codeFinal

November 26, 2022

[ ]:
```
#imports
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.neural_network import MLPClassifier
import tensorflow as tf

#Tools for modelling
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import confusion_matrix, classification_report,␣
↪accuracy_score
from sklearn.model_selection import cross_val_score, train_test_split

#Used to display NN diagram
from ann_visualizer.visualize import ann_viz

from tensorflow.keras.callbacks import TensorBoard
import time
```

[ ]:
```
#Assigning the model name & tensorboard location
NAME = "redWine_Quality_1H{}".format(int(time.time()))
tensorboard = TensorBoard(log_dir="logs/{}".format(NAME))
```

[ ]:
```
#read data
red = pd.read_csv('winequality-red.csv', delimiter = ';')

#Setting X as input and y as output
X = red[['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
         'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
         'pH', 'sulphates', 'alcohol']]

y = red['quality']

print(X.shape)
print("==========SHOWING y==========")
```

```
print(y.shape)
```

[ ]:
```
#Displaying data from red db
 #Using seaborn to display the data
dims = (11.7,8.27)
fig,ax = plt.subplots(figsize=dims)
CorMap = sns.heatmap(red.corr(), annot = True,fmt= '.1f', ax=ax)
plt.title('Red Wine Correlation')
```

[ ]:
```
#Red Wine Display Quality Bar Chart
barClass = pd.value_counts(red['quality']).sort_index()
print(barClass)
plt.bar(barClass.index, barClass, align = 'center')
plt.xticks(barClass.index)
plt.title('Red Wine Quality')
```

[ ]:
```
sns.barplot(x = 'quality', y='alcohol', data = red)
plt.title('Red Wine Alcohol vs Quality')
```

[ ]:
```
#Random 80/20 training/test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,␣
 ↪random_state=1)

print("X_train_full shape :",X_train.shape)
print("X_test shape :", X_test.shape)
```

[ ]:
```
# Initialize the model
model = tf.keras.models.Sequential()

# Configuring layers
# Input layer using the rectified linear unit activation function
model.add(tf.keras.layers.Dense(11, activation='relu', input_shape=(11,)))

# Additional hidden layers
model.add(tf.keras.layers.Dense(11, activation='relu'))

# Output layer using the sigmoid (s-function) activation function
model.add(tf.keras.layers.Dense(10, activation='sigmoid'))

# Compile and fit the model
model.compile(loss='sparse_categorical_crossentropy',
                    optimizer='adam',
                    metrics=['accuracy'])

model.fit(X_train, y_train, epochs=200, validation_split=0.1, shuffle=True,␣
 ↪callbacks=[tensorboard])
```

[ ]:
```
model.summary()
```

[ ]:
```
#Checking if model learned patterns and attributes
val_loss, val_acc = model.evaluate(X_test, y_test)
```

[ ]:
```
#saving model
model.save(NAME)
```

[ ]:
```
#creating NN diagram
ann_viz(model, title="Red Wine Quality NN")
```

[ ]:
```

```