```python
# -*- coding: utf-8 -*-
"""
Predicting house prices: a regression example
"""

from keras.datasets import boston_housing
(train_data, train_targets), (test_data, test_targets) =  boston_housing.load_data()

print("train_data.shape=",train_data.shape)
print("test_data.shape=",test_data.shape)

print("train_data[0]",train_data[0].reshape(len(train_data[0]),1))
print("train_targets=",train_targets)


#önişleme preprocessing
mean = train_data.mean(axis=0)
train_data -= mean
std = train_data.std(axis=0)
train_data /= std
test_data -= mean
test_data /= std



from keras import models
from keras import layers

def build_model():
    # Because we will need to instantiate
    # the same model multiple times,
    # we use a function to construct it.
    model = models.Sequential()
    model.add(layers.Dense(64, activation='relu',
                           input_shape=(train_data.shape[1],)))
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dense(1, activation='linear')) #linear
    model.compile(optimizer='rmsprop',
                  loss='mse',
                  metrics=['accuracy','mae'])
    return model

#Validating our approach using K-fold validation   (cross validation)
import numpy as np
k = 4
num_val_samples = len(train_data) // k # //tamsayı bölme
num_epochs = 10
all_scores = []


from keras import backend as K
# Some memory clean-up
K.clear_session()

num_epochs = 10
all_mae_histories = []
for i in range(k):
    print('processing fold #', i)
    # Prepare the validation data: data from partition # k
    val_data = train_data[i * num_val_samples: (i + 1) * num_val_samples]
    val_targets = train_targets[i * num_val_samples: (i + 1) * num_val_samples]

    # Prepare the training data: data from all other partitions
    partial_train_data = np.concatenate(
        [train_data[:i * num_val_samples],
         train_data[(i + 1) * num_val_samples:]],
        axis=0)
```

```
    partial_train_targets = np.concatenate(
        [train_targets[:i * num_val_samples],
         train_targets[(i + 1) * num_val_samples:]],
        axis=0)

    # Build the Keras model (already compiled)
    model = build_model()
    # Train the model (in silent mode, verbose=0)
    history = model.fit(partial_train_data,
                        partial_train_targets,
                        validation_data=(val_data, val_targets),
                        epochs=num_epochs,
                        batch_size=1,
                        verbose=0)

    mae_history = history.history['val_mean_absolute_error']
    all_mae_histories.append(mae_history)


average_mae_history = [
        np.mean([x[i] for x in all_mae_histories]) for i in range(num_epochs)]


# Grafik çizim işlemleri -----------------------------------------------------
import matplotlib.pyplot as plt

plt.plot(range(1, len(average_mae_history) + 1), average_mae_history)
plt.xlabel('Epochs')
plt.ylabel('Validation MAE')
plt.show()
```