
Workspace
Data Science Introduction
Private
Run
Edit
Export to PDF

```
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import MinMaxScaler

import pandas as pd
import numpy as np
```

```
data_wine = pd.read_csv('Calidad_de_vinos/winequality-white.csv', sep=";" )
data_wine2 = pd.read_csv('Calidad_de_vinos/winequality-red.csv', sep=";" )
data_wine2['quality'].unique()
data_wine2['quality'] = np.where(data_wine2['quality'] >= 6, 1, 0)
data_wine2
```

	fixed acidity float... 4.6 - 15.9	volatile acidity fl... 0.12 - 1.58	citric acid float64 0.0 - 1.0	residual sugar flo... 0.9 - 15.5	chlorides float64 0.012 - 0.611	free sulfur dioxide f 1.0 - 72.0	total sulfur dioxide 6.0 - 289.0	d 0
270	7.9	0.545	0.06	4	0.087	27	61	
271	11.5	0.18	0.51	4	0.104	4	23	
272	10.9	0.37	0.58	4	0.071	17	65	
273	8.4	0.715	0.2	2.4	0.076	10	38	
274	7.5	0.65	0.18	7	0.088	27	94	
275	7.9	0.545	0.06	4	0.087	27	61	
276	6.9	0.54	0.04	3	0.077	7	27	
277	11.5	0.18	0.51	4	0.104	4	23	
278	10.3	0.32	0.45	6.4	0.073	5	13	
279	8.9	0.4	0.32	5.6	0.087	10	47	

1599 rows, 12 cols 10 / page Page 28 of 160

```
x = data_wine.drop('quality', axis=1)
y = data_wine['quality']
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1,random_state=3,stratify=y)
print("Predictores Entrenamiento")
print(x_train)
print("Predictores Prueba")
print(x_test)
print("Objetivos Entrenamiento")
print(y_train)
print("Objetivos Prueba")
print(y_test)
```

```
Predictores Entrenamiento
      fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
2526            6.5              0.18         0.33             1.4      0.029
1274            8.4              0.35         0.56            13.8      0.048
2226            7.7              0.28         0.58            12.1      0.046
1855            8.0              0.22         0.28            14.0      0.053
4290            5.7              0.26         0.24            17.8      0.059
...             ...               ...          ...             ...        ...
3783            6.4              0.27         0.45             8.3      0.050
4355            6.4              0.31         0.28             2.5      0.039
1034            7.9              0.64         0.46            10.6      0.244
2835            6.3              0.25         0.22             3.3      0.048
4891            5.7              0.21         0.32             0.9      0.038

      free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
2526                 35.0                 138.0  0.99114  3.36       0.60
1274                 55.0                 190.0  0.99930  3.07       0.58
2226                 60.0                 177.0  0.99830  3.08       0.46
1855                 83.0                 197.0  0.99810  3.14       0.45
4290                 23.0                 124.0  0.99773  3.30       0.50
...                   ...                   ...      ...   ...        ...
3783                 52.0                 196.0  0.99550  3.18       0.48
4355                 34.0                 137.0  0.98946  3.22       0.38
1034                 33.0                 227.0  0.99830  2.87       0.74
2835                 41.0                 161.0  0.99256  3.16       0.50
4891                 38.0                 121.0  0.99074  3.24       0.46

      alcohol
2526     11.5
1274      9.4
2226      8.9
```

```
knn=KNeighborsClassifier(n_neighbors=1)
knn.fit(x_train,y_train)
predicciones=knn.predict(x_test)
print(confusion_matrix(y_test,predicciones))
```

```
[[ 1  0  1  0  0  0]
 [ 0  4  4  8  0  0]
 [ 0  2 96 43  4  1]
 [ 0  7 33 133 37 10]
 [ 0  0 10 31 44  3]
 [ 0  0  0  4  5  9]]
```

```
x = data_wine2.copy()
x.drop('quality', axis=1)
scaler = MinMaxScaler(feature_range=(0,1))
x = scaler.fit_transform(x)
x = pd.DataFrame(x)
y = data_wine2['quality']
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.1,random_state=3,stratify=y)
```

```
print("Predictores Entrenamiento")
print(x_train)
print("Predictores Prueba")
print(x_test)
print("Objetivos Entrenamiento")
print(y_train)
print("Objetivos Prueba")
print(y_test)
```

```
Predictores Entrenamiento
            0         1     2         3         4         5         6   \
848   0.159292  0.356164  0.21  0.061644  0.115192  0.183099  0.088339
1418  0.283186  0.280822  0.01  0.047945  0.108514  0.028169  0.045936
933   0.247788  0.335616  0.01  0.075342  0.103506  0.169014  0.113074
16    0.345133  0.109589  0.56  0.061644  0.133556  0.478873  0.342756
169   0.256637  0.400685  0.24  0.061644  0.580968  0.197183  0.201413
...        ...       ...   ...       ...       ...       ...       ...
1403  0.230088  0.143836  0.33  0.054795  0.081803  0.028169  0.024735
545   0.398230  0.239726  0.49  0.116438  0.136895  0.521127  0.353357
187   0.274336  0.400685  0.10  0.116438  0.120200  0.112676  0.070671
554   0.964602  0.359589  0.49  0.226027  0.138564  0.126761  0.060071
1561  0.283186  0.328767  0.26  0.075342  0.113523  0.422535  0.441696

            7         8         9         10   11
848   0.500734  0.669291  0.197605  0.215385  0.0
1418  0.361968  0.330709  0.077844  0.215385  0.0
933   0.544053  0.582677  0.191617  0.215385  0.0
16    0.501468  0.440945  0.251497  0.323077  1.0
169   0.464758  0.204724  0.754491  0.169231  0.0
...        ...       ...       ...       ...  ...
1403  0.435389  0.385827  0.461078  0.246154  1.0
545   0.596916  0.267717  0.155689  0.107692  0.0
187   0.552863  0.511811  0.095808  0.200000  0.0
554   0.960352  0.141732  0.245509  0.415385  0.0
1561  0.451542  0.370079  0.113772  0.230769  0.0

[1439 rows x 12 columns]
Predictores Prueba
            0         1     2         3         4         5         6   \
1446  0.203540  0.349315  0.02  0.068493  0.110184  0.239437  0.084806
```

```
knn=KNeighborsClassifier(n_neighbors=1)
knn.fit(x_train,y_train)
predicciones=knn.predict(x_test)
print(confusion_matrix(y_test,predicciones))
```

```
[[74  0]
 [ 0 86]]
```