Neural Networks in python

Imports

```
import numpy as np
import pandas as pd
import tensorflow as tf
import sklearn.model_selection as sk
```

Feature - Target split

```
df_eq = pd.read_csv('winequality.csv')
```

```
print(df_eq.shape)
df_eq.head()
y = df_eq['quality']
x = df_eq.drop('quality', axis=1)
y.head()
```

```
(1599, 12)
0    5
1    5
2    5
3    6
4    5
Name: quality, dtype: int64
```

```
x.head()
```

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

Train - Test split

```
X_train, X_test, y_train, y_test = sk.train_test_split(
                                   x ,y ,
                                   random_state=104,
                                   test_size=0.2,
                                   shuffle=True)
```

```
X_train.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates
1052	5.6	0.605	0.05	2.4	0.073	19.0	25.0	0.99258	3.56	0.55
175	6.9	0.500	0.04	1.5	0.085	19.0	49.0	0.99580	3.35	0.78
1001	9.9	0.350	0.38	1.5	0.058	31.0	47.0	0.99676	3.26	0.82
507	11.2	0.670	0.55	2.3	0.084	6.0	13.0	1.00000	3.17	0.71
111	8.4	0.620	0.09	2.2	0.084	11.0	108.0	0.99640	3.15	0.66

```
X_test.head()
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates
769	7.9	0.720	0.01	1.9	0.076	7.0	32.0	0.99668	3.39	0.54
379	8.3	0.420	0.38	2.5	0.094	24.0	60.0	0.99790	3.31	0.70
1531	6.1	0.705	0.10	2.8	0.081	13.0	28.0	0.99631	3.60	0.66
225	7.7	0.430	0.25	2.6	0.073	29.0	63.0	0.99615	3.37	0.58
1594	6.2	0.600	0.08	2.0	0.090	32.0	44.0	0.99490	3.45	0.58

```
y_train.head()
```

```
1052    5
175     5
1001    7
507     6
111     5
Name: quality, dtype: int64
```

```
y_test.head()
```

```
769     5
379     6
1531    5
225     6
1594    5
Name: quality, dtype: int64
```

```
df_model = tf.keras.Sequential([
  tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dense(32, activation='relu'),
  tf.keras.layers.Dense(1)
])
df_model.compile(loss='mae', optimizer='adam')
df_model.fit(X_train, y_train, epochs=100)

df_model.summary()
```

```
Epoch 83/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5064
Epoch 84/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5118
Epoch 85/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5174
Epoch 86/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5207
Epoch 87/100
40/40 [==============================] - 0s 2ms/step - loss: 0.4970
Epoch 88/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5237
Epoch 89/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5246
Epoch 90/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5336
Epoch 91/100
40/40 [==============================] - 0s 2ms/step - loss: 0.4857
Epoch 92/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5045
Epoch 93/100
40/40 [==============================] - 0s 2ms/step - loss: 0.4983
Epoch 94/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5052
Epoch 95/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5221
Epoch 96/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5154
Epoch 97/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5203
Epoch 98/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5067
Epoch 99/100
40/40 [==============================] - 0s 2ms/step - loss: 0.5079
Epoch 100/100
40/40 [==============================] - 0s 2ms/step - loss: 0.4983
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_15 (Dense)            (None, 64)                768

 dense_16 (Dense)            (None, 32)                2080

 dense_17 (Dense)            (None, 1)                 33

=================================================================
Total params: 2,881
Trainable params: 2,881
Non-trainable params: 0
_________________________________________________________________
```

```
losses = df_model.fit(X_train, y_train,
                      validation_data = (X_test, y_test),
                      batch_size = 100,
                      epochs = 100)
```

```
Epoch 1/100
13/13 [==============================] - 0s 35ms/step - loss: 0.4728 - val_loss: 0.4699
Epoch 2/100
13/13 [==============================] - 0s 4ms/step - loss: 0.4754 - val_loss: 0.4728
Epoch 3/100
13/13 [==============================] - 0s 5ms/step - loss: 0.4687 - val_loss: 0.4797
Epoch 4/100
13/13 [==============================] - 0s 6ms/step - loss: 0.4719 - val_loss: 0.4715
Epoch 5/100
13/13 [==============================] - 0s 7ms/step - loss: 0.4733 - val_loss: 0.4722
Epoch 6/100
13/13 [==============================] - 0s 5ms/step - loss: 0.4707 - val_loss: 0.4810
Epoch 7/100
13/13 [==============================] - 0s 4ms/step - loss: 0.4707 - val_loss: 0.4746
```

```
Epoch 8/100
13/13 [==============================] - 0s 7ms/step - loss: 0.4734 - val_loss: 0.4834
Epoch 9/100
13/13 [==============================] - 0s 8ms/step - loss: 0.4862 - val_loss: 0.4725
Epoch 10/100
13/13 [==============================] - 0s 9ms/step - loss: 0.4723 - val_loss: 0.4692
Epoch 11/100
13/13 [==============================] - 0s 15ms/step - loss: 0.4708 - val_loss: 0.4710
Epoch 12/100
13/13 [==============================] - 0s 8ms/step - loss: 0.4808 - val_loss: 0.4724
Epoch 13/100
13/13 [==============================] - 0s 6ms/step - loss: 0.4776 - val_loss: 0.5097
Epoch 14/100
13/13 [==============================] - 0s 5ms/step - loss: 0.4741 - val_loss: 0.4711
Epoch 15/100
13/13 [==============================] - 0s 8ms/step - loss: 0.4773 - val_loss: 0.4710
Epoch 16/100
13/13 [==============================] - 0s 7ms/step - loss: 0.4702 - val_loss: 0.4713
Epoch 17/100
13/13 [==============================] - 0s 8ms/step - loss: 0.4814 - val_loss: 0.4706
Epoch 18/100
13/13 [==============================] - 0s 8ms/step - loss: 0.4843 - val_loss: 0.4781
Epoch 19/100
13/13 [==============================] - 0s 7ms/step - loss: 0.4937 - val_loss: 0.4949
Epoch 20/100
13/13 [==============================] - 0s 8ms/step - loss: 0.4752 - val_loss: 0.4732
Epoch 21/100
13/13 [==============================] - 0s 5ms/step - loss: 0.4768 - val_loss: 0.4745
Epoch 22/100
13/13 [==============================] - 0s 6ms/step - loss: 0.4771 - val_loss: 0.4689
Epoch 23/100
13/13 [==============================] - 0s 5ms/step - loss: 0.4742 - val_loss: 0.4709
Epoch 24/100
13/13 [==============================] - 0s 6ms/step - loss: 0.4710 - val_loss: 0.4713
Epoch 25/100
13/13 [==============================] - 0s 5ms/step - loss: 0.4764 - val_loss: 0.4756
Epoch 26/100
13/13 [==============================] - 0s 4ms/step - loss: 0.4698 - val_loss: 0.4733
Epoch 27/100
13/13 [==============================] - 0s 5ms/step - loss: 0.4728 - val_loss: 0.4732
Epoch 28/100
13/13 [==============================] - 0s 5ms/step - loss: 0.4835 - val_loss: 0.4693
Epoch 29/100
13/13 [==============================] - 0s 7ms/step - loss: 0.4970 - val_loss: 0.5125
```

✓ 8s completed at 10:50 AM