.46109976, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.89906632, -0.53786921, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.01315969,  1.76521436,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.41099062,  0.07628641, -0.53590119, -0.51849697,
0.87743169,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.14540583,  0.57528785, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([-0.06767046, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042,  3.21360743, -
1.65922031,
        -0.43379723]),
 array([-1.48412267, -0.53786921, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([ 0.82531027,  1.61167546,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
         0.78907913]),
 array([ 1.37957418, -0.15402195, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.54817832, -0.53786921, -0.53590119, -0.51849697,
0.87743169,
```

```
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.22946104,  1.68844491,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.69506641,  0.53690312, -0.53590119,  1.92865159, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-1.20314166e+00, -4.83043745e-04, -5.35901187e-01,
1.92865159e+00,
        -1.13968987e+00,  1.38740139e+00, -1.38740139e+00, -
4.63060419e-01,
        -3.11176776e-01,  6.02692718e-01, -4.33797232e-01]),
 array([ 1.01006491, -0.53786921,  1.86601565, -0.51849697, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.37881991,  1.61167546,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
        -0.43379723]),
 array([-0.38329296,  0.84398093, -0.53590119,  1.92865159, -
1.13968987,
         1.38740139, -1.38740139, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 1.44500811, -0.15402195, -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.72138579, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.04010307,  2.76321724,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([ 0.20946149, -0.92171647, -0.53590119, -0.51849697,
```

```
0.87743169,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723]),
 array([-0.05997235, -0.0772525 ,  1.86601565, -0.51849697, -
1.13968987,
        -0.72077194,  0.72077194, -0.46306042, -0.31117678,
0.60269272,
        -0.43379723])]

input_data=( 1.01391396, -0.0772525 , -0.53590119, -0.51849697,
0.87743169,
        -0.72077194,  0.72077194,  2.15954541, -0.31117678, -
1.65922031,
         0.78907913)
input_data_array = np.asarray(input_data)

input_data_array_reshaped = input_data_array.reshape(1,-1)

pred =neigh.predict(input_data_array_reshaped)

print(pred)

if (pred[0] ==0):
    print('THE PASSANGER HAS NOT SURVIVED')
else:
    print('THE PASSANGER HAS SURVIED')



[0]
THE PASSANGER HAS NOT SURVIVED
```

## Decision Tree Classifier

```
from sklearn.tree import DecisionTreeClassifier

deci=DecisionTreeClassifier()
deci.fit(x_train,y_train)

DecisionTreeClassifier()

y_pred=deci.predict(x_test)

from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score

score=accuracy_score(y_train,deci.predict(x_train))
print('train accuracy score is : ',score)
f1=f1_score(y_train,deci.predict(x_train))
print('train f1 score : ',f1)
```

```
train accuracy score is :  1.0
train f1 score :  1.0
```

```
score=accuracy_score(y_test,y_pred)
print('test accuracy score is : ',score)
f1=f1_score(y_test,y_pred)
print('test f1 score : ',f1)
```

```
test accuracy score is :  0.7761194029850746
test f1 score :  0.7058823529411764
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

best_accuracy_dt = 0
best_depth_dt = 1

for depth in range(1, 26):
    dt_classifier = DecisionTreeClassifier(max_depth=depth)
    dt_classifier.fit(x_train, y_train)
    y_pred_dt = dt_classifier.predict(x_test)
    accuracy_dt = accuracy_score(y_test, y_pred_dt)

    if accuracy_dt > best_accuracy_dt:
        best_accuracy_dt = accuracy_dt
        best_depth_dt = depth

best_depth_dt
```

```
3
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

for depth in range(1, 26):
    dt_classifier = DecisionTreeClassifier(max_depth=depth)
    dt_classifier.fit(x_train, y_train)

    # Use the previously trained model to make predictions on the
training set
    y_pred_train = dt_classifier.predict(x_train)
    y_pred_test = dt_classifier.predict(x_test)

    score_test = accuracy_score(y_pred_test, y_test)
    print('for', depth, 'test accuracy is ', score_test)

    score_train = accuracy_score(y_pred_train, y_train)
    print('train accuracy is ', score_train)
```

```
for 1 test accuracy is  0.7910447761194029
train accuracy is  0.7849117174959872
```

```
for 2 test accuracy is  0.7723880597014925
train accuracy is  0.8057784911717496
for 3 test accuracy is  0.8208955223880597
train accuracy is  0.8298555377207063
for 4 test accuracy is  0.8097014925373134
train accuracy is  0.8394863563402889
for 5 test accuracy is  0.8208955223880597
train accuracy is  0.8603531300160514
for 6 test accuracy is  0.8097014925373134
train accuracy is  0.8731942215088283
for 7 test accuracy is  0.8059701492537313
train accuracy is  0.884430176565008
for 8 test accuracy is  0.7985074626865671
train accuracy is  0.9020866773675762
for 9 test accuracy is  0.7835820895522388
train accuracy is  0.913322632423756
for 10 test accuracy is  0.7835820895522388
train accuracy is  0.9325842696629213
for 11 test accuracy is  0.7910447761194029
train accuracy is  0.9470304975922953
for 12 test accuracy is  0.753731343283582
train accuracy is  0.9598715890850722
for 13 test accuracy is  0.75
train accuracy is  0.9727126805778491
for 14 test accuracy is  0.753731343283582
train accuracy is  0.985553772070626
for 15 test accuracy is  0.753731343283582
train accuracy is  0.9903691813804173
for 16 test accuracy is  0.7723880597014925
train accuracy is  0.9983948635634029
for 17 test accuracy is  0.7723880597014925
train accuracy is  1.0
for 18 test accuracy is  0.75
train accuracy is  1.0
for 19 test accuracy is  0.7723880597014925
train accuracy is  1.0
for 20 test accuracy is  0.7723880597014925
train accuracy is  1.0
for 21 test accuracy is  0.753731343283582
train accuracy is  1.0
for 22 test accuracy is  0.753731343283582
train accuracy is  1.0
for 23 test accuracy is  0.75
train accuracy is  1.0
for 24 test accuracy is  0.7649253731343284
train accuracy is  1.0
for 25 test accuracy is  0.7574626865671642
train accuracy is  1.0
```

```
from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier

parameters = {'max_depth': [3, 5, 10, 15, 20], 'min_samples_split':
[2, 5, 10], 'max_leaf_nodes': [15,20,25,30]}

dt_classifier = DecisionTreeClassifier()


grid_search = GridSearchCV(dt_classifier, parameters, cv=5)

grid_search.fit(x_train, y_train)


print("Best Hyperparameters:", grid_search.best_params_)
best_dt_model = grid_search.best_estimator_

y_pred_test = best_dt_model.predict(x_test)
```

```
Best Hyperparameters: {'max_depth': 3, 'max_leaf_nodes': 15,
'min_samples_split': 2}
```

```
prune=DecisionTreeClassifier(max_depth=3, max_leaf_nodes=25,
random_state=101)
prune.fit(x_train,y_train)
```

```
DecisionTreeClassifier(max_depth=3, max_leaf_nodes=25,
random_state=101)
```

```
y_prune_pred=prune.predict(x_test)

score=accuracy_score(y_train,prune.predict(x_train))
print('train accuracy score is : ',score)
f1=f1_score(y_train,prune.predict(x_train))
print('train f1 score : ',f1)
```

```
train accuracy score is :  0.8298555377207063
train f1 score :  0.7568807339449543
```

```
score=accuracy_score(y_test,y_prune_pred)
print('test accuracy score is : ',score)
f1=f1_score(y_test,y_prune_pred)
print('test f1 score : ',f1)
```

```
test accuracy score is :  0.8208955223880597
test f1 score :  0.7669902912621359
```

```
input_data=(-1.726220,0.663861,1.767767,-0.510152,-1.107926,1.355574,-
1.355574,2.074505,-0.307562,-1.614710,-0.432344)
input_data_array = np.asarray(input_data)

input_data_array_reshaped = input_data_array.reshape(1,-1)
```

```
pred =deci.predict(input_data_array_reshaped)

print(pred)

if (pred[0] ==0):
    print('THE PASSANGER HAS NOT SURVIVED')
else:
    print('THE PASSANGER HAS SURVIED')



[1]
THE PASSANGER HAS SURVIED
```

## Random Forest

```
from sklearn.ensemble import RandomForestClassifier

rf_classifier = RandomForestClassifier(random_state=42)
rf_classifier.fit(x_train, y_train)

RandomForestClassifier(random_state=42)

y_pred = rf_classifier.predict(x_test)
y_pred

array([0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0,
0,
       0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0,
1,
       0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 1,
1,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
0,
       1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1,
0,
       0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0,
1,
       0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
0,
       0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1,
0,
       1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1,
0,
       0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0,
1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1,
```

```
0,
       0, 0, 0, 0], dtype=int64)
```

```
accuracy = accuracy_score(y_test, y_pred)
accuracy
```

```
0.8208955223880597
```

```
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4]
}
grid_search = GridSearchCV(estimator=rf_classifier,
param_grid=param_grid, cv=5, scoring='accuracy')
grid_search.fit(x_train, y_train)
```

```
GridSearchCV(cv=5, estimator=RandomForestClassifier(random_state=42),
             param_grid={'max_depth': [None, 10, 20, 30],
                         'min_samples_leaf': [1, 2, 4],
                         'min_samples_split': [2, 5, 10],
                         'n_estimators': [50, 100, 200]},
             scoring='accuracy')
```

```
best_params = grid_search.best_params_
best_params
```

```
{'max_depth': None,
 'min_samples_leaf': 4,
 'min_samples_split': 2,
 'n_estimators': 200}
```

```
best_rf_classifier = RandomForestClassifier(**best_params,
random_state=42)
best_rf_classifier.fit(x_train, y_train)
y_pred = best_rf_classifier.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
accuracy
```

```
0.8208955223880597
```

## Linear Regression

```
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import LinearRegression

linear_model = LinearRegression()

linear_model.fit(x_train, y_train)
```

```
LinearRegression()

y_pred = linear_model.predict(x_test)
y_pred

array([ 0.16112919,  0.28290512,  0.14168509,  0.91443049,
0.73867013,
        0.8845796 ,  0.64906567,  0.08323031,  0.70048549,
0.85948632,
        0.32587878,  0.02095677,  0.48408883,  0.17597859,
0.25420837,
        0.96231603,  0.32875994,  0.66831783,  0.32609637,
0.2773751 ,
        0.09309451,  0.35689512,  0.60966927,  0.13306798,
0.08758949,
        0.08833867,  0.41976351,  0.29566717,  0.07532611,
0.56435779,
        0.14840613,  0.59277845,  0.5178139 ,  0.5696864 ,
0.11640471,
        0.21059448,  0.3619894 ,  0.64350273,  1.0097022 ,
0.09855176,
        0.2629223 ,  0.07955951,  0.10389656,  0.17716749,
0.61048958,
        0.02014313,  0.1042298 ,  0.10085179,  0.08851326,
0.31979616,
        0.75368836,  0.74158801, -0.07778802,  0.43481186, -
0.02466247,
        0.88867731,  0.25807723,  0.88602541,  0.75027998,
0.6774318 ,
        0.11342126,  0.86656509,  0.7949276 ,  0.36989751,
0.14477151,
        0.67740766,  0.30992056,  0.07655989,  0.2032101 ,
0.89254448,
        0.74850029,  0.93794729,  0.53245584,  0.91737628,
0.12719397,
        0.05361867,  0.66277745,  0.98118577,  0.74413328,
0.45996603,
        0.02996009,  0.77059055,  0.8845312 ,  0.17651303,
0.35236725,
        0.29921837,  0.97080333,  0.98063308,  0.35853467,
0.07188235,
        0.10698961,  0.52688605,  0.23961892,  0.15524294,
0.06239262,
        0.07357305,  0.30550427, -0.00327119,  0.75190335,
0.08045771,
        0.22114648,  0.0575037 ,  1.03138565,  0.03657702,
0.09088438,
        0.09646312,  0.70218154,  0.29657271,  0.06443951,
0.42952593,
        0.89584353,  0.10590132,  1.0194902 ,  0.35103391,
```

```
0.3894769 ,
        0.14364848,  0.27756651,  0.36520206,  0.76791267,
0.40371575,
        0.27582239,  0.96626383,  0.70974629,  0.35549349,
0.11516989,
        0.47458429,  0.80400454,  0.50095806,  0.70222288,
0.11811159,
        0.65228345,  0.12600627,  0.29930509,  0.707418  ,
0.40846799,
        0.77710712,  0.9898757 ,  0.066143  ,  0.03848926,
0.60657335,
        0.1267287 ,  0.74302032,  0.29172421,  0.26313993,
0.5300043 ,
        0.7165474 ,  0.27528249,  0.09615341,  0.97487756,
0.0241328 ,
        0.20392797,  0.10509302,  0.14280144,  0.63864878,
0.06244715,
        0.09157081,  0.09331605,  0.63717624,  0.76232024,
0.6001074 ,
        0.20938485,  0.38838613,  0.26540498,  0.99363887,
0.13435462,
        0.30349458,  0.31829395,  0.88428133,  0.11488102,
0.04512342,
        0.50337055,  0.79185177,  0.35430623,  0.75690953,
0.13359719,
        0.21423579,  0.36235377,  0.82800962,  0.62067382,
0.29648282,
        0.33656441,  0.060269  ,  0.10604914,  0.55582991,
0.49981274,
        0.31001481,  0.40188703,  0.0598293 ,  0.36373464,
0.75302648,
        0.00345716,  0.26259115,  0.14930433,  0.05796219,
0.68235112,
        0.85294115,  0.96520541,  0.28848424,  0.95368613,
0.51753436,
       -0.07917549,  0.57658724,  0.85002159,  0.41860299,
0.25126078,
        0.65015252,  0.09243064,  0.29476406,  0.19907403, -
0.0189332 ,
        0.4502128 ,  0.06874807,  0.93414577,  0.15960603,
0.47994389,
        0.08167818,  0.76781977,  0.37484659,  0.8337975 ,
0.08142662,
        0.06375608,  0.80135977,  0.0787871 ,  0.37698754,
0.12056922,
        0.75436151,  0.11253585,  0.45511353,  0.8077408 ,
0.56638512,
        0.26694146,  0.94734412,  0.14826199,  0.69071522,
0.06686479,
```

```
        0.61530293,  0.92901096,  0.64082365,  0.92806796,
0.24665713,
        0.11704639,  0.72941123,  0.20054658,  0.61652836,
0.18748325,
        0.07760891,  0.08355363,  0.12283088,  0.09802856,
0.27234814,
        0.55177214,  0.19155111,  0.15742449,  0.20356839,
0.24364591,
        0.10679989,  0.11853511,  0.79290762,  0.24989981,
0.77424465,
        0.92618923,  0.40076525,  0.73412713,  0.10772705,
0.395258  ,
        0.21158942,  0.13725128,  0.38418665])

mse = mean_squared_error(y_test, y_pred)
mse

0.13878444869449827
```

## Logistic Regression

```
from sklearn.linear_model import LogisticRegression

logistic_model = LogisticRegression(random_state=42)
logistic_model.fit(x_train, y_train)

LogisticRegression(random_state=42)

y_pred = logistic_model.predict(x_test)

accuracy = accuracy_score(y_test, y_pred)
accuracy

0.8097014925373134
```

## Naive bayes

```
from sklearn.naive_bayes import GaussianNB
from sklearn.datasets import make_classification
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

x, y = make_classification(n_samples=1000, n_features=20, n_classes=2,
random_state=42)

X_train,X_test, y_train, y_test = train_test_split(x, y,
test_size=0.2, random_state=42)

naive_bayes_classifier = GaussianNB()
naive_bayes_classifier.fit(X_train, y_train)

GaussianNB()
```

```
y_pred = naive_bayes_classifier.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
confusion_mat = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:\n", confusion_mat)
classification_rep = classification_report(y_test, y_pred)
print("Classification Report:\n", classification_rep)
```

```
Accuracy: 0.795
Confusion Matrix:
 [[84  9]
 [32 75]]
Classification Report:
              precision    recall  f1-score   support

           0       0.72      0.90      0.80        93
           1       0.89      0.70      0.79       107

    accuracy                           0.80       200
   macro avg       0.81      0.80      0.79       200
weighted avg       0.81      0.80      0.79       200
```