```
import pandas as pd
import numpy as np
import random
from sklearn import preprocessing
le = preprocessing.LabelEncoder()

from sklearn.neighbors import KNeighborsClassifier

dataFile = pd.read_csv('tic-tac-toe.data')

dataFile.head()

top- top- middle- middle-
left- middle- right- left- middle-
```

		top- left- square	top- middle- square	right-		middle- middle- square			bottom- middle- square	bottom- right- square	
	0	х	х	х	х	0	0	х	0	0	ŗ
	1	х	Х	х	Х	0	0	0	х	0	ķ
	2	х	х	х	х	0	0	0	0	х	ŗ
	3	х	Х	х	Х	0	0	0	b	b	ŗ
	4	х	Х	х	Х	0	0	b	0	b	ŗ
4											<b>&gt;</b>

dataFile=dataFile.dropna()

```
dataFile.info()
```

<class 'pandas.core.frame.DataFrame'> RangeIndex: 958 entries, 0 to 957 Data columns (total 10 columns):

#	Column	Non	-Null Count	Dtype
0	top-left-square	958	non-null	object
1	top-middle-square	958	non-null	object
2	top-right-square	958	non-null	object
3	middle-left-square	958	non-null	object
4	middle-middle-square	958	non-null	object
5	middle-right-square	958	non-null	object
_	1	252	non-null	object
7anisano	pomyślnie.	X	non-null	object
	,		non-null	object
9	Class	958	non-null	object

dtypes: object(10)
memory usage: 75.0+ KB

top\_left\_square=list(dataFile["top-left-square"])
top\_middle\_square=list(dataFile["top-middle-square"])
top\_right\_square=list(dataFile["top-right-square"])
middle\_left\_square=list(dataFile["middle-left-square"])
middle\_middle\_square=list(dataFile["middle-middle-square"])
middle\_right\_square=list(dataFile["middle-right-square"])
bottom\_left\_square=list(dataFile["bottom-left-square"])
bottom\_middle\_square=list(dataFile["bottom-middle-square"])
bottom\_right\_square=list(dataFile["bottom-right-square"])

top\_left\_square\_encoded=le.fit\_transform(top\_left\_square)
top\_middle\_square\_encoded=le.fit\_transform(top\_middle\_square)
top\_right\_square\_encoded=le.fit\_transform(top\_right\_square)
middle\_left\_square\_encoded=le.fit\_transform(middle\_left\_square)
middle\_middle\_square\_encoded=le.fit\_transform(middle\_middle\_square)
middle\_right\_square\_encoded=le.fit\_transform(middle\_right\_square)
bottom\_left\_square\_encoded=le.fit\_transform(bottom\_left\_square)
bottom\_middle\_square\_encoded=le.fit\_transform(bottom\_middle\_square)
bottom\_right\_square\_encoded=le.fit\_transform(bottom\_right\_square)
Class\_encoded=le.fit\_transform(Class)

data=list(zip(top\_left\_square\_encoded,top\_middle\_square\_encoded,top\_right\_square\_encoded,middle\_left\_square\_encoded,middle\_middle\_square\_print(data)

```
[(2, 2, 2, 2, 1, 1, 2, 1, 1), (2, 2, 2, 2, 1, 1, 1, 2, 1), (2, 2, 2, 2, 1, 1, 1, 1, 2), (2, 2, 2, 2, 1, 1, 1, 0, 0), (2, 2, 2, 2, 1)
```

4

```
label=le.fit_transform(Class_encoded)
print(label)
 from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(data, label, test_size=800)
for i in range (3,38):
neigh = KNeighborsClassifier(n_neighbors=i,p=2)
neigh.fit(x_train,y_train)
neigh.score(x_test,y_test)
print(i, " " ,neigh.score(x_test,y_test))
 3 0.71
Zapisano pomyślnie.
  0.7175
 8
  0.7325
  0.72625
 10
  0.7275
  0.72
 11
 12
  0.71
  0.7
 13
  0.6775
 14
 15
  0.6725
 16
  0.66125
 17
  0.66
  0.66375
  0.67375
  0.68
 21
  0.67375
  0.67125
 22
  0.67875
 23
 24
  0.68375
 25
  0.68
 26
  0.68
 27
  0.67875
 28
  0.68375
  0.68125
  0.68125
 31
  0.67375
 32
  0.6725
  0.6775
 33
 34
  0.675
 35
  0.66375
 36
  0.66875
 37
  0.665
```

√ 0 s ukończono o 14:50

Zapisano pomyślnie.