Carlos J. Costa

ARTIFICIAL NEURAL NETWORKS

2021

LISBOA

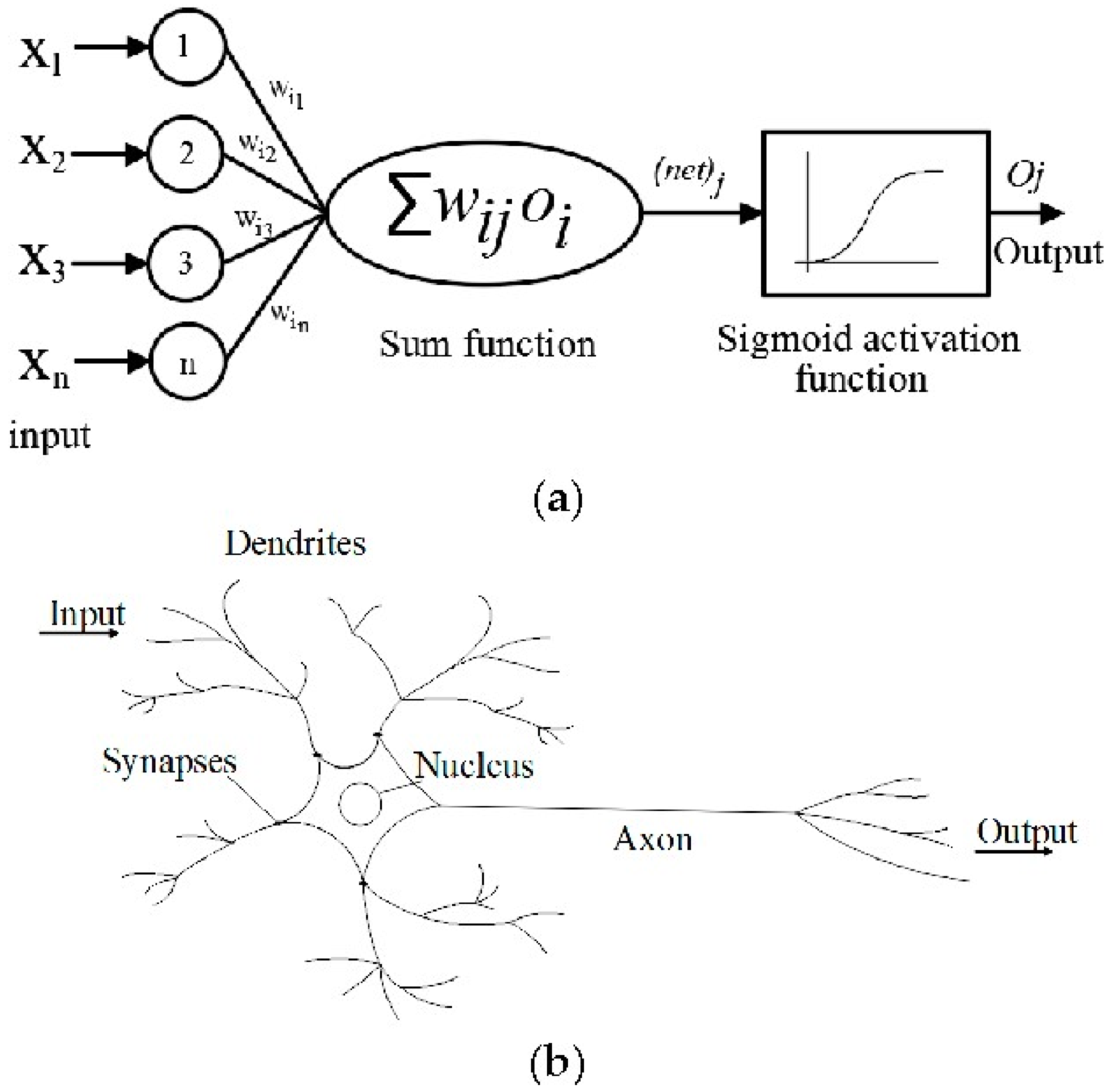
UNIVERSIDADE DE LISBOA

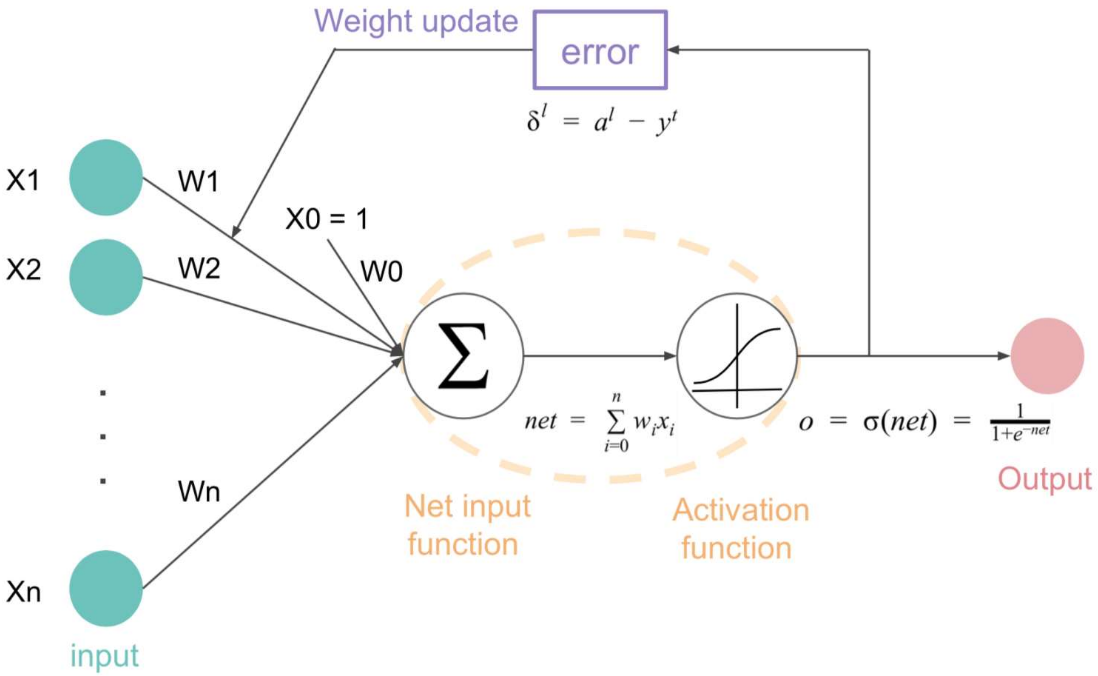
Agenda

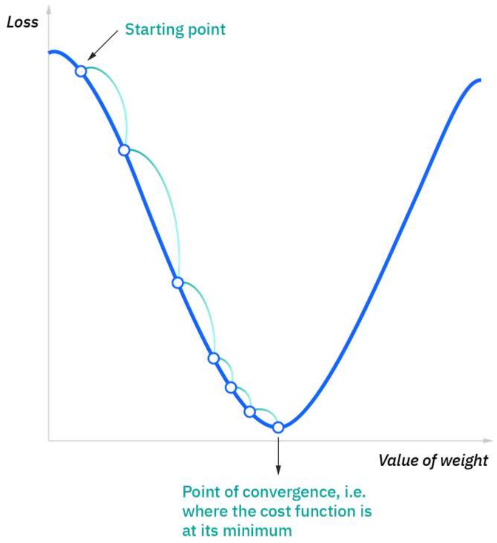
- History o ANN
- Concept of neural network
- Feedforward neural networks and backpropagation
- Types of neural networks
- TensorFlow
- Keras

History of neural networks

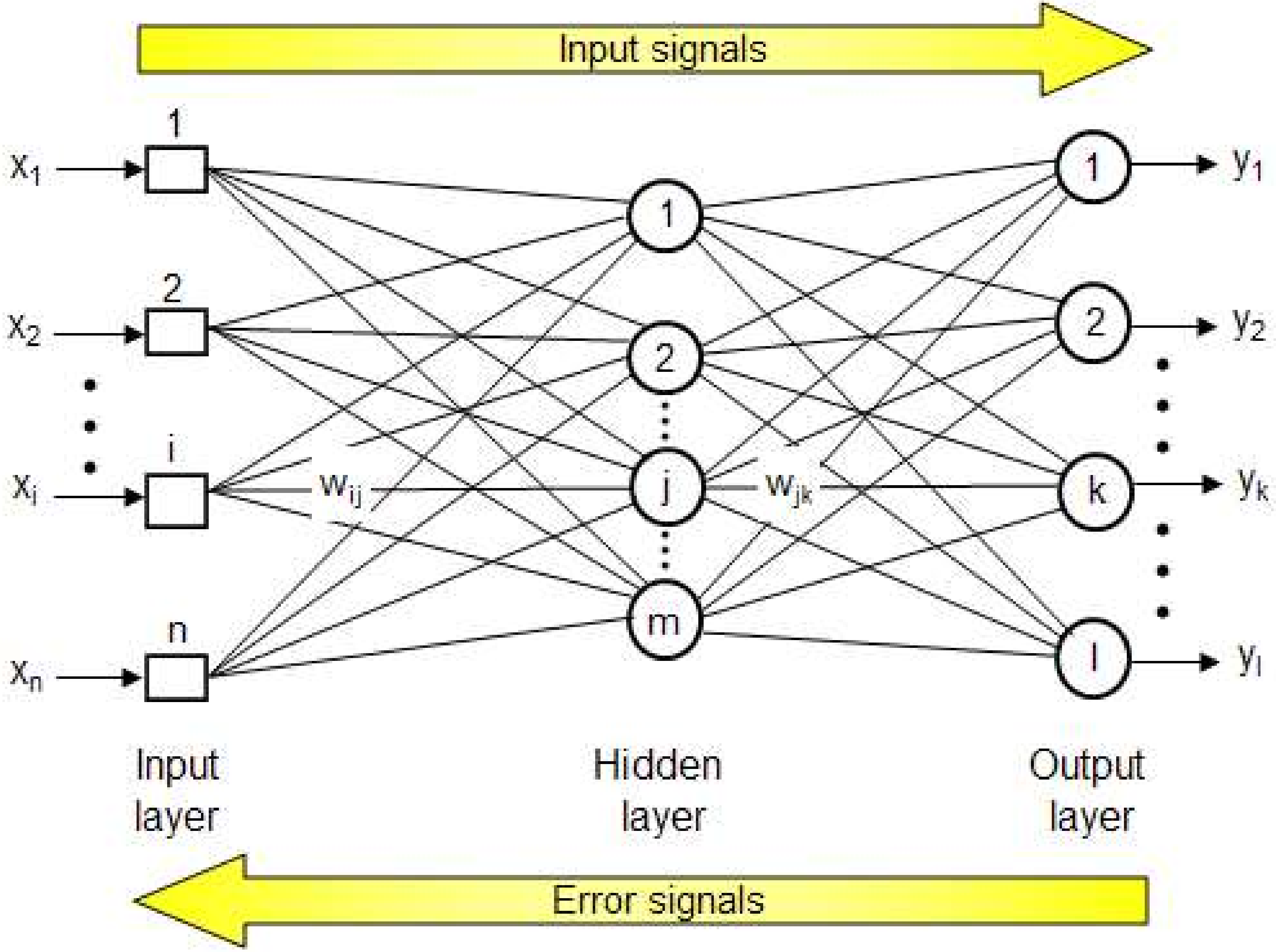
- 1943 - Warren S. McCulloch and Walter Pitts
- 1958 - Frank Rosenblatt
- 1974 - Paul Werbos
- 1989 - Yann LeCun

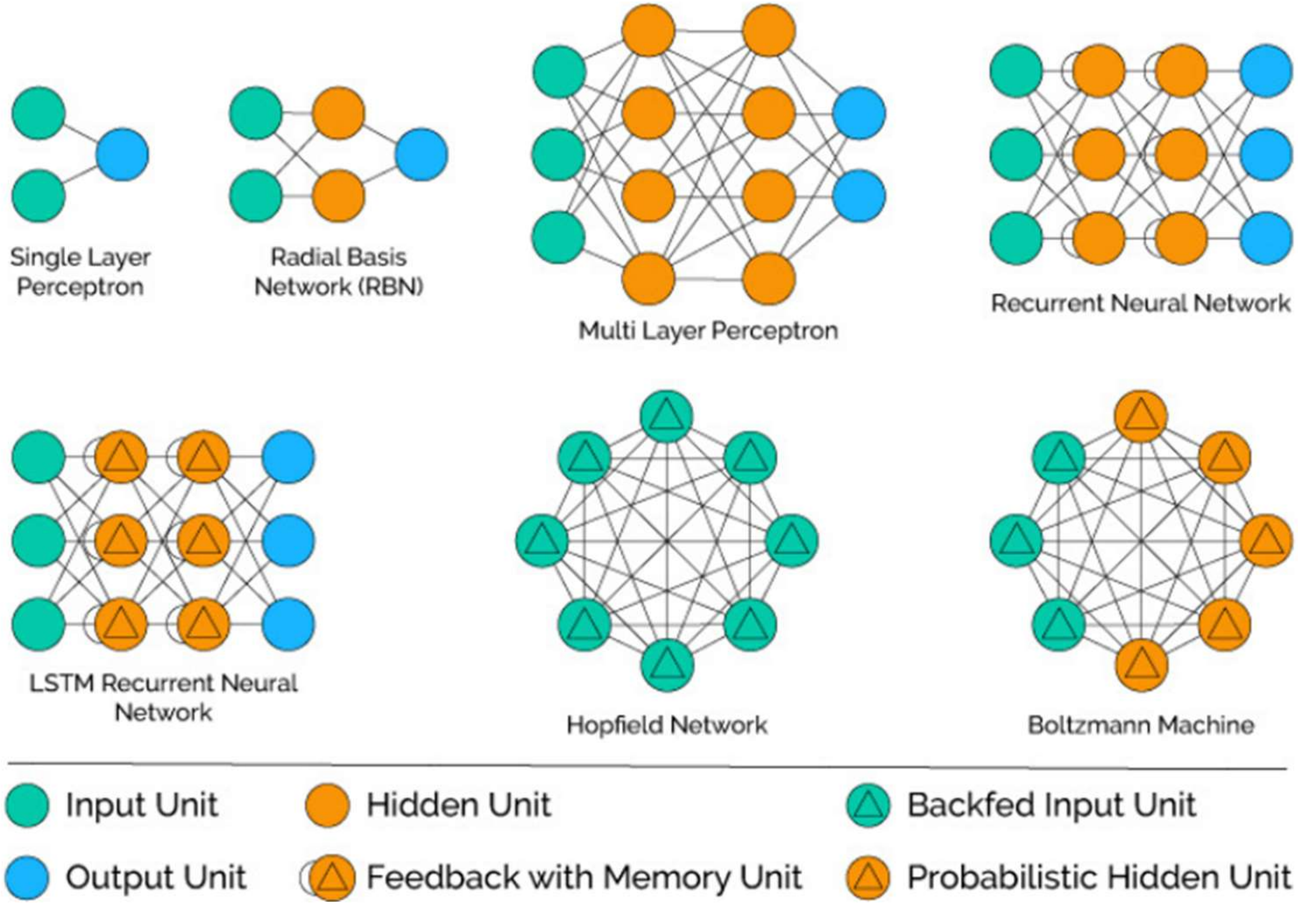

X1
X2
X3
Xn
1
2
3
n
input
wi1
wi2
wi3
win
∑wij oi
Sum function
(net)j
Sigmoid activation function
Oj
Output
(a)
Dendrites
Input
Synapses
Nucleus
Axon
Output
(b)





$$Cost\ Function = MSE = \frac{1}{2m}\sum_{i=1}^{m}(\hat{y} - y)^2$$


Input signals
x_1
x_2
x_i
x_n
1
2
i
n
w_{ij}
1
2
j
m
w_{jk}
1
2
k
l
y_1
y_2
y_k
y_l
Input layer
Hidden layer
Output layer
Error signals


Single Layer Perceptron
Radial Basis Network (RBN)
Multi Layer Perceptron
Recurrent Neural Network
LSTM Recurrent Neural Network
Hopfield Network
Boltzmann Machine
Input Unit
Hidden Unit
Backfed Input Unit
Output Unit
Feedback with Memory Unit
Probabilistic Hidden Unit

TensorFlow

- is a free and open-source software library for ML.
- particular focus on training and inference of deep neural networks
- symbolic math library based on dataflow and differentiable programming
- Google
- Apache License 2.0 since 2015
- Repository:

  https://github.com/tensorflow/tensorflow

  https://www.tensorflow.org/

Keras

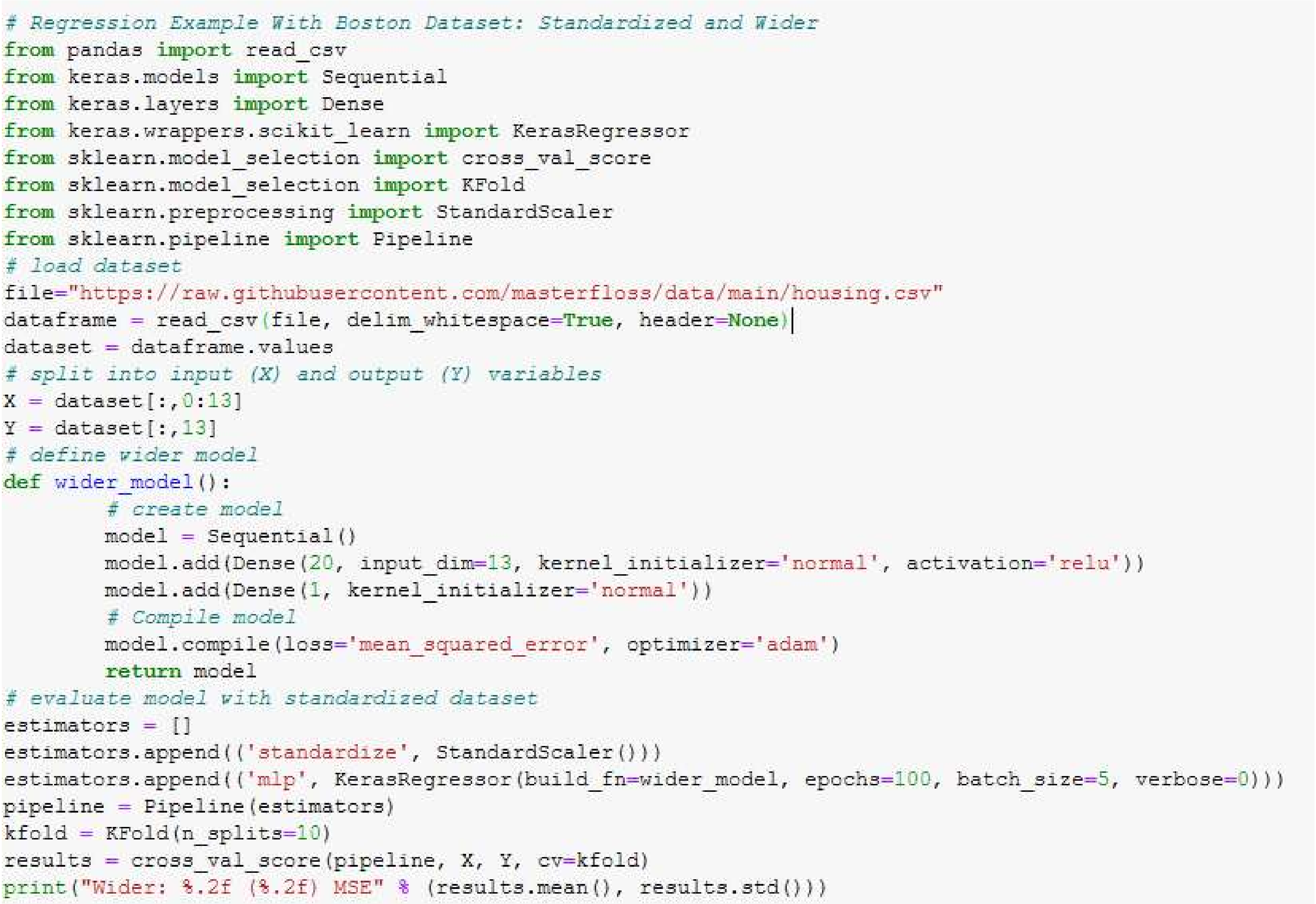
- Interface for the TensorFlow library.
- Open source
- Repository:

  https://github.com/keras-team/keras

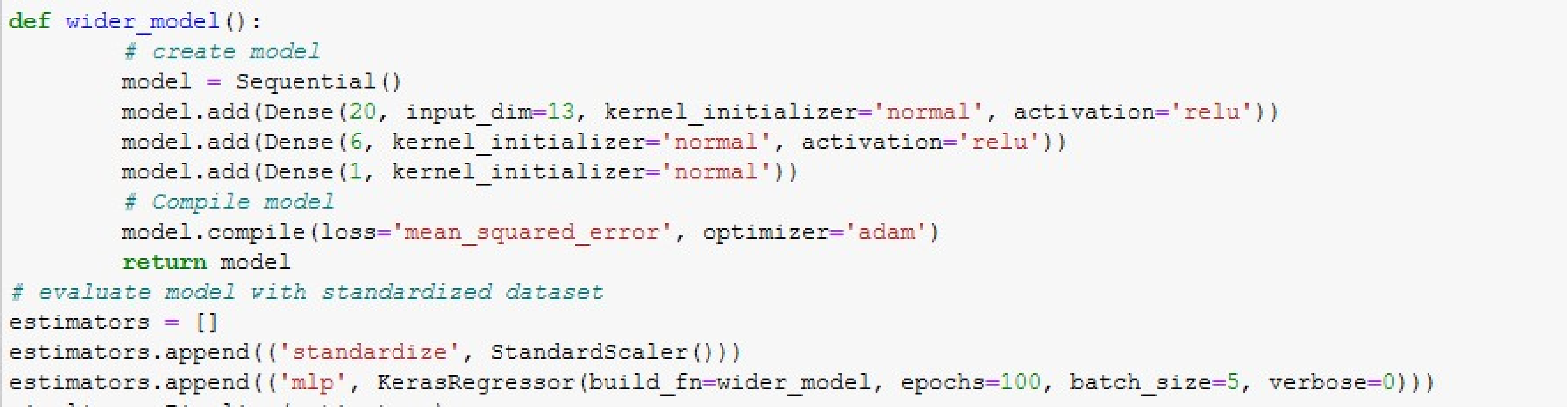
  https://keras.io/

Keras

- https://keras.io/guides/



```
# Regression Example With Boston Dataset: Standardized and Wider
from pandas import read_csv
from keras.models import Sequential
from keras.layers import Dense
from keras.wrappers.scikit_learn import KerasRegressor
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
# load dataset
file="https://raw.githubusercontent.com/masterfloss/data/main/housing.csv"
dataframe = read_csv(file, delim_whitespace=True, header=None)
dataset = dataframe.values
# split into input (X) and output (Y) variables
X = dataset[:,0:13]
Y = dataset[:,13]
# define wider model
def wider_model():
        # create model
        model = Sequential()
        model.add(Dense(20, input_dim=13, kernel_initializer='normal', activation='relu'))
        model.add(Dense(1, kernel_initializer='normal'))
        # Compile model
        model.compile(loss='mean_squared_error', optimizer='adam')
        return model
# evaluate model with standardized dataset
estimators = []
estimators.append(('standardize', StandardScaler()))
estimators.append(('mlp', KerasRegressor(build_fn=wider_model, epochs=100, batch_size=5, verbose=0)))
pipeline = Pipeline(estimators)
kfold = KFold(n_splits=10)
results = cross_val_score(pipeline, X, Y, cv=kfold)
print("Wider: %.2f (%.2f) MSE" % (results.mean(), results.std()))
```



```python
def wider_model():
        # create model
        model = Sequential()
        model.add(Dense(20, input_dim=13, kernel_initializer='normal', activation='relu'))
        model.add(Dense(6, kernel_initializer='normal', activation='relu'))
        model.add(Dense(1, kernel_initializer='normal'))
        # Compile model
        model.compile(loss='mean_squared_error', optimizer='adam')
        return model
# evaluate model with standardized dataset
estimators = []
estimators.append(('standardize', StandardScaler()))
estimators.append(('mlp', KerasRegressor(build_fn=wider_model, epochs=100, batch_size=5, verbose=0)))
```

scikit
learn

- Scikitlearn implements ANN
- Multi-layer Perceptron
- https://scikit-learn.org/stable/modules/classes.html#module-sklearn.neural_network
- https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html