Creating a keras 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 Creating a keras model

1.1 What are the model-building steps?

- The Keras workflow has four steps:
  - specify Architecture
  - compile
  - fit
  - predict

1.2 Code of model specification:

[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))
```

1.3 Practice question for understanding the data:

- It will be started soon to building models in Keras to predict wages based on various professional and demographic factors by the next steps. Before starting building a model, it's good to understand the data by performing some exploratory analysis.
- It is recommended to use the `.head()` and `.describe()` methods in the IPython Shell to quickly overview the DataFrame.
- The target variable which will be predicting is `wage_per_hour`. Some of the predictor variables are binary indicators, where a value of `1` represents `True`, and `0` represents `False`.
- Of the nine predictor variables in the DataFrame, how many are binary indicators? The min and max values, as shown by `.describe()` will be informative here. How many binary indicator predictors are there?

  ☐ 0.

  ☐ 5.

  ☒ 6.

▶ Package pre-loading:

[2]:
```
import pandas as pd
```

▶ Data pre-loading:

[3]:
```
df = pd.read_csv('ref2. Hourly wages.csv')
```

▶ Question-solving method:

[4]:
```
df.head()
```

[4]:

	wage_per_hour	union	education_yrs	experience_yrs	age	female	marr	\
0	5.10	0	8	21	35	1	1	
1	4.95	0	9	42	57	1	1	
2	6.67	0	12	1	19	0	0	
3	4.00	0	12	4	22	0	0	
4	7.50	0	12	17	35	0	1	

	south	manufacturing	construction
0	0	1	0
1	0	1	0
2	0	1	0
3	0	0	0
4	0	0	0

[5]:
```
df.describe()
```

[5]:

	wage_per_hour	union	education_yrs	experience_yrs	age	\
count	534.000000	534.000000	534.000000	534.000000	534.000000	

mean	9.024064	0.179775	13.018727	17.822097	36.833333
std	5.139097	0.384360	2.615373	12.379710	11.726573
min	1.000000	0.000000	2.000000	0.000000	18.000000
25%	5.250000	0.000000	12.000000	8.000000	28.000000
50%	7.780000	0.000000	12.000000	15.000000	35.000000
75%	11.250000	0.000000	15.000000	26.000000	44.000000
max	44.500000	1.000000	18.000000	55.000000	64.000000

	female	marr	south	manufacturing	construction
count	534.000000	534.000000	534.000000	534.000000	534.000000
mean	0.458801	0.655431	0.292135	0.185393	0.044944
std	0.498767	0.475673	0.455170	0.388981	0.207375
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	1.000000	0.000000	0.000000	0.000000
75%	1.000000	1.000000	1.000000	0.000000	0.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000

[6]:
```
cols = df.columns
count = 0
for i in range(len(cols)):
    if ((df.iloc[:, i].unique()[0] in [0, 1])
            and (df.iloc[:, i].unique()[1] in [0, 1])):
        count += 1
    else:
        pass
print('There are {} binary indicator predictors here.'.format(count))
```

```
There are 6 binary indicator predictors here.
```

1.4 Practice exercises for creating a Keras model:

▶ **Package pre-loading:**

[7]:
```
import pandas as pd
```

▶ **Data pre-loading:**

[8]:
```
df = pd.read_csv('ref2. Hourly wages.csv')

target = df.iloc[:, 0].to_numpy()
predictors = df.iloc[:, 1:].to_numpy()
```

▶ **Model specifying practice:**

[9]:
```
# Import necessary modules
import keras
from keras.layers import Dense
```

```
from keras.models import Sequential

# Save the number of columns in predictors: n_cols
n_cols = predictors.shape[1]

# Set up the model: model
model = Sequential()

# Add the first layer
model.add(Dense(50, activation='relu', input_shape=(n_cols, )))

# Add the second layer
model.add(Dense(32, activation='relu'))

# Add the output layer
model.add(Dense(1))
```