Compiling and fitting a model

Puteaux, Fall/Winter 2020-2021

```
################################
##                            ##
##   Deep Learning in Python  ##
##                            ##
################################
```

§1 Introduction to Deep Learning in Python

§1.3 Building deep learning models with keras

§1.3.2 Compiling and fitting a model

1. Why is it necessary to compile the model?

- Specify the optimizer:
    - many options and mathematically complex
    - `adam` is usually a good choice
- Loss function:
    - `mean_squared_error` is common for regression

2. Code of compiling a model:

```python
[1]: import numpy as np
     from keras.layers import Dense
     from keras.models import Sequential

     predictors = np.loadtxt('ref1. Hourly wages predictors data.csv',
                             delimiter=',')
     n_cols = predictors.shape[1]

     model = Sequential()
     model.add(Dense(100, activation='relu', input_shape=(n_cols, )))
     model.add(Dense(100, activation='relu'))
     model.add(Dense(1))
     model.compile(optimizer='adam', loss='mean_squared_error')
```

3. What is fitting a model?

- Apply backpropagation and gradient descent with the data to update the weights.

- Scale data before fitting can ease optimization.

4. Code of fitting a model:

```
[2]: target = np.loadtxt('ref3. Hourly wages target data.csv', delimiter=',')

     model.fit(predictors, target)
```

```
17/17 [==============================] - 1s 41ms/step - loss: 422.6907
```

```
[2]: <tensorflow.python.keras.callbacks.History at 0x7fa93625abd0>
```

5. Practice exercises for compiling and fitting a model:

▶ **Package pre-loading:**

```
[3]: import pandas as pd
```

▶ **Data pre-loading:**

```
[4]: df = pd.read_csv('ref2. Hourly wages.csv')

     predictors = df.iloc[:, 1:].to_numpy()
```

▶ **Model compiling practice:**

```
[5]: # Import necessary modules
     import keras
     from keras.layers import Dense
     from keras.models import Sequential

     # Specify the model
     n_cols = predictors.shape[1]
     model = Sequential()
     model.add(Dense(50, activation='relu', input_shape=(n_cols, )))
     model.add(Dense(32, activation='relu'))
     model.add(Dense(1))

     # Compile the model
     model.compile(optimizer='adam', loss='mean_squared_error')

     # Verify that model contains information from compiling
     print("Loss function: " + model.loss)
```

```
Loss function: mean_squared_error
```

▶ **Data re-pre-loading:**

```
[6]: target = df.iloc[:, 0].to_numpy()
```

▶ **Model fitting practice:**

```python
[7]:  # Import necessary modules
      import keras
      from keras.layers import Dense
      from keras.models import Sequential

      # Specify the model
      n_cols = predictors.shape[1]
      model = Sequential()
      model.add(Dense(50, activation='relu', input_shape=(n_cols, )))
      model.add(Dense(32, activation='relu'))
      model.add(Dense(1))

      # Compile the model
      model.compile(optimizer='adam', loss='mean_squared_error')

      # Fit the model
      model.fit(predictors, target)
```

      17/17 [==============================] - 1s 59ms/step - loss: 32.0553

[7]: <tensorflow.python.keras.callbacks.History at 0x7fa9368a0590>