IMDB Collection Download. We are limited to 10000 words.

```
import matplotlib.pyplot as plt
import keras
from keras import models
from keras import layers
```

```
from keras.datasets import imdb
(train_data, train_labels), (test_data, test_labels) = imdb.load_data(num_words=10000)
```

```
train_data[0]
```

```
 104,
 4,
 226,
 65,
 16,
 38,
 1334,
 88,
 12,
 16,
 283,
 5,
 16,
 4472,
 113,
 103,
 32,
 15,
 16,
 5345,
 19,
 178,
 32]
```

```
train_labels[0]
```

```
1
```

Decoding reviews:

```
word_index = imdb.get_word_index()
reverse_word_index = dict([(value, key) for (key, value) in word_index.items()])
decoded_review = ' '.join([reverse_word_index.get(i - 3, '?') for i in train_data[0]])
```

```
decoded_review
```

```
'? this film was just brilliant casting location scenery story direction everyone's really suited the part they play
ed and you could just imagine being there robert ? is an amazing actor and now the same being director ? father came
from the same scottish island as myself so i loved the fact there was a real connection with this film the witty rem
arks throughout the film were great it was just brilliant so much that i bought the film as soon as it was released
for ? and would recommend it to everyone to watch and the fly fishing was amazing really cried at the end it was so
sad and you know what they say if you cry at a film it must have been good and this definitely was also ? to the two
little boy's that played the ? of norman and paul they were just brilliant children are often left out of the ? list
i think because the stars that play them all grown up are such a big profile for the whole film but these children a
```

We store the reviews in a binary matrix:

```
import numpy as np
def vectorize_sequences(sequences, dimension=10000):
  results = np.zeros((len(sequences), dimension))
  for i, sequence in enumerate(sequences):
    results[i, sequence] = 1.
  return results

x_train = vectorize_sequences(train_data)
x_test = vectorize_sequences(test_data)


x_train[0]
```

```
array([0., 1., 1., ..., 0., 0., 0.])
```

Still determining the type of expected values:

```
y_train = np.asarray(train_labels).astype('float32')
y_test = np.asarray(test_labels).astype('float32')


y_train[0]
```

```
1.0
```

Model no 1.

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=512,validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 5s 71ms/step - loss: 0.6247 - acc: 0.6788 - val_loss: 0.5594 - val_acc: 0.7565
Epoch 2/20
30/30 [==============================] - 1s 26ms/step - loss: 0.5245 - acc: 0.8121 - val_loss: 0.5157 - val_acc: 0.8447
Epoch 3/20
30/30 [==============================] - 1s 25ms/step - loss: 0.4786 - acc: 0.8630 - val_loss: 0.4931 - val_acc: 0.8669
Epoch 4/20
30/30 [==============================] - 1s 40ms/step - loss: 0.4478 - acc: 0.8924 - val_loss: 0.4875 - val_acc: 0.8272
Epoch 5/20
30/30 [==============================] - 1s 33ms/step - loss: 0.4243 - acc: 0.9086 - val_loss: 0.4664 - val_acc: 0.8686
Epoch 6/20
30/30 [==============================] - 1s 40ms/step - loss: 0.4038 - acc: 0.9225 - val_loss: 0.4584 - val_acc: 0.8754
Epoch 7/20
30/30 [==============================] - 1s 24ms/step - loss: 0.3862 - acc: 0.9346 - val_loss: 0.4659 - val_acc: 0.8531
Epoch 8/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3704 - acc: 0.9431 - val_loss: 0.4631 - val_acc: 0.8577
Epoch 9/20
30/30 [==============================] - 1s 26ms/step - loss: 0.3565 - acc: 0.9494 - val_loss: 0.4567 - val_acc: 0.8646
Epoch 10/20
30/30 [==============================] - 1s 25ms/step - loss: 0.3433 - acc: 0.9578 - val_loss: 0.4529 - val_acc: 0.8679
Epoch 11/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3302 - acc: 0.9632 - val_loss: 0.4467 - val_acc: 0.8745
Epoch 12/20
30/30 [==============================] - 1s 24ms/step - loss: 0.3199 - acc: 0.9677 - val_loss: 0.4661 - val_acc: 0.8617
Epoch 13/20
30/30 [==============================] - 1s 22ms/step - loss: 0.3094 - acc: 0.9721 - val_loss: 0.4578 - val_acc: 0.8684
Epoch 14/20
30/30 [==============================] - 1s 23ms/step - loss: 0.2991 - acc: 0.9751 - val_loss: 0.4525 - val_acc: 0.8697
Epoch 15/20
30/30 [==============================] - 1s 23ms/step - loss: 0.2903 - acc: 0.9779 - val_loss: 0.4742 - val_acc: 0.8629
Epoch 16/20
```

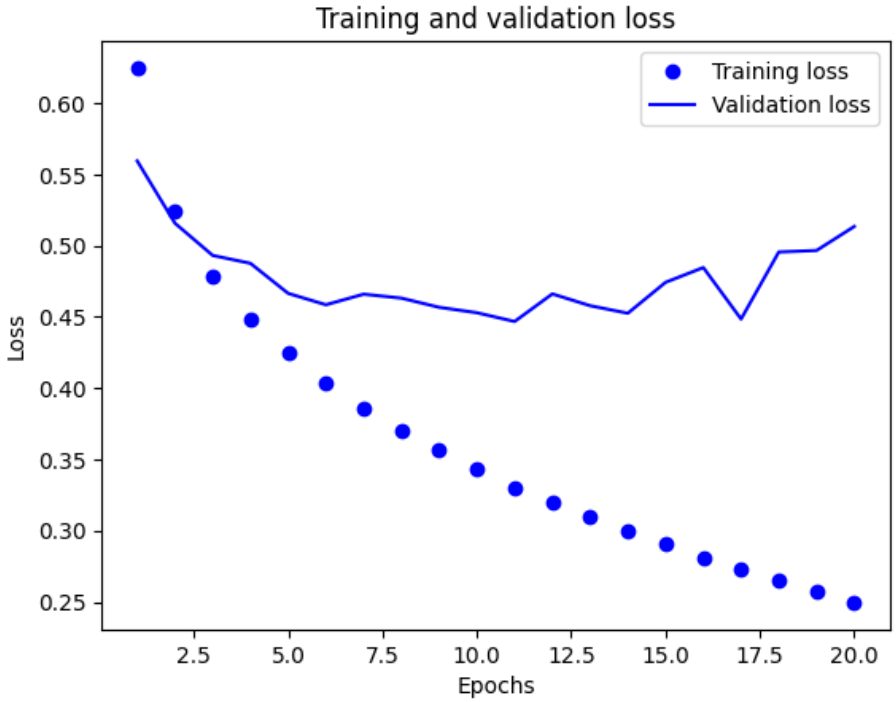
```
30/30 [==============================] - 1s 25ms/step - loss: 0.2813 - acc: 0.9799 - val_loss: 0.4846 - val_acc: 0.8611
Epoch 17/20
30/30 [==============================] - 1s 23ms/step - loss: 0.2728 - acc: 0.9822 - val_loss: 0.4484 - val_acc: 0.8721
Epoch 18/20
30/30 [==============================] - 1s 24ms/step - loss: 0.2645 - acc: 0.9835 - val_loss: 0.4955 - val_acc: 0.8607
Epoch 19/20
30/30 [==============================] - 1s 23ms/step - loss: 0.2570 - acc: 0.9847 - val_loss: 0.4965 - val_acc: 0.8593
Epoch 20/20
30/30 [==============================] - 1s 24ms/step - loss: 0.2495 - acc: 0.9853 - val_loss: 0.5134 - val_acc: 0.8584
```

```
history_dict = history.history
history_dict.keys()
```

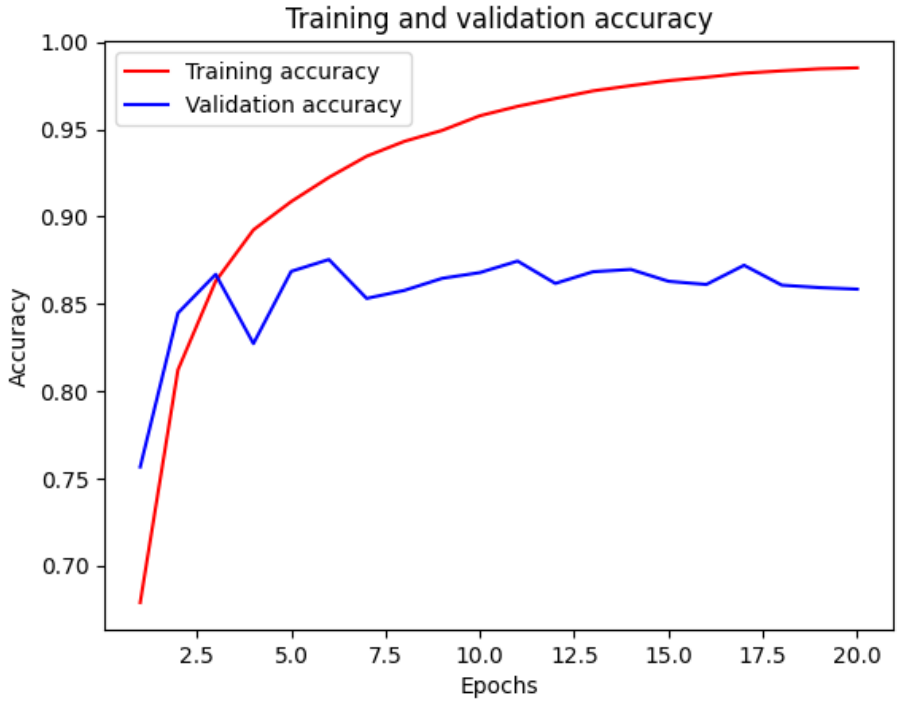
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Model no 2.

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(16, activation='relu', input_shape=(10000,)))
model.add(layers.Dense(8, activation='relu'))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=512,validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 3s 70ms/step - loss: 0.5111 - acc: 0.7755 - val_loss: 0.3856 - val_acc: 0.8631
Epoch 2/20
30/30 [==============================] - 1s 24ms/step - loss: 0.3185 - acc: 0.8916 - val_loss: 0.3095 - val_acc: 0.8858
Epoch 3/20
30/30 [==============================] - 1s 25ms/step - loss: 0.2382 - acc: 0.9214 - val_loss: 0.2823 - val_acc: 0.8890
Epoch 4/20
30/30 [==============================] - 1s 24ms/step - loss: 0.1887 - acc: 0.9383 - val_loss: 0.2738 - val_acc: 0.8899
Epoch 5/20
30/30 [==============================] - 1s 24ms/step - loss: 0.1580 - acc: 0.9499 - val_loss: 0.2904 - val_acc: 0.8838
Epoch 6/20
30/30 [==============================] - 1s 23ms/step - loss: 0.1342 - acc: 0.9578 - val_loss: 0.2855 - val_acc: 0.8879
Epoch 7/20
30/30 [==============================] - 1s 23ms/step - loss: 0.1156 - acc: 0.9643 - val_loss: 0.3058 - val_acc: 0.8804
Epoch 8/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0978 - acc: 0.9713 - val_loss: 0.3444 - val_acc: 0.8715
Epoch 9/20
30/30 [==============================] - 1s 41ms/step - loss: 0.0838 - acc: 0.9758 - val_loss: 0.3278 - val_acc: 0.8842
Epoch 10/20
30/30 [==============================] - 1s 32ms/step - loss: 0.0722 - acc: 0.9816 - val_loss: 0.4092 - val_acc: 0.8605
Epoch 11/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0611 - acc: 0.9851 - val_loss: 0.3705 - val_acc: 0.8794
Epoch 12/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0507 - acc: 0.9893 - val_loss: 0.3978 - val_acc: 0.8757
Epoch 13/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0443 - acc: 0.9900 - val_loss: 0.4208 - val_acc: 0.8750
Epoch 14/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0365 - acc: 0.9933 - val_loss: 0.4296 - val_acc: 0.8752
Epoch 15/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0339 - acc: 0.9929 - val_loss: 0.4582 - val_acc: 0.8689
Epoch 16/20
```

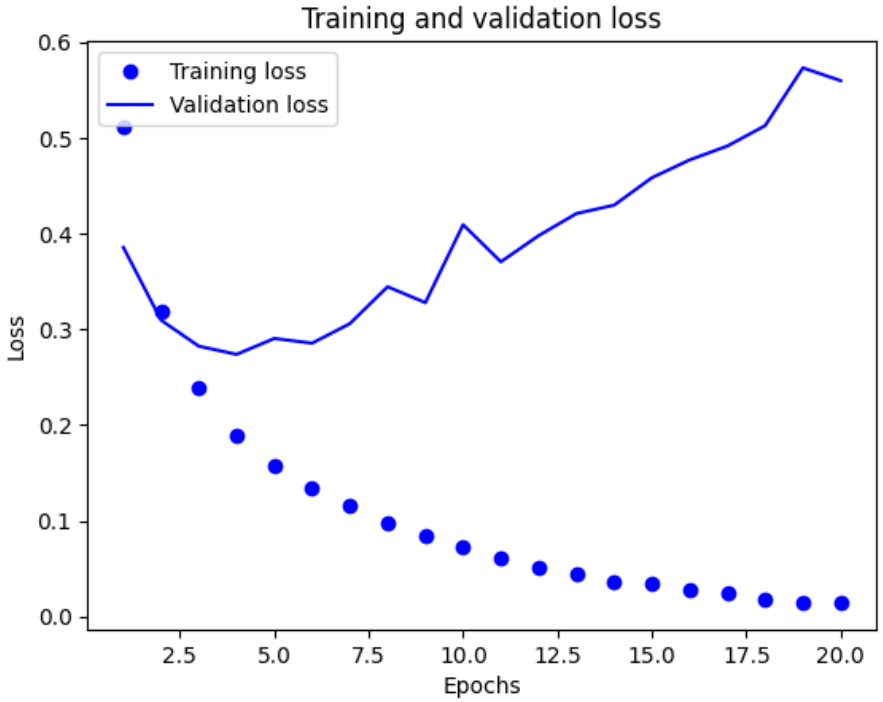
```
30/30 [==============================] - 1s 23ms/step - loss: 0.0275 - acc: 0.9952 - val_loss: 0.4768 - val_acc: 0.8688
Epoch 17/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0244 - acc: 0.9956 - val_loss: 0.4913 - val_acc: 0.8728
Epoch 18/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0186 - acc: 0.9981 - val_loss: 0.5126 - val_acc: 0.8710
Epoch 19/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0147 - acc: 0.9990 - val_loss: 0.5730 - val_acc: 0.8623
Epoch 20/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0149 - acc: 0.9973 - val_loss: 0.5596 - val_acc: 0.8698
```

```
history_dict = history.history
history_dict.keys()
```

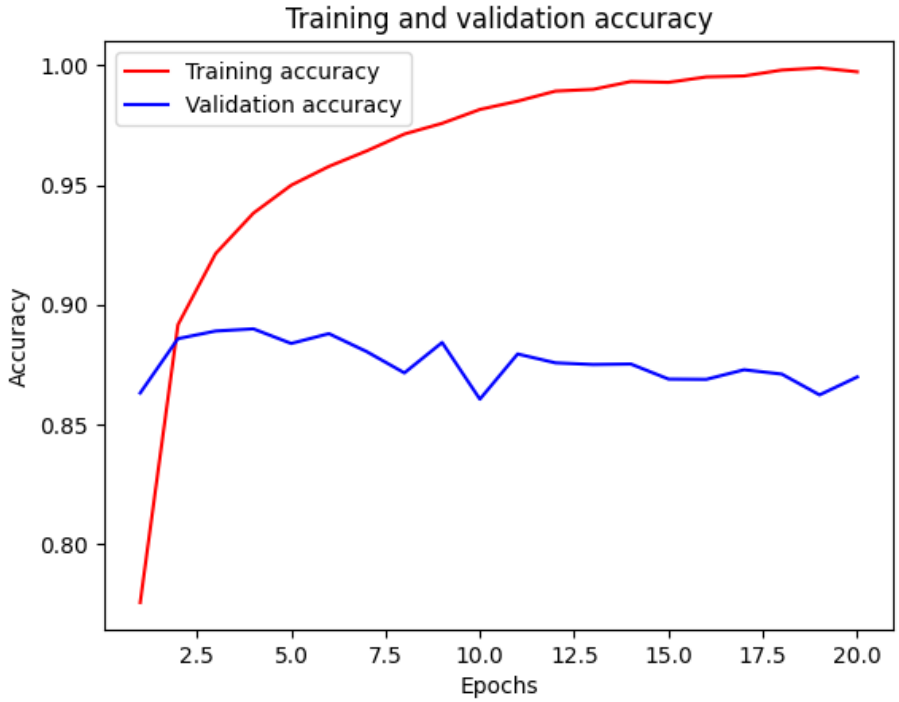
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Model no 3.

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dense(2, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=512,validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 3s 87ms/step - loss: 0.6477 - acc: 0.6452 - val_loss: 0.6048 - val_acc: 0.6331
Epoch 2/20
30/30 [==============================] - 1s 25ms/step - loss: 0.5616 - acc: 0.7757 - val_loss: 0.5517 - val_acc: 0.7175
Epoch 3/20
30/30 [==============================] - 1s 25ms/step - loss: 0.5101 - acc: 0.8303 - val_loss: 0.5116 - val_acc: 0.8058
Epoch 4/20
30/30 [==============================] - 1s 22ms/step - loss: 0.4731 - acc: 0.8673 - val_loss: 0.4912 - val_acc: 0.8277
Epoch 5/20
30/30 [==============================] - 1s 23ms/step - loss: 0.4453 - acc: 0.8894 - val_loss: 0.4729 - val_acc: 0.8703
Epoch 6/20
30/30 [==============================] - 1s 25ms/step - loss: 0.4225 - acc: 0.9075 - val_loss: 0.4656 - val_acc: 0.8553
Epoch 7/20
30/30 [==============================] - 1s 23ms/step - loss: 0.4032 - acc: 0.9206 - val_loss: 0.4591 - val_acc: 0.8579
Epoch 8/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3857 - acc: 0.9303 - val_loss: 0.4457 - val_acc: 0.8804
Epoch 9/20
30/30 [==============================] - 1s 25ms/step - loss: 0.3701 - acc: 0.9403 - val_loss: 0.4494 - val_acc: 0.8660
Epoch 10/20
30/30 [==============================] - 1s 24ms/step - loss: 0.3553 - acc: 0.9469 - val_loss: 0.4375 - val_acc: 0.8786
Epoch 11/20
30/30 [==============================] - 1s 24ms/step - loss: 0.3426 - acc: 0.9550 - val_loss: 0.4378 - val_acc: 0.8759
Epoch 12/20
30/30 [==============================] - 1s 22ms/step - loss: 0.3295 - acc: 0.9617 - val_loss: 0.4463 - val_acc: 0.8704
Epoch 13/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3176 - acc: 0.9669 - val_loss: 0.4400 - val_acc: 0.8739
Epoch 14/20
30/30 [==============================] - 1s 25ms/step - loss: 0.3068 - acc: 0.9715 - val_loss: 0.4729 - val_acc: 0.8588
Epoch 15/20
30/30 [==============================] - 1s 22ms/step - loss: 0.2967 - acc: 0.9739 - val_loss: 0.4804 - val_acc: 0.8592
Epoch 16/20
```

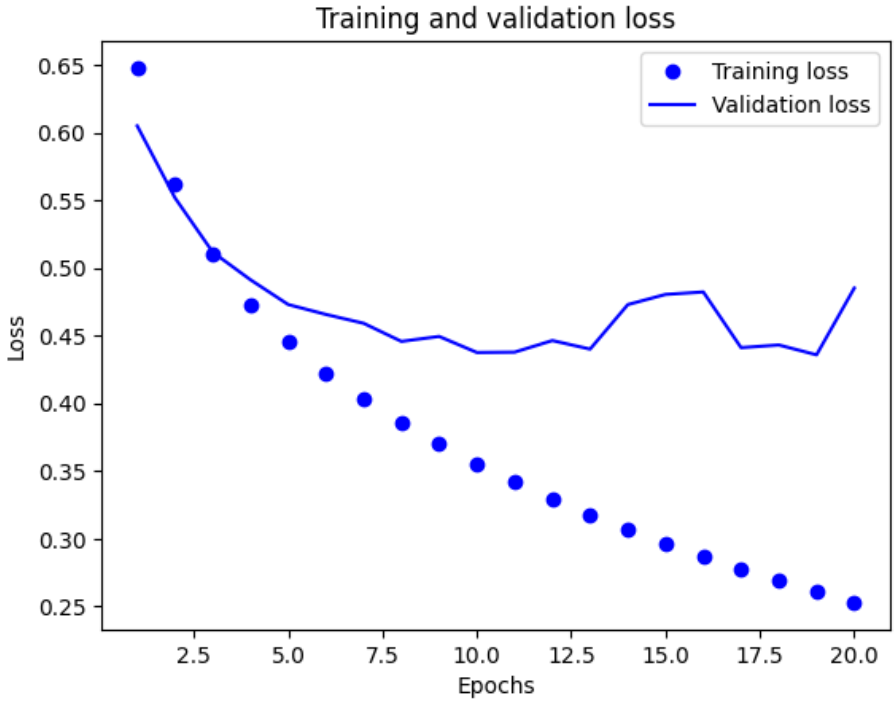
```
30/30 [==============================] - 1s 27ms/step - loss: 0.2868 - acc: 0.9776 - val_loss: 0.4822 - val_acc: 0.8598
Epoch 17/20
30/30 [==============================] - 1s 41ms/step - loss: 0.2774 - acc: 0.9804 - val_loss: 0.4411 - val_acc: 0.8741
Epoch 18/20
30/30 [==============================] - 1s 32ms/step - loss: 0.2693 - acc: 0.9823 - val_loss: 0.4431 - val_acc: 0.8740
Epoch 19/20
30/30 [==============================] - 1s 28ms/step - loss: 0.2611 - acc: 0.9827 - val_loss: 0.4358 - val_acc: 0.8733
Epoch 20/20
30/30 [==============================] - 1s 22ms/step - loss: 0.2529 - acc: 0.9850 - val_loss: 0.4851 - val_acc: 0.8668
```

```
history_dict = history.history
history_dict.keys()
```

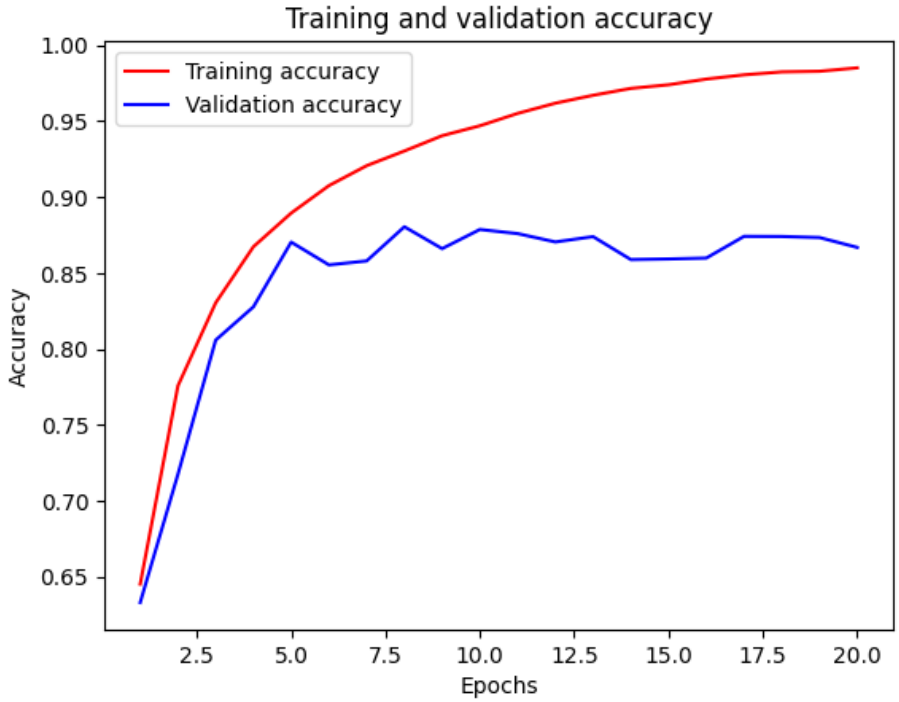
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Model no 4.

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(8, activation='relu', input_shape=(10000,)))
model.add(layers.Dense(8, activation='relu'))
model.add(layers.Dense(8, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=512,validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 4s 119ms/step - loss: 0.5942 - acc: 0.7443 - val_loss: 0.4822 - val_acc: 0.8555
Epoch 2/20
30/30 [==============================] - 2s 50ms/step - loss: 0.3908 - acc: 0.8851 - val_loss: 0.3540 - val_acc: 0.8722
Epoch 3/20
30/30 [==============================] - 1s 27ms/step - loss: 0.2799 - acc: 0.9113 - val_loss: 0.2997 - val_acc: 0.8870
Epoch 4/20
30/30 [==============================] - 1s 25ms/step - loss: 0.2203 - acc: 0.9281 - val_loss: 0.2799 - val_acc: 0.8887
Epoch 5/20
30/30 [==============================] - 1s 24ms/step - loss: 0.1793 - acc: 0.9430 - val_loss: 0.2782 - val_acc: 0.8867
Epoch 6/20
30/30 [==============================] - 1s 24ms/step - loss: 0.1513 - acc: 0.9533 - val_loss: 0.2799 - val_acc: 0.8862
Epoch 7/20
30/30 [==============================] - 1s 25ms/step - loss: 0.1266 - acc: 0.9620 - val_loss: 0.2923 - val_acc: 0.8858
Epoch 8/20
30/30 [==============================] - 1s 24ms/step - loss: 0.1094 - acc: 0.9669 - val_loss: 0.3030 - val_acc: 0.8846
Epoch 9/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0918 - acc: 0.9736 - val_loss: 0.3244 - val_acc: 0.8812
Epoch 10/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0783 - acc: 0.9778 - val_loss: 0.3449 - val_acc: 0.8789
Epoch 11/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0648 - acc: 0.9837 - val_loss: 0.3933 - val_acc: 0.8754
Epoch 12/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0552 - acc: 0.9873 - val_loss: 0.3807 - val_acc: 0.8760
Epoch 13/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0468 - acc: 0.9901 - val_loss: 0.4144 - val_acc: 0.8784
Epoch 14/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0398 - acc: 0.9921 - val_loss: 0.4260 - val_acc: 0.8756
Epoch 15/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0316 - acc: 0.9944 - val_loss: 0.4590 - val_acc: 0.8716
Epoch 16/20
```

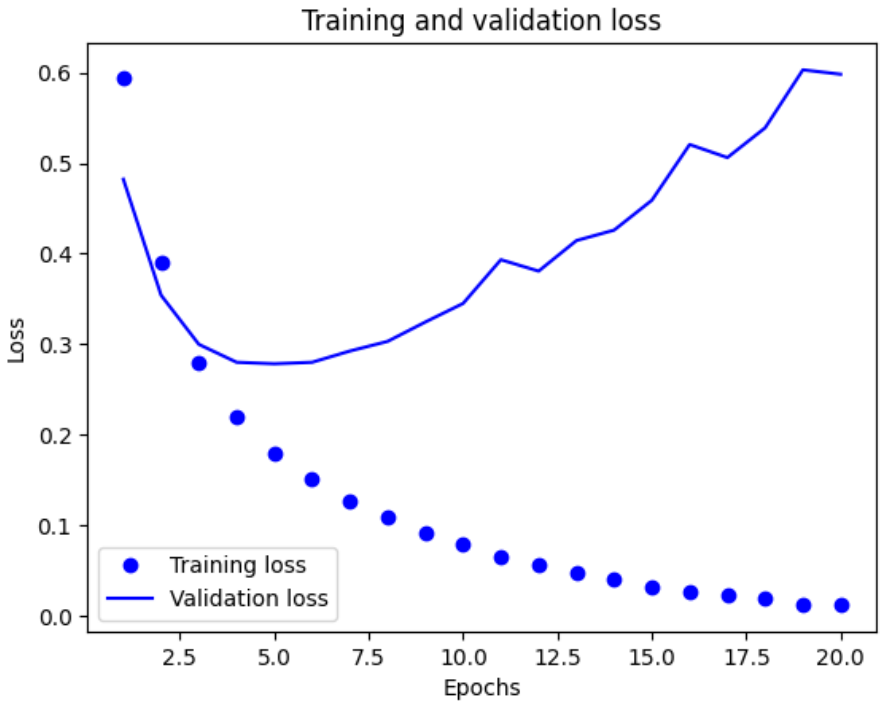
```
30/30 [==============================] - 1s 26ms/step - loss: 0.0255 - acc: 0.9959 - val_loss: 0.5206 - val_acc: 0.8706
Epoch 17/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0225 - acc: 0.9963 - val_loss: 0.5061 - val_acc: 0.8730
Epoch 18/20
30/30 [==============================] - 1s 42ms/step - loss: 0.0193 - acc: 0.9964 - val_loss: 0.5390 - val_acc: 0.8731
Epoch 19/20
30/30 [==============================] - 1s 31ms/step - loss: 0.0122 - acc: 0.9992 - val_loss: 0.6031 - val_acc: 0.8686
Epoch 20/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0128 - acc: 0.9987 - val_loss: 0.5983 - val_acc: 0.8709
```

```
history_dict = history.history
history_dict.keys()
```

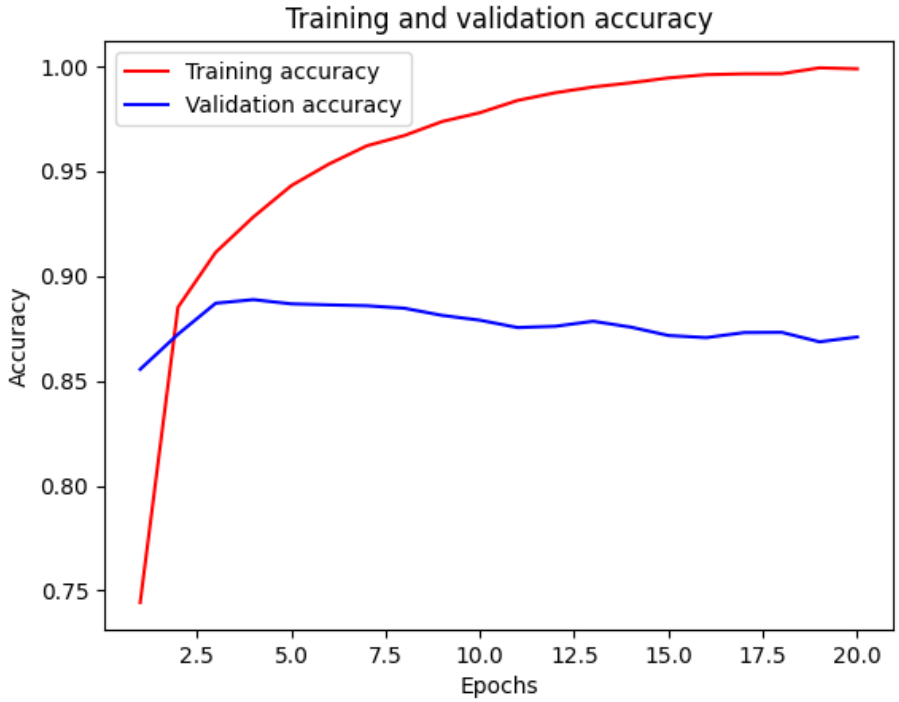
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Model no 5 EXTRA

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=512,validation_data=(x_val, y_val))
```

References - Definitions (5)

```
model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
# %% [markdown]
# We create validation data from some of the training data:
# %%
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
# %% [markdown]
# We are training the model:
# %%
```

partial_x_train = x_train[10000:]
partial_x_train = x_train[10000:]
partial_x_train = x_train[10000:]
partial_x_train = x_train[10000:]
partial_x_train = x_train[10000:]

```
Epoch 1/20
30/30 [==============================] - 4s 70ms/step - loss: 0.6594 - acc: 0.6088 - val_loss: 0.6181 - val_acc: 0.8140
Epoch 2/20
30/30 [==============================] - 1s 24ms/step - loss: 0.6024 - acc: 0.6732 - val_loss: 0.5601 - val_acc: 0.8473
Epoch 3/20
30/30 [==============================] - 1s 25ms/step - loss: 0.5602 - acc: 0.6920 - val_loss: 0.5131 - val_acc: 0.8593
Epoch 4/20
30/30 [==============================] - 1s 25ms/step - loss: 0.5254 - acc: 0.7064 - val_loss: 0.4747 - val_acc: 0.8668
Epoch 5/20
30/30 [==============================] - 1s 22ms/step - loss: 0.4969 - acc: 0.7179 - val_loss: 0.4363 - val_acc: 0.8760
Epoch 6/20
30/30 [==============================] - 1s 31ms/step - loss: 0.4773 - acc: 0.7280 - val_loss: 0.4220 - val_acc: 0.8778
Epoch 7/20
30/30 [==============================] - 1s 33ms/step - loss: 0.4456 - acc: 0.7875 - val_loss: 0.3836 - val_acc: 0.8817
Epoch 8/20
30/30 [==============================] - 1s 30ms/step - loss: 0.4219 - acc: 0.8103 - val_loss: 0.3836 - val_acc: 0.8784
Epoch 9/20
30/30 [==============================] - 1s 41ms/step - loss: 0.4050 - acc: 0.8172 - val_loss: 0.3493 - val_acc: 0.8871
```

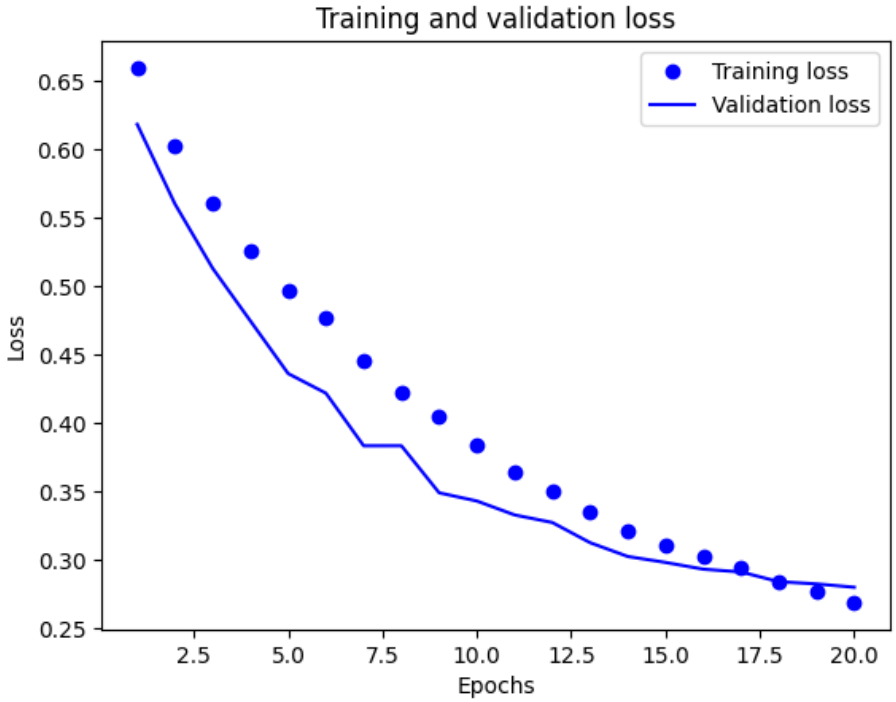
```
Epoch 10/20
30/30 [==============================] - 1s 25ms/step - loss: 0.3842 - acc: 0.8299 - val_loss: 0.3433 - val_acc: 0.8876
Epoch 11/20
30/30 [==============================] - 1s 25ms/step - loss: 0.3642 - acc: 0.8391 - val_loss: 0.3331 - val_acc: 0.8885
Epoch 12/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3507 - acc: 0.8470 - val_loss: 0.3276 - val_acc: 0.8892
Epoch 13/20
30/30 [==============================] - 1s 26ms/step - loss: 0.3350 - acc: 0.8586 - val_loss: 0.3129 - val_acc: 0.8891
Epoch 14/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3216 - acc: 0.8617 - val_loss: 0.3028 - val_acc: 0.8879
Epoch 15/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3107 - acc: 0.8679 - val_loss: 0.2985 - val_acc: 0.8865
Epoch 16/20
30/30 [==============================] - 1s 26ms/step - loss: 0.3027 - acc: 0.8651 - val_loss: 0.2935 - val_acc: 0.8872
Epoch 17/20
30/30 [==============================] - 1s 25ms/step - loss: 0.2942 - acc: 0.8733 - val_loss: 0.2915 - val_acc: 0.8866
Epoch 18/20
30/30 [==============================] - 1s 22ms/step - loss: 0.2845 - acc: 0.8759 - val_loss: 0.2844 - val_acc: 0.8870
Epoch 19/20
30/30 [==============================] - 1s 25ms/step - loss: 0.2769 - acc: 0.8793 - val_loss: 0.2828 - val_acc: 0.8864
Epoch 20/20
30/30 [==============================] - 1s 22ms/step - loss: 0.2692 - acc: 0.8802 - val_loss: 0.2804 - val_acc: 0.8869
```

```
history_dict = history.history
history_dict.keys()
```

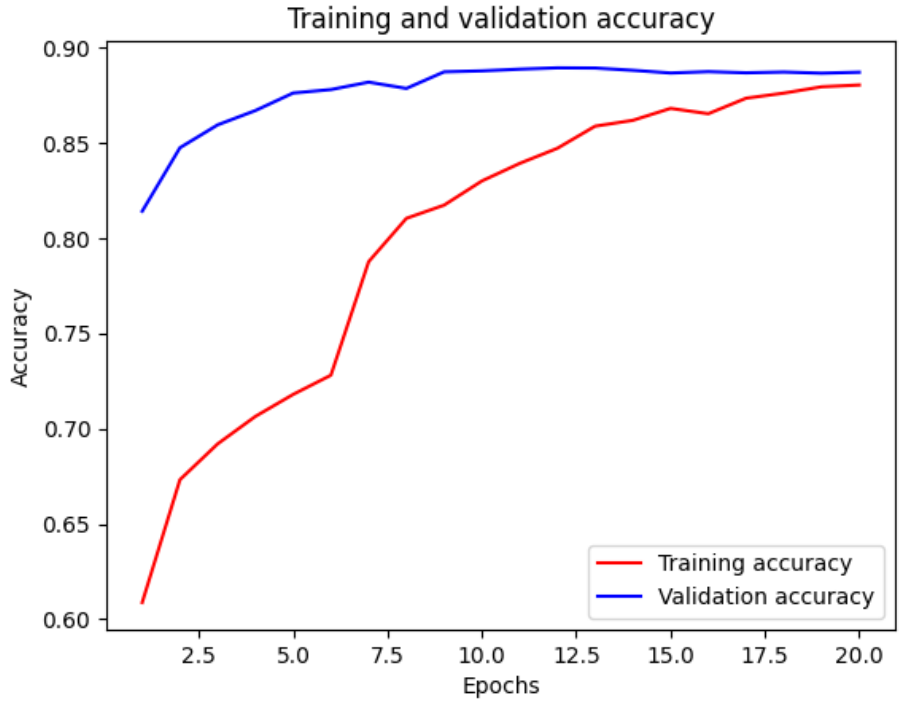
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Number of epochs 10

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=10,batch_size=512,validation_data=(x_val, y_val))
```

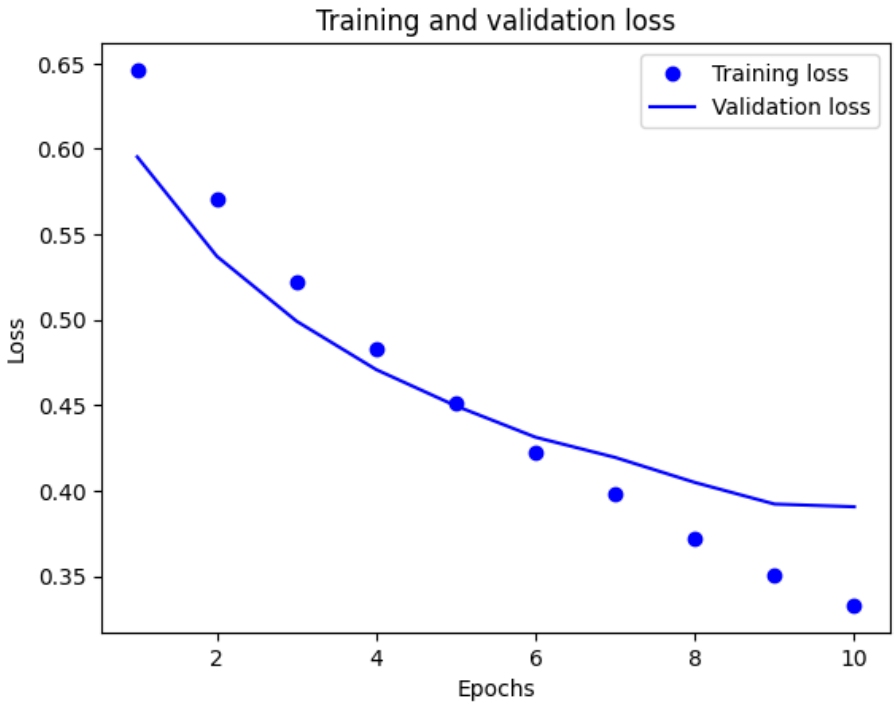
```
Epoch 1/10
30/30 [==============================] - 3s 73ms/step - loss: 0.6462 - acc: 0.6404 - val_loss: 0.5954 - val_acc: 0.7649
Epoch 2/10
30/30 [==============================] - 1s 23ms/step - loss: 0.5702 - acc: 0.7839 - val_loss: 0.5371 - val_acc: 0.8027
Epoch 3/10
30/30 [==============================] - 1s 23ms/step - loss: 0.5221 - acc: 0.8375 - val_loss: 0.4992 - val_acc: 0.8418
Epoch 4/10
30/30 [==============================] - 1s 24ms/step - loss: 0.4830 - acc: 0.8671 - val_loss: 0.4708 - val_acc: 0.8485
Epoch 5/10
30/30 [==============================] - 1s 23ms/step - loss: 0.4509 - acc: 0.8879 - val_loss: 0.4496 - val_acc: 0.8610
Epoch 6/10
30/30 [==============================] - 1s 26ms/step - loss: 0.4226 - acc: 0.9006 - val_loss: 0.4312 - val_acc: 0.8809
Epoch 7/10
30/30 [==============================] - 1s 23ms/step - loss: 0.3981 - acc: 0.9115 - val_loss: 0.4194 - val_acc: 0.8738
Epoch 8/10
30/30 [==============================] - 1s 25ms/step - loss: 0.3722 - acc: 0.9208 - val_loss: 0.4048 - val_acc: 0.8807
Epoch 9/10
30/30 [==============================] - 1s 25ms/step - loss: 0.3507 - acc: 0.9277 - val_loss: 0.3921 - val_acc: 0.8833
Epoch 10/10
30/30 [==============================] - 1s 23ms/step - loss: 0.3329 - acc: 0.9321 - val_loss: 0.3905 - val_acc: 0.8801
```

```
history_dict = history.history
history_dict.keys()
```

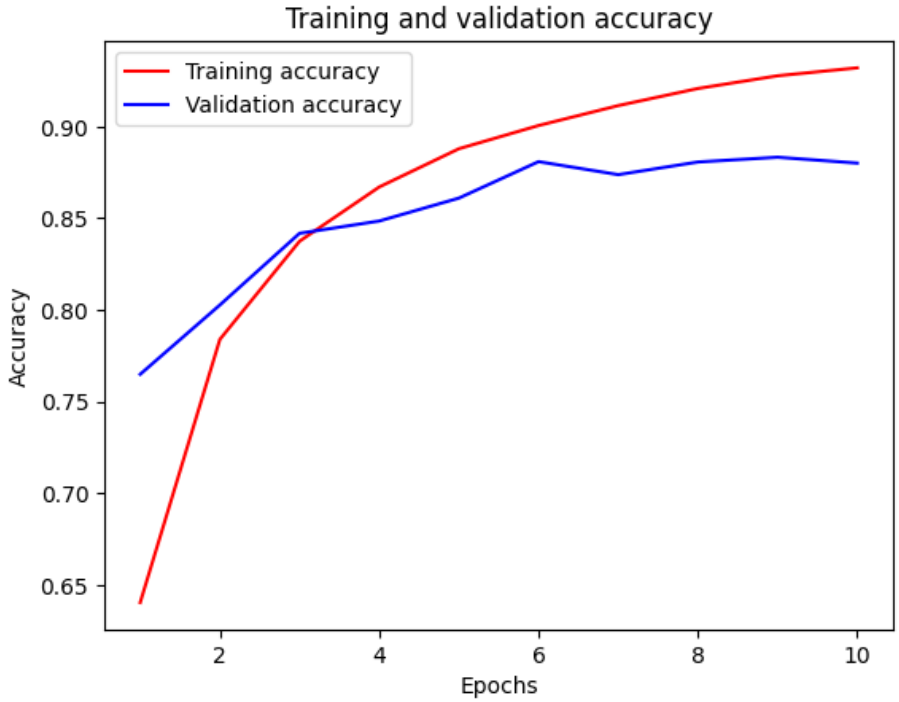
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Number of epochs 15

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=15,batch_size=512,validation_data=(x_val, y_val))
```

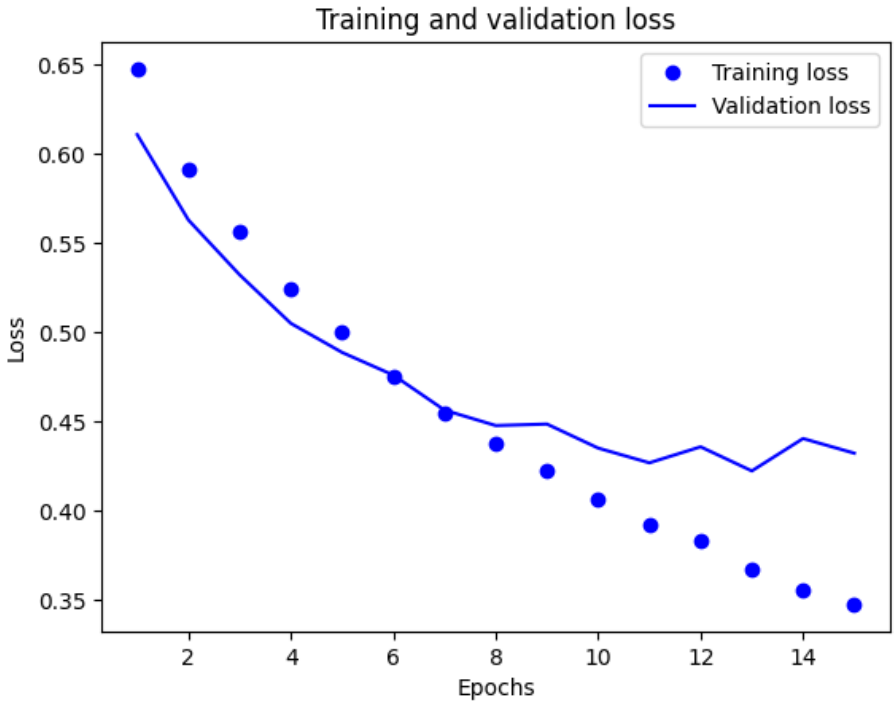
```
Epoch 1/15
30/30 [==============================] - 3s 71ms/step - loss: 0.6474 - acc: 0.6183 - val_loss: 0.6106 - val_acc: 0.7388
Epoch 2/15
30/30 [==============================] - 1s 25ms/step - loss: 0.5910 - acc: 0.7176 - val_loss: 0.5627 - val_acc: 0.7645
Epoch 3/15
30/30 [==============================] - 1s 26ms/step - loss: 0.5559 - acc: 0.7665 - val_loss: 0.5320 - val_acc: 0.7945
Epoch 4/15
30/30 [==============================] - 1s 22ms/step - loss: 0.5239 - acc: 0.8119 - val_loss: 0.5048 - val_acc: 0.8338
Epoch 5/15
30/30 [==============================] - 1s 24ms/step - loss: 0.4997 - acc: 0.8439 - val_loss: 0.4885 - val_acc: 0.8296
Epoch 6/15
30/30 [==============================] - 1s 23ms/step - loss: 0.4748 - acc: 0.8648 - val_loss: 0.4758 - val_acc: 0.8370
Epoch 7/15
30/30 [==============================] - 1s 25ms/step - loss: 0.4548 - acc: 0.8764 - val_loss: 0.4562 - val_acc: 0.8738
Epoch 8/15
30/30 [==============================] - 1s 26ms/step - loss: 0.4372 - acc: 0.8908 - val_loss: 0.4475 - val_acc: 0.8737
Epoch 9/15
30/30 [==============================] - 1s 25ms/step - loss: 0.4226 - acc: 0.8996 - val_loss: 0.4483 - val_acc: 0.8627
Epoch 10/15
30/30 [==============================] - 1s 26ms/step - loss: 0.4066 - acc: 0.9065 - val_loss: 0.4349 - val_acc: 0.8788
Epoch 11/15
30/30 [==============================] - 1s 23ms/step - loss: 0.3920 - acc: 0.9134 - val_loss: 0.4266 - val_acc: 0.8868
Epoch 12/15
30/30 [==============================] - 1s 23ms/step - loss: 0.3830 - acc: 0.9149 - val_loss: 0.4356 - val_acc: 0.8737
Epoch 13/15
30/30 [==============================] - 1s 32ms/step - loss: 0.3671 - acc: 0.9221 - val_loss: 0.4221 - val_acc: 0.8840
Epoch 14/15
30/30 [==============================] - 1s 32ms/step - loss: 0.3549 - acc: 0.9251 - val_loss: 0.4402 - val_acc: 0.8709
Epoch 15/15
30/30 [==============================] - 1s 39ms/step - loss: 0.3473 - acc: 0.9275 - val_loss: 0.4320 - val_acc: 0.8761
```

```
history_dict = history.history
history_dict.keys()
```

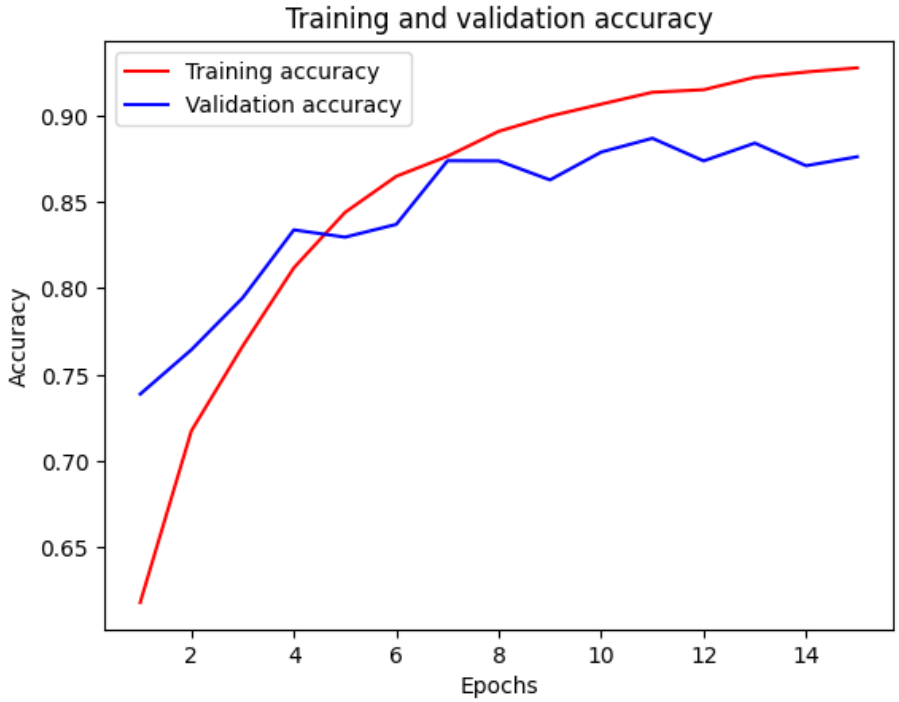
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```



```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Number of epochs 17

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=17,batch_size=512,validation_data=(x_val, y_val))
```

```
Epoch 1/17
30/30 [==============================] - 3s 70ms/step - loss: 0.6439 - acc: 0.6215 - val_loss: 0.5806 - val_acc: 0.7408
Epoch 2/17
30/30 [==============================] - 1s 26ms/step - loss: 0.5747 - acc: 0.7572 - val_loss: 0.5403 - val_acc: 0.8177
Epoch 3/17
30/30 [==============================] - 1s 24ms/step - loss: 0.5320 - acc: 0.8137 - val_loss: 0.5106 - val_acc: 0.8201
Epoch 4/17
30/30 [==============================] - 1s 25ms/step - loss: 0.4998 - acc: 0.8443 - val_loss: 0.4928 - val_acc: 0.8259
Epoch 5/17
30/30 [==============================] - 1s 23ms/step - loss: 0.4756 - acc: 0.8624 - val_loss: 0.4738 - val_acc: 0.8698
Epoch 6/17
30/30 [==============================] - 1s 23ms/step - loss: 0.4538 - acc: 0.8803 - val_loss: 0.4646 - val_acc: 0.8599
Epoch 7/17
30/30 [==============================] - 1s 25ms/step - loss: 0.4380 - acc: 0.8903 - val_loss: 0.4541 - val_acc: 0.8786
Epoch 8/17
30/30 [==============================] - 1s 29ms/step - loss: 0.4229 - acc: 0.8977 - val_loss: 0.4525 - val_acc: 0.8676
Epoch 9/17
30/30 [==============================] - 1s 44ms/step - loss: 0.4064 - acc: 0.9067 - val_loss: 0.4559 - val_acc: 0.8613
Epoch 10/17
30/30 [==============================] - 1s 43ms/step - loss: 0.3920 - acc: 0.9117 - val_loss: 0.4479 - val_acc: 0.8701
```

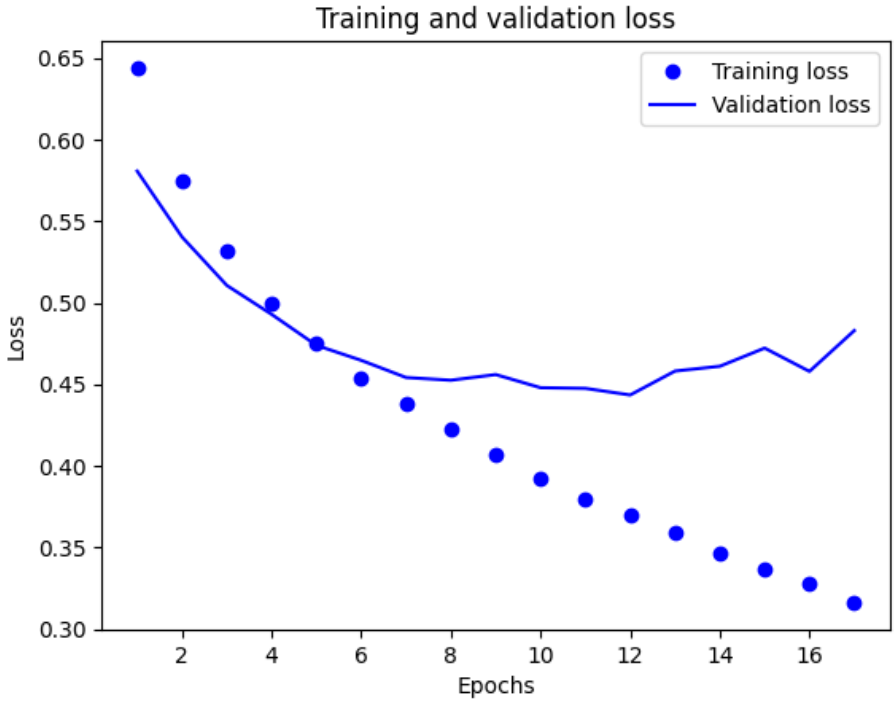
```
Epoch 11/17
30/30 [==============================] - 1s 24ms/step - loss: 0.3797 - acc: 0.9185 - val_loss: 0.4475 - val_acc: 0.8713
Epoch 12/17
30/30 [==============================] - 1s 23ms/step - loss: 0.3701 - acc: 0.9200 - val_loss: 0.4434 - val_acc: 0.8742
Epoch 13/17
30/30 [==============================] - 1s 25ms/step - loss: 0.3586 - acc: 0.9256 - val_loss: 0.4581 - val_acc: 0.8694
Epoch 14/17
30/30 [==============================] - 1s 23ms/step - loss: 0.3462 - acc: 0.9317 - val_loss: 0.4610 - val_acc: 0.8710
Epoch 15/17
30/30 [==============================] - 1s 23ms/step - loss: 0.3370 - acc: 0.9346 - val_loss: 0.4722 - val_acc: 0.8669
Epoch 16/17
30/30 [==============================] - 1s 24ms/step - loss: 0.3283 - acc: 0.9369 - val_loss: 0.4579 - val_acc: 0.8730
Epoch 17/17
30/30 [==============================] - 1s 23ms/step - loss: 0.3162 - acc: 0.9431 - val_loss: 0.4829 - val_acc: 0.8680
```

```
history_dict = history.history
history_dict.keys()
```

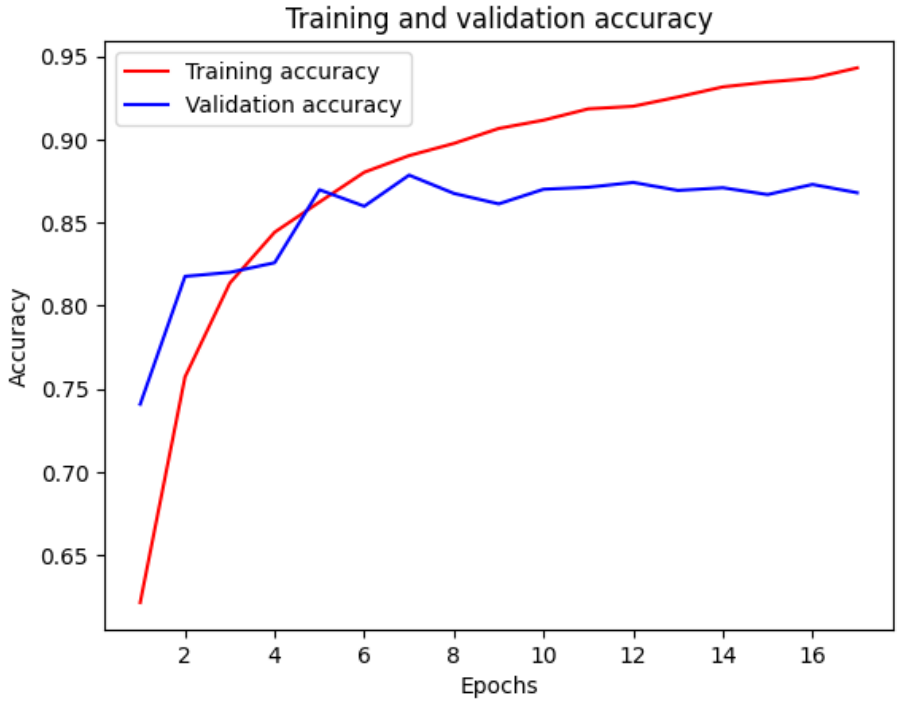
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Batch size 256

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=256,validation_data=(x_val, y_val))
```

```
Epoch 1/20
59/59 [==============================] - 4s 40ms/step - loss: 0.6499 - acc: 0.6100 - val_loss: 0.5847 - val_acc: 0.8409
Epoch 2/20
59/59 [==============================] - 1s 12ms/step - loss: 0.5678 - acc: 0.6793 - val_loss: 0.5062 - val_acc: 0.8670
Epoch 3/20
59/59 [==============================] - 1s 12ms/step - loss: 0.5135 - acc: 0.7055 - val_loss: 0.4463 - val_acc: 0.8772
Epoch 4/20
59/59 [==============================] - 1s 14ms/step - loss: 0.4705 - acc: 0.7226 - val_loss: 0.4207 - val_acc: 0.8765
Epoch 5/20
59/59 [==============================] - 1s 14ms/step - loss: 0.4433 - acc: 0.7339 - val_loss: 0.3835 - val_acc: 0.8869
Epoch 6/20
59/59 [==============================] - 1s 14ms/step - loss: 0.4185 - acc: 0.7459 - val_loss: 0.3455 - val_acc: 0.8874
Epoch 7/20
59/59 [==============================] - 1s 12ms/step - loss: 0.3990 - acc: 0.7649 - val_loss: 0.3278 - val_acc: 0.8891
Epoch 8/20
59/59 [==============================] - 1s 14ms/step - loss: 0.3833 - acc: 0.8507 - val_loss: 0.3111 - val_acc: 0.8882
Epoch 9/20
59/59 [==============================] - 1s 13ms/step - loss: 0.3602 - acc: 0.8617 - val_loss: 0.2961 - val_acc: 0.8883
Epoch 10/20
59/59 [==============================] - 1s 14ms/step - loss: 0.3346 - acc: 0.8695 - val_loss: 0.2871 - val_acc: 0.8878
Epoch 11/20
59/59 [==============================] - 1s 12ms/step - loss: 0.3197 - acc: 0.8721 - val_loss: 0.2862 - val_acc: 0.8860
Epoch 12/20
59/59 [==============================] - 1s 14ms/step - loss: 0.3056 - acc: 0.8758 - val_loss: 0.2857 - val_acc: 0.8865
Epoch 13/20
59/59 [==============================] - 1s 18ms/step - loss: 0.2922 - acc: 0.8783 - val_loss: 0.2858 - val_acc: 0.8861
Epoch 14/20
59/59 [==============================] - 1s 17ms/step - loss: 0.2763 - acc: 0.8833 - val_loss: 0.2854 - val_acc: 0.8861
Epoch 15/20
59/59 [==============================] - 1s 15ms/step - loss: 0.2687 - acc: 0.8853 - val_loss: 0.2918 - val_acc: 0.8856
Epoch 16/20
```

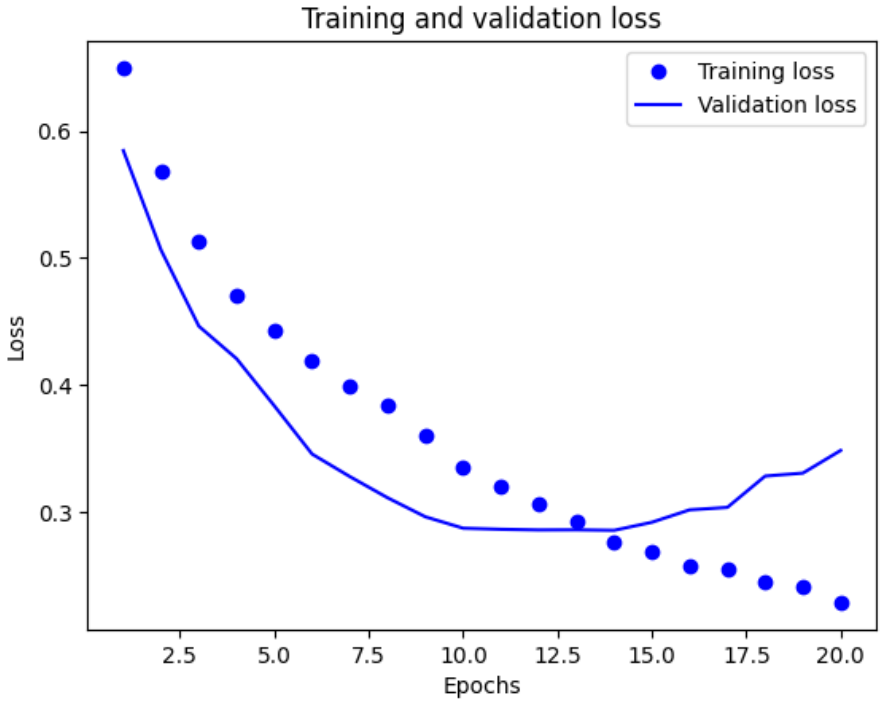
```
59/59 [==============================] - 1s 15ms/step - loss: 0.2570 - acc: 0.8911 - val_loss: 0.3016 - val_acc: 0.8854
Epoch 17/20
59/59 [==============================] - 1s 14ms/step - loss: 0.2554 - acc: 0.8875 - val_loss: 0.3035 - val_acc: 0.8830
Epoch 18/20
59/59 [==============================] - 1s 13ms/step - loss: 0.2450 - acc: 0.8948 - val_loss: 0.3282 - val_acc: 0.8839
Epoch 19/20
59/59 [==============================] - 1s 13ms/step - loss: 0.2415 - acc: 0.8945 - val_loss: 0.3305 - val_acc: 0.8816
Epoch 20/20
59/59 [==============================] - 1s 14ms/step - loss: 0.2286 - acc: 0.8985 - val_loss: 0.3484 - val_acc: 0.8812
```

```
history_dict = history.history
history_dict.keys()
```

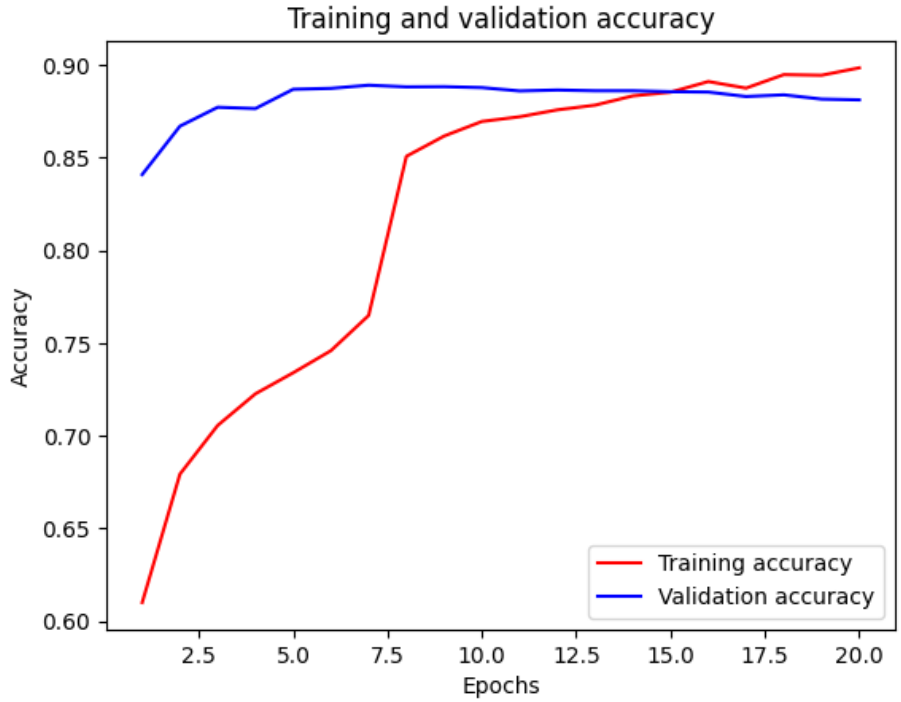
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Batch size 1024

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=1024,validation_data=(x_val, y_val))
```

```
Epoch 1/20
15/15 [==============================] - 9s 336ms/step - loss: 0.6681 - acc: 0.6020 - val_loss: 0.6155 - val_acc: 0.7305
Epoch 2/20
15/15 [==============================] - 1s 96ms/step - loss: 0.6085 - acc: 0.6874 - val_loss: 0.5509 - val_acc: 0.8408
Epoch 3/20
15/15 [==============================] - 1s 75ms/step - loss: 0.5681 - acc: 0.7149 - val_loss: 0.5075 - val_acc: 0.8447
Epoch 4/20
15/15 [==============================] - 1s 72ms/step - loss: 0.5366 - acc: 0.7429 - val_loss: 0.4714 - val_acc: 0.8621
Epoch 5/20
15/15 [==============================] - 1s 54ms/step - loss: 0.5082 - acc: 0.7708 - val_loss: 0.4467 - val_acc: 0.8611
Epoch 6/20
15/15 [==============================] - 1s 55ms/step - loss: 0.4857 - acc: 0.7895 - val_loss: 0.4191 - val_acc: 0.8712
Epoch 7/20
15/15 [==============================] - 1s 76ms/step - loss: 0.4631 - acc: 0.8030 - val_loss: 0.4035 - val_acc: 0.8703
Epoch 8/20
15/15 [==============================] - 1s 57ms/step - loss: 0.4446 - acc: 0.8151 - val_loss: 0.3829 - val_acc: 0.8757
Epoch 9/20
15/15 [==============================] - 1s 76ms/step - loss: 0.4279 - acc: 0.8283 - val_loss: 0.3745 - val_acc: 0.8763
Epoch 10/20
15/15 [==============================] - 1s 73ms/step - loss: 0.4130 - acc: 0.8405 - val_loss: 0.3586 - val_acc: 0.8760
Epoch 11/20
15/15 [==============================] - 1s 54ms/step - loss: 0.3950 - acc: 0.8517 - val_loss: 0.3421 - val_acc: 0.8851
Epoch 12/20
15/15 [==============================] - 1s 75ms/step - loss: 0.3809 - acc: 0.8603 - val_loss: 0.3401 - val_acc: 0.8839
Epoch 13/20
15/15 [==============================] - 1s 71ms/step - loss: 0.3725 - acc: 0.8686 - val_loss: 0.3252 - val_acc: 0.8869
Epoch 14/20
15/15 [==============================] - 1s 90ms/step - loss: 0.3538 - acc: 0.8807 - val_loss: 0.3224 - val_acc: 0.8840
Epoch 15/20
15/15 [==============================] - 1s 81ms/step - loss: 0.3419 - acc: 0.8838 - val_loss: 0.3106 - val_acc: 0.8868
Epoch 16/20
```

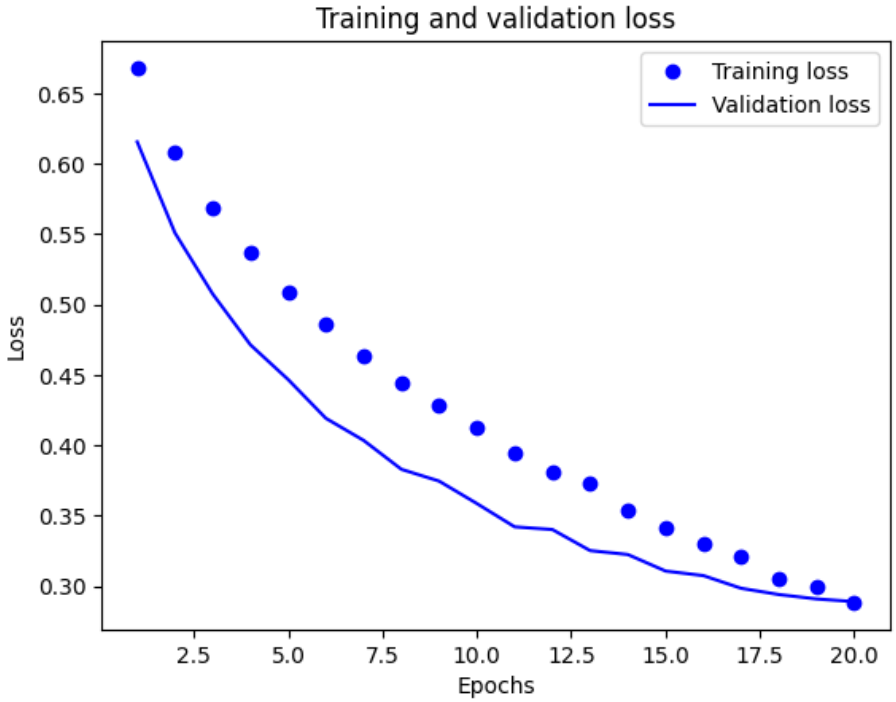
```
15/15 [==============================] - 1s 59ms/step - loss: 0.3302 - acc: 0.8937 - val_loss: 0.3074 - val_acc: 0.8887
Epoch 17/20
15/15 [==============================] - 1s 52ms/step - loss: 0.3205 - acc: 0.8971 - val_loss: 0.2984 - val_acc: 0.8875
Epoch 18/20
15/15 [==============================] - 1s 74ms/step - loss: 0.3057 - acc: 0.9021 - val_loss: 0.2940 - val_acc: 0.8883
Epoch 19/20
15/15 [==============================] - 1s 72ms/step - loss: 0.2994 - acc: 0.9034 - val_loss: 0.2908 - val_acc: 0.8875
Epoch 20/20
15/15 [==============================] - 1s 43ms/step - loss: 0.2883 - acc: 0.9070 - val_loss: 0.2889 - val_acc: 0.8868
```

```
history_dict = history.history
history_dict.keys()
```

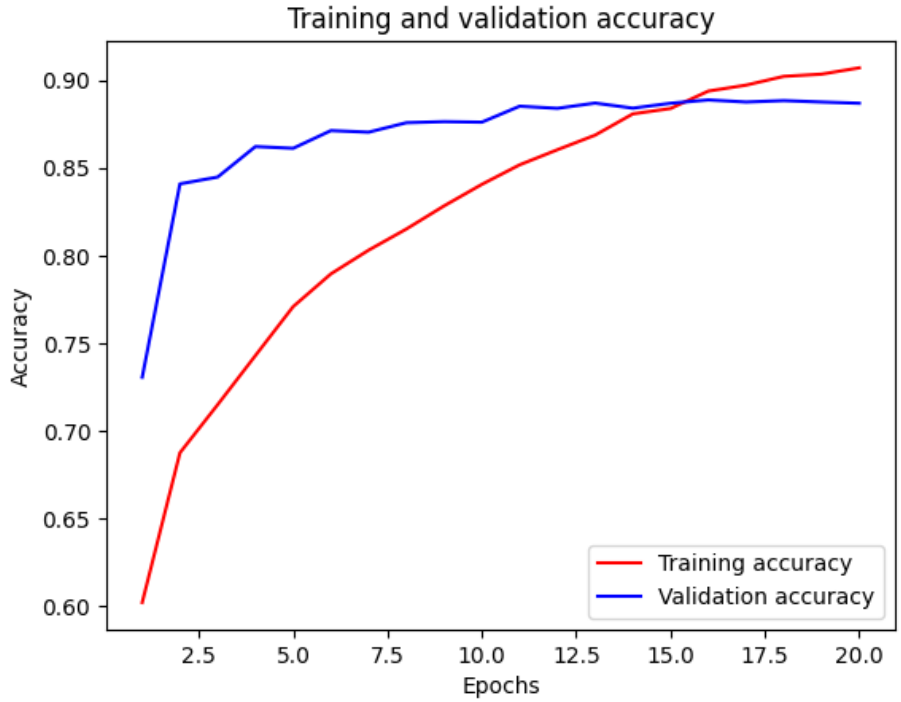
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Learning rate 0.0005

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.0005)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=512,validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 4s 86ms/step - loss: 0.6656 - acc: 0.5919 - val_loss: 0.6407 - val_acc: 0.7261
Epoch 2/20
30/30 [==============================] - 1s 25ms/step - loss: 0.6088 - acc: 0.7225 - val_loss: 0.5901 - val_acc: 0.7652
Epoch 3/20
30/30 [==============================] - 1s 40ms/step - loss: 0.5621 - acc: 0.7903 - val_loss: 0.5493 - val_acc: 0.7689
Epoch 4/20
30/30 [==============================] - 1s 45ms/step - loss: 0.5231 - acc: 0.8295 - val_loss: 0.5191 - val_acc: 0.8408
Epoch 5/20
30/30 [==============================] - 1s 26ms/step - loss: 0.4925 - acc: 0.8580 - val_loss: 0.4943 - val_acc: 0.8420
Epoch 6/20
30/30 [==============================] - 1s 26ms/step - loss: 0.4653 - acc: 0.8769 - val_loss: 0.4780 - val_acc: 0.8284
Epoch 7/20
30/30 [==============================] - 1s 24ms/step - loss: 0.4458 - acc: 0.8890 - val_loss: 0.4565 - val_acc: 0.8659
Epoch 8/20
30/30 [==============================] - 1s 23ms/step - loss: 0.4262 - acc: 0.9037 - val_loss: 0.4429 - val_acc: 0.8768
Epoch 9/20
30/30 [==============================] - 1s 38ms/step - loss: 0.4061 - acc: 0.9140 - val_loss: 0.4325 - val_acc: 0.8717
Epoch 10/20
30/30 [==============================] - 1s 31ms/step - loss: 0.3895 - acc: 0.9203 - val_loss: 0.4192 - val_acc: 0.8838
Epoch 11/20
30/30 [==============================] - 1s 42ms/step - loss: 0.3711 - acc: 0.9252 - val_loss: 0.4114 - val_acc: 0.8821
Epoch 12/20
30/30 [==============================] - 1s 26ms/step - loss: 0.3542 - acc: 0.9317 - val_loss: 0.4134 - val_acc: 0.8739
Epoch 13/20
30/30 [==============================] - 1s 24ms/step - loss: 0.3438 - acc: 0.9325 - val_loss: 0.4090 - val_acc: 0.8745
Epoch 14/20
30/30 [==============================] - 1s 26ms/step - loss: 0.3258 - acc: 0.9403 - val_loss: 0.3926 - val_acc: 0.8854
Epoch 15/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3103 - acc: 0.9457 - val_loss: 0.4051 - val_acc: 0.8755
Epoch 16/20
```

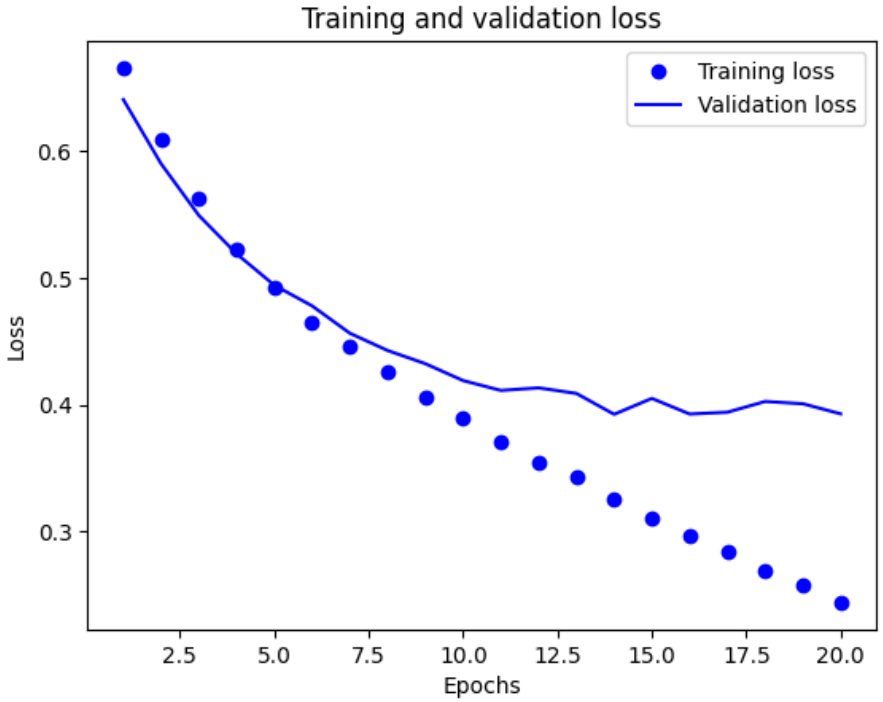
```
30/30 [==============================] - 1s 33ms/step - loss: 0.2965 - acc: 0.9463 - val_loss: 0.3928 - val_acc: 0.8815
Epoch 17/20
30/30 [==============================] - 1s 23ms/step - loss: 0.2841 - acc: 0.9477 - val_loss: 0.3942 - val_acc: 0.8811
Epoch 18/20
30/30 [==============================] - 1s 25ms/step - loss: 0.2688 - acc: 0.9532 - val_loss: 0.4027 - val_acc: 0.8779
Epoch 19/20
30/30 [==============================] - 1s 29ms/step - loss: 0.2575 - acc: 0.9545 - val_loss: 0.4009 - val_acc: 0.8792
Epoch 20/20
30/30 [==============================] - 1s 30ms/step - loss: 0.2443 - acc: 0.9579 - val_loss: 0.3929 - val_acc: 0.8808
```

```
history_dict = history.history
history_dict.keys()
```

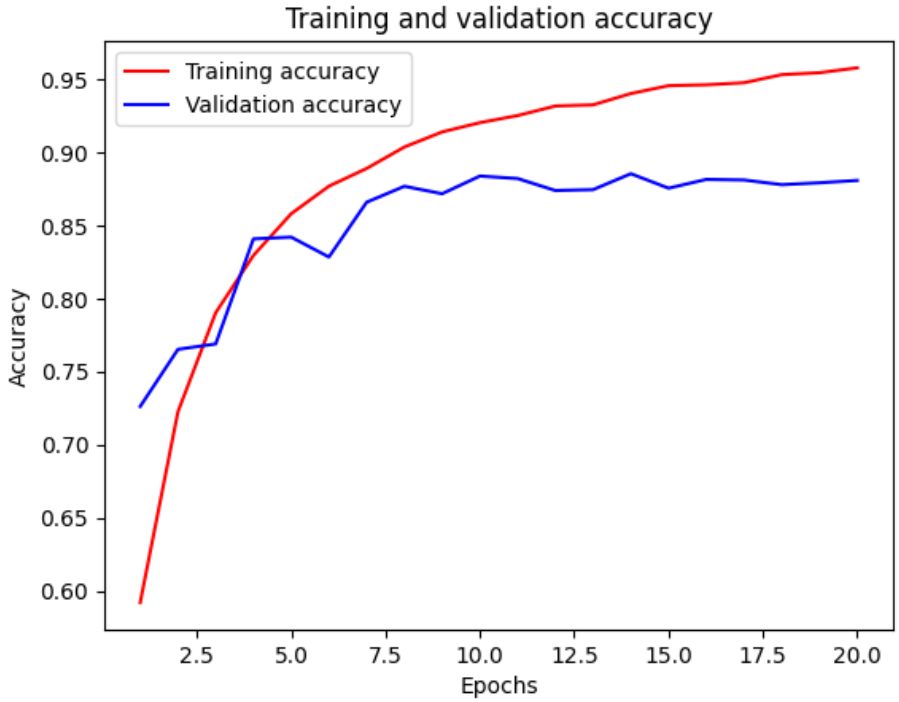
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Learning rate 0.002

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.002)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=512,validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 3s 83ms/step - loss: 0.6502 - acc: 0.6283 - val_loss: 0.5878 - val_acc: 0.8251
Epoch 2/20
30/30 [==============================] - 1s 26ms/step - loss: 0.5613 - acc: 0.7368 - val_loss: 0.5120 - val_acc: 0.8635
Epoch 3/20
30/30 [==============================] - 1s 23ms/step - loss: 0.5060 - acc: 0.7656 - val_loss: 0.4682 - val_acc: 0.8738
Epoch 4/20
30/30 [==============================] - 1s 24ms/step - loss: 0.4523 - acc: 0.8064 - val_loss: 0.4264 - val_acc: 0.8791
Epoch 5/20
30/30 [==============================] - 1s 26ms/step - loss: 0.4121 - acc: 0.8378 - val_loss: 0.3956 - val_acc: 0.8820
Epoch 6/20
30/30 [==============================] - 1s 25ms/step - loss: 0.3801 - acc: 0.8545 - val_loss: 0.3690 - val_acc: 0.8843
Epoch 7/20
30/30 [==============================] - 1s 25ms/step - loss: 0.3479 - acc: 0.8735 - val_loss: 0.3457 - val_acc: 0.8848
Epoch 8/20
30/30 [==============================] - 1s 24ms/step - loss: 0.3238 - acc: 0.8867 - val_loss: 0.3239 - val_acc: 0.8874
Epoch 9/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3044 - acc: 0.8959 - val_loss: 0.3116 - val_acc: 0.8868
Epoch 10/20
30/30 [==============================] - 1s 25ms/step - loss: 0.2843 - acc: 0.9004 - val_loss: 0.2975 - val_acc: 0.8881
Epoch 11/20
30/30 [==============================] - 1s 24ms/step - loss: 0.2665 - acc: 0.9149 - val_loss: 0.2890 - val_acc: 0.8880
Epoch 12/20
30/30 [==============================] - 1s 26ms/step - loss: 0.2530 - acc: 0.9164 - val_loss: 0.2850 - val_acc: 0.8892
Epoch 13/20
30/30 [==============================] - 1s 24ms/step - loss: 0.2377 - acc: 0.9211 - val_loss: 0.2826 - val_acc: 0.8878
Epoch 14/20
30/30 [==============================] - 1s 23ms/step - loss: 0.2281 - acc: 0.9237 - val_loss: 0.2814 - val_acc: 0.8883
Epoch 15/20
30/30 [==============================] - 1s 40ms/step - loss: 0.2146 - acc: 0.9276 - val_loss: 0.2801 - val_acc: 0.8895
Epoch 16/20
```

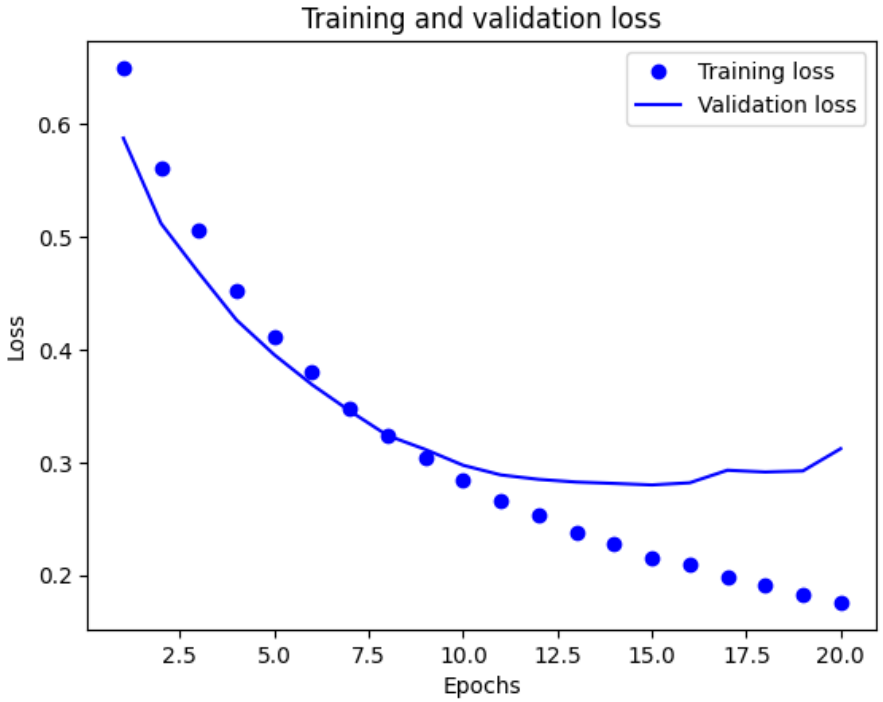
```
30/30 [==============================] - 1s 41ms/step - loss: 0.2093 - acc: 0.9293 - val_loss: 0.2820 - val_acc: 0.8885
Epoch 17/20
30/30 [==============================] - 1s 28ms/step - loss: 0.1989 - acc: 0.9329 - val_loss: 0.2931 - val_acc: 0.8856
Epoch 18/20
30/30 [==============================] - 1s 23ms/step - loss: 0.1915 - acc: 0.9353 - val_loss: 0.2915 - val_acc: 0.8878
Epoch 19/20
30/30 [==============================] - 1s 26ms/step - loss: 0.1823 - acc: 0.9389 - val_loss: 0.2925 - val_acc: 0.8874
Epoch 20/20
30/30 [==============================] - 1s 24ms/step - loss: 0.1758 - acc: 0.9419 - val_loss: 0.3123 - val_acc: 0.8862
```

```
history_dict = history.history
history_dict.keys()
```

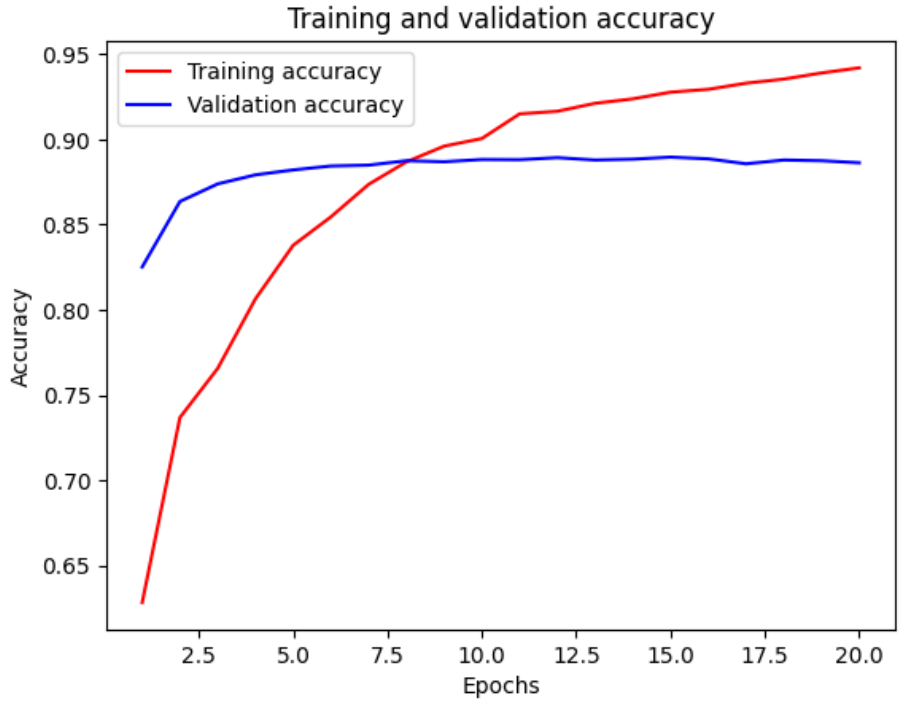
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Validation split 0.8

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x_val = x_train[20000:]
partial_x_train = x_train[:20000]
y_val = y_train[20000:]
partial_y_train = y_train[:20000]
```

```
x_val.shape
```

```
(5000, 10000)
```

```
partial_x_train.shape
```

```
(20000, 10000)
```

We are training the model:

```
model.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['acc'])
history = model.fit(partial_x_train,partial_y_train,epochs=20,batch_size=512,validation_data=(x_val, y_val))
```

```
Epoch 1/20
40/40 [==============================] - 5s 39ms/step - loss: 0.6253 - acc: 0.6536 - val_loss: 0.5679 - val_acc: 0.7210
Epoch 2/20
40/40 [==============================] - 1s 18ms/step - loss: 0.5414 - acc: 0.7893 - val_loss: 0.5268 - val_acc: 0.7412
Epoch 3/20
40/40 [==============================] - 1s 19ms/step - loss: 0.4990 - acc: 0.8431 - val_loss: 0.4936 - val_acc: 0.8472
Epoch 4/20
40/40 [==============================] - 1s 20ms/step - loss: 0.4696 - acc: 0.8737 - val_loss: 0.4758 - val_acc: 0.8586
Epoch 5/20
40/40 [==============================] - 1s 24ms/step - loss: 0.4448 - acc: 0.8904 - val_loss: 0.4612 - val_acc: 0.8726
Epoch 6/20
40/40 [==============================] - 1s 24ms/step - loss: 0.4249 - acc: 0.9050 - val_loss: 0.4617 - val_acc: 0.8508
Epoch 7/20
40/40 [==============================] - 1s 23ms/step - loss: 0.4044 - acc: 0.9131 - val_loss: 0.4434 - val_acc: 0.8758
Epoch 8/20
40/40 [==============================] - 1s 20ms/step - loss: 0.3890 - acc: 0.9196 - val_loss: 0.4375 - val_acc: 0.8790
Epoch 9/20
40/40 [==============================] - 1s 17ms/step - loss: 0.3729 - acc: 0.9265 - val_loss: 0.4367 - val_acc: 0.8728
Epoch 10/20
40/40 [==============================] - 1s 18ms/step - loss: 0.3580 - acc: 0.9344 - val_loss: 0.4384 - val_acc: 0.8760
Epoch 11/20
40/40 [==============================] - 1s 18ms/step - loss: 0.3452 - acc: 0.9382 - val_loss: 0.4555 - val_acc: 0.8660
Epoch 12/20
40/40 [==============================] - 1s 18ms/step - loss: 0.3320 - acc: 0.9408 - val_loss: 0.4346 - val_acc: 0.8778
Epoch 13/20
```

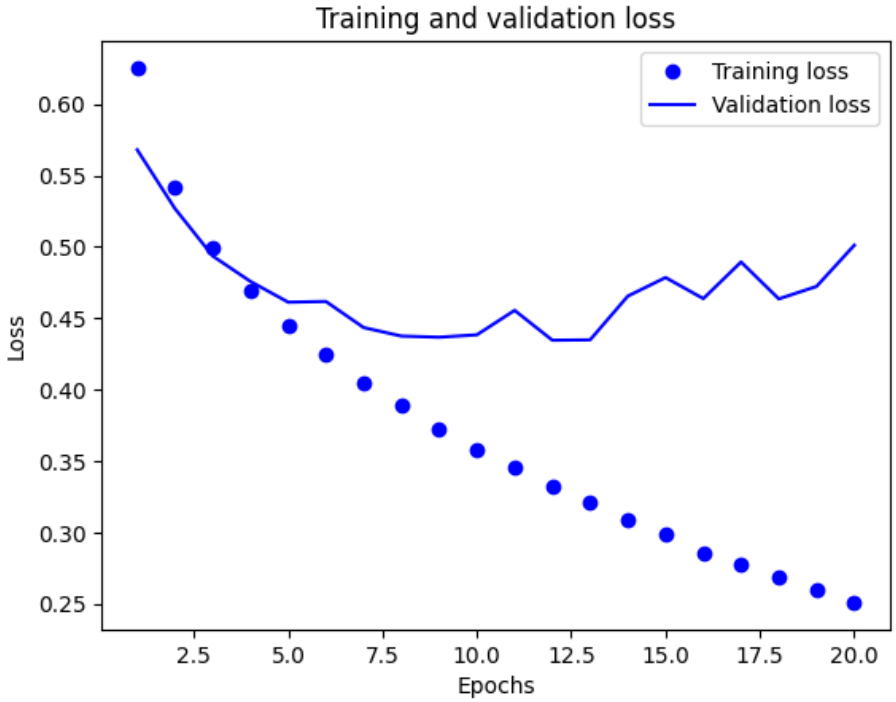
```
40/40 [==============================] - 1s 17ms/step - loss: 0.3209 - acc: 0.9456 - val_loss: 0.4348 - val_acc: 0.8792
Epoch 14/20
40/40 [==============================] - 1s 18ms/step - loss: 0.3089 - acc: 0.9500 - val_loss: 0.4655 - val_acc: 0.8710
Epoch 15/20
40/40 [==============================] - 1s 17ms/step - loss: 0.2982 - acc: 0.9512 - val_loss: 0.4785 - val_acc: 0.8674
Epoch 16/20
40/40 [==============================] - 1s 18ms/step - loss: 0.2856 - acc: 0.9562 - val_loss: 0.4637 - val_acc: 0.8748
Epoch 17/20
40/40 [==============================] - 1s 18ms/step - loss: 0.2776 - acc: 0.9560 - val_loss: 0.4894 - val_acc: 0.8694
Epoch 18/20
40/40 [==============================] - 1s 18ms/step - loss: 0.2691 - acc: 0.9560 - val_loss: 0.4635 - val_acc: 0.8748
Epoch 19/20
40/40 [==============================] - 1s 18ms/step - loss: 0.2595 - acc: 0.9600 - val_loss: 0.4723 - val_acc: 0.8776
Epoch 20/20
40/40 [==============================] - 1s 17ms/step - loss: 0.2509 - acc: 0.9620 - val_loss: 0.5011 - val_acc: 0.8720
```

```
history_dict = history.history
history_dict.keys()
```

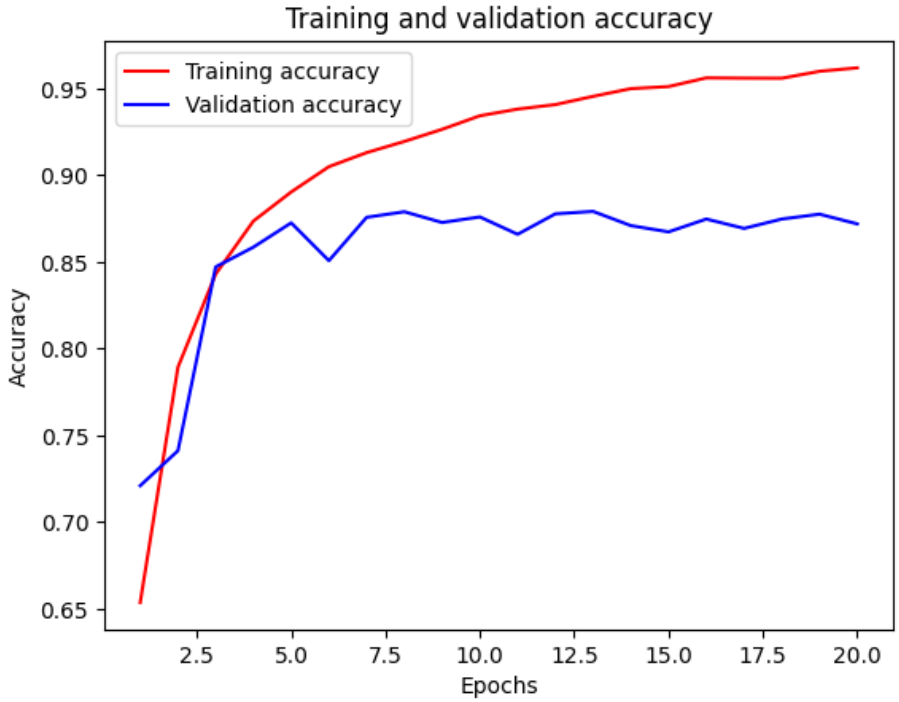
```
dict_keys(['loss', 'acc', 'val_loss', 'val_acc'])
```

Training and validation error:

```
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
acc=history_dict['acc']
val_acc=history_dict['val_acc']
plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



Validation split 0.7

```
x_val.size
```

```
50000000
```

```
partial_x_train.size
```

```
200000000
```

```
x_val.shape
```

```
(5000, 10000)
```

```
partial_x_train.shape
```

```
(20000, 10000)
```

We define the model:

```
model = models.Sequential()
model.add(layers.Dense(4, activation='relu', input_shape=(10000,)))
model.add(layers.Dropout(0.4))
model.add(layers.Dense(4, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Optimizer and model compilation:

```
opt=keras.optimizers.RMSprop(learning_rate=0.001)

model.compile(optimizer=opt,loss='binary_crossentropy',metrics=['accuracy'])
```

We create validation data from some of the training data:

```
x val = x train[17500:]
```