```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from tensorflow import keras
import pandas as pd
%matplotlib inline
from sklearn.model_selection import *
from sklearn.preprocessing import *


data_frame = pd.read_csv('winequality-red.csv', parse_dates=True)


data_frame.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

```
data_frame.groupby('quality').count().reset_index()
```

	quality	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
0	3	10	10	10	10	10	10	10	10	10	10	10
1	4	53	53	53	53	53	53	53	53	53	53	53
2	5	681	681	681	681	681	681	681	681	681	681	681
3	6	638	638	638	638	638	638	638	638	638	638	638
4	7	199	199	199	199	199	199	199	199	199	199	199
5	8	18	18	18	18	18	18	18	18	18	18	18

```
data_frame['quality'].replace(to_replace={3: 0, 4: 1, 5: 2, 6: 3, 7: 4, 8: 5}, inplace=True)


X = data_frame[['fixed acidity','volatile acidity','citric acid','residual sugar','chlorides',
'free sulfur dioxide','total sulfur dioxide','density','pH','sulphates','alcohol']]
Y = data_frame['quality']


X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.2, random_state=24)
```

```
s = StandardScaler()
X_train = s.fit_transform(X_train)
X_test = s.transform(X_test)


model = tf.keras.models.Sequential([
  keras.layers.Dense(units=128, input_shape=(X_train.shape[1],), activation='relu'),
  keras.layers.Dense(units=64, activation='relu'),
  keras.layers.Dense(units=32, activation='relu'),
  keras.layers.Dense(units=16, activation='relu'),
  keras.layers.Dense(units=6, activation='softmax')
])
model.compile(loss='sparse_categorical_crossentropy', optimizer='sgd',  metrics=['accuracy'])
model.summary()
```

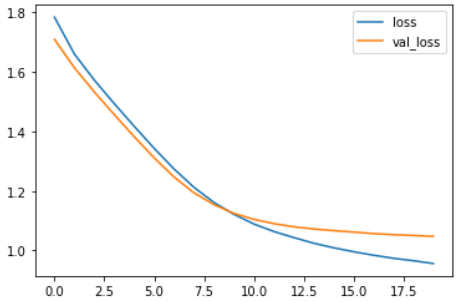
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               1536

 dense_1 (Dense)             (None, 64)                8256

 dense_2 (Dense)             (None, 32)                2080

 dense_3 (Dense)             (None, 16)                528

 dense_4 (Dense)             (None, 6)                 102

=================================================================
Total params: 12,502
Trainable params: 12,502
Non-trainable params: 0
_________________________________________________________________
```

```
h = model.fit(X_train, y_train, validation_data=(X_test,y_test), epochs=20, batch_size=32)
```

```
Epoch 1/20
40/40 [==============================] - 1s 7ms/step - loss: 1.7843 - accuracy: 0.2330 - val_loss: 1.7097 - val_accuracy: 0.4375
Epoch 2/20
40/40 [==============================] - 0s 3ms/step - loss: 1.6603 - accuracy: 0.4926 - val_loss: 1.6140 - val_accuracy: 0.5156
Epoch 3/20
40/40 [==============================] - 0s 3ms/step - loss: 1.5726 - accuracy: 0.5051 - val_loss: 1.5329 - val_accuracy: 0.5312
Epoch 4/20
40/40 [==============================] - 0s 3ms/step - loss: 1.4934 - accuracy: 0.5278 - val_loss: 1.4566 - val_accuracy: 0.5344
Epoch 5/20
40/40 [==============================] - 0s 3ms/step - loss: 1.4173 - accuracy: 0.5410 - val_loss: 1.3822 - val_accuracy: 0.5469
Epoch 6/20
40/40 [==============================] - 0s 3ms/step - loss: 1.3432 - accuracy: 0.5590 - val_loss: 1.3112 - val_accuracy: 0.5375
Epoch 7/20
40/40 [==============================] - 0s 3ms/step - loss: 1.2736 - accuracy: 0.5661 - val_loss: 1.2466 - val_accuracy: 0.5406
Epoch 8/20
40/40 [==============================] - 0s 2ms/step - loss: 1.2119 - accuracy: 0.5754 - val_loss: 1.1934 - val_accuracy: 0.5437
Epoch 9/20
40/40 [==============================] - 0s 3ms/step - loss: 1.1611 - accuracy: 0.5653 - val_loss: 1.1540 - val_accuracy: 0.5625
Epoch 10/20
40/40 [==============================] - 0s 3ms/step - loss: 1.1213 - accuracy: 0.5770 - val_loss: 1.1245 - val_accuracy: 0.5562
Epoch 11/20
40/40 [==============================] - 0s 3ms/step - loss: 1.0889 - accuracy: 0.5786 - val_loss: 1.1045 - val_accuracy: 0.5469
Epoch 12/20
40/40 [==============================] - 0s 2ms/step - loss: 1.0637 - accuracy: 0.5770 - val_loss: 1.0901 - val_accuracy: 0.5656
Epoch 13/20
40/40 [==============================] - 0s 2ms/step - loss: 1.0431 - accuracy: 0.5919 - val_loss: 1.0794 - val_accuracy: 0.5625
Epoch 14/20
```

```
40/40 [==============================] - 0s 3ms/step - loss: 1.0243 - accuracy: 0.5872 - val_loss: 1.0720 - val_accuracy: 0.5656
Epoch 15/20
40/40 [==============================] - 0s 3ms/step - loss: 1.0089 - accuracy: 0.5919 - val_loss: 1.0665 - val_accuracy: 0.5688
Epoch 16/20
40/40 [==============================] - 0s 2ms/step - loss: 0.9957 - accuracy: 0.5903 - val_loss: 1.0618 - val_accuracy: 0.5688
Epoch 17/20
40/40 [==============================] - 0s 2ms/step - loss: 0.9839 - accuracy: 0.5966 - val_loss: 1.0566 - val_accuracy: 0.5688
Epoch 18/20
40/40 [==============================] - 0s 3ms/step - loss: 0.9735 - accuracy: 0.5919 - val_loss: 1.0529 - val_accuracy: 0.5688
Epoch 19/20
40/40 [==============================] - 0s 3ms/step - loss: 0.9653 - accuracy: 0.5973 - val_loss: 1.0503 - val_accuracy: 0.5719
Epoch 20/20
40/40 [==============================] - 0s 3ms/step - loss: 0.9559 - accuracy: 0.5958 - val_loss: 1.0472 - val_accuracy: 0.5625
```

```
plt.plot(h.history['loss'], label='loss')
plt.plot(h.history['val_loss'], label='val_loss')
plt.legend()
plt.show()
```




```
ModelLoss, ModelAccuracy = model.evaluate(X_test, y_test)

print("Loss")
print(ModelLoss)
print("Accuracy")
print(ModelAccuracy)
```

```
10/10 [==============================] - 0s 2ms/step - loss: 1.0472 - accuracy: 0.5625
Loss
1.0472028255462646
Accuracy
0.5625
```

✓ 0 s ukończono o 13:52