# Assignment_1

February 27, 2022

[129]:
```
# Importing the required libraries
```

[130]:
```
import matplotlib
import matplotlib.pyplot as plt
import numpy as np
from keras import models
from keras import layers
```

[131]:
```
# Loading the IMDB dataset using tensorflow and keras
```

[132]:
```
from tensorflow.keras.datasets import imdb
(train_data, train_labels), (test_data, test_labels) = imdb.load_data(
    num_words=10000)
```

[133]:
```
# Decoding reviews back to text
```

[134]:
```
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]])
```

[135]:
```
# Vectorized the dataset by Creating an all-zero matrix of shape␣
↪(len(sequences), dimension) and Sets specific indices of results[i] to 1s
```

[136]:
```
def vectorize_sequences(sequences, dimension=10000):
    results = np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        for j in sequence:
            results[i, j] = 1.
    return results
```

[137]:
```
# Vectorizing Training and Test Dataset
```

[138]:
```
x_train = vectorize_sequences(train_data)
x_test = vectorize_sequences(test_data)
y_train = np.asarray(train_labels).astype("float32")
y_test = np.asarray(test_labels).astype("float32")
```

[139]:
```
# Model defination
```

[140]:
```
from tensorflow import keras
model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

[141]:
```
# Model Compilation
```

[142]:
```
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
```

[143]:
```
# Setting aside a validation set
```

[144]:
```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

[145]:
```
# Model Training
```

[146]:
```
history = model.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))
```

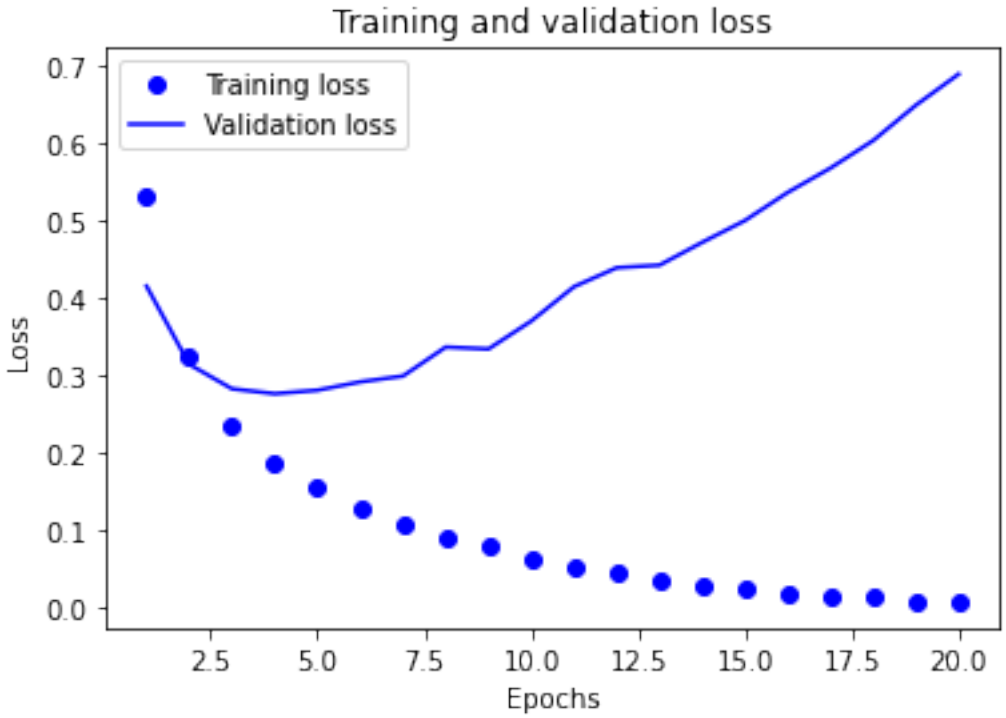
```
Epoch 1/20
30/30 [==============================] - 6s 152ms/step - loss: 0.5336 -
accuracy: 0.7844 - val_loss: 0.4159 - val_accuracy: 0.8505
Epoch 2/20
30/30 [==============================] - 1s 21ms/step - loss: 0.3238 - accuracy:
0.9024 - val_loss: 0.3150 - val_accuracy: 0.8853
Epoch 3/20
30/30 [==============================] - 1s 21ms/step - loss: 0.2369 - accuracy:
0.9251 - val_loss: 0.2834 - val_accuracy: 0.8895
Epoch 4/20
30/30 [==============================] - 1s 21ms/step - loss: 0.1892 - accuracy:
0.9381 - val_loss: 0.2771 - val_accuracy: 0.8895
Epoch 5/20
30/30 [==============================] - 1s 20ms/step - loss: 0.1550 - accuracy:
0.9509 - val_loss: 0.2816 - val_accuracy: 0.8860
Epoch 6/20
30/30 [==============================] - 1s 20ms/step - loss: 0.1299 - accuracy:
0.9589 - val_loss: 0.2924 - val_accuracy: 0.8856
```

```
Epoch 7/20
30/30 [==============================] - 1s 21ms/step - loss: 0.1093 - accuracy:
0.9677 - val_loss: 0.2999 - val_accuracy: 0.8843
Epoch 8/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0921 - accuracy:
0.9728 - val_loss: 0.3373 - val_accuracy: 0.8761
Epoch 9/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0792 - accuracy:
0.9777 - val_loss: 0.3348 - val_accuracy: 0.8812
Epoch 10/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0638 - accuracy:
0.9835 - val_loss: 0.3708 - val_accuracy: 0.8781
Epoch 11/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0522 - accuracy:
0.9873 - val_loss: 0.4151 - val_accuracy: 0.8732
Epoch 12/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0448 - accuracy:
0.9889 - val_loss: 0.4396 - val_accuracy: 0.8677
Epoch 13/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0374 - accuracy:
0.9920 - val_loss: 0.4429 - val_accuracy: 0.8730
Epoch 14/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0297 - accuracy:
0.9938 - val_loss: 0.4723 - val_accuracy: 0.8742
Epoch 15/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0247 - accuracy:
0.9956 - val_loss: 0.5005 - val_accuracy: 0.8731
Epoch 16/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0188 - accuracy:
0.9980 - val_loss: 0.5368 - val_accuracy: 0.8699
Epoch 17/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0163 - accuracy:
0.9979 - val_loss: 0.5681 - val_accuracy: 0.8705
Epoch 18/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0140 - accuracy:
0.9980 - val_loss: 0.6037 - val_accuracy: 0.8696
Epoch 19/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0101 - accuracy:
0.9987 - val_loss: 0.6494 - val_accuracy: 0.8637
Epoch 20/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0088 - accuracy:
0.9987 - val_loss: 0.6895 - val_accuracy: 0.8624
```

[147]:
```
# Plotting of Training and Validation Loss
```

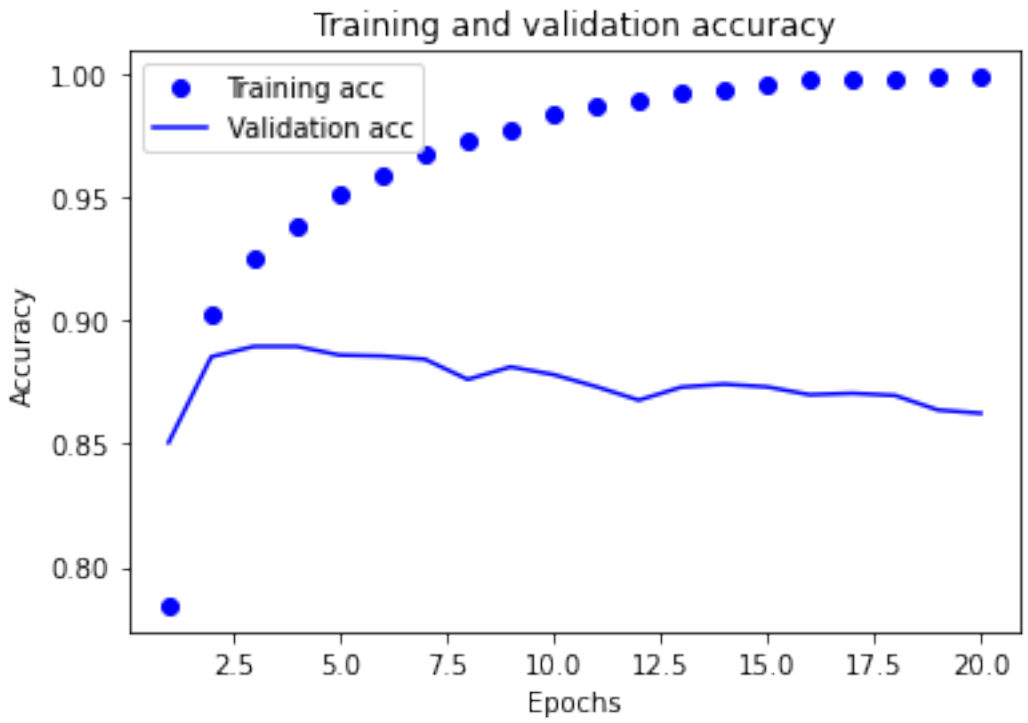
[148]:
```
history_dict = history.history
loss_values = history_dict["loss"]
```

```
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()
```




[149]:
```
#Plotting Training and Validation accuracy
```

[150]:
```
plt.clf()
acc = history_dict["accuracy"]
val_acc = history_dict["val_accuracy"]
plt.plot(epochs, acc, "bo", label="Training acc")
plt.plot(epochs, val_acc, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```




[151]:
```
# Retraining Model from begining
```

[152]:
```
model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
model.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
model.fit(x_train, y_train, epochs=4, batch_size=512)
results = model.evaluate(x_test, y_test)
```

```
Epoch 1/4
49/49 [==============================] - 2s 18ms/step - loss: 0.4724 - accuracy:
0.8111
Epoch 2/4
49/49 [==============================] - 1s 21ms/step - loss: 0.2698 - accuracy:
0.9076
Epoch 3/4
49/49 [==============================] - 1s 15ms/step - loss: 0.2050 - accuracy:
0.9284
Epoch 4/4
49/49 [==============================] - 1s 14ms/step - loss: 0.1725 - accuracy:
```

```
0.9391
782/782 [==============================] - 2s 1ms/step - loss: 0.2914 -
accuracy: 0.8844
```

[153]:
```
# ASSIGNMENT
# You used two hidden layers. Try using one or three hidden layers, and see how␣
 ↪doing so affects validation and test accuracy.
```

[154]:
```
model_1 = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])

model_2 = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

[155]:
```
model_1.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])

model_2.compile(optimizer="rmsprop",
              loss="binary_crossentropy",
              metrics=["accuracy"])
```

[156]:
```
# Model Training
```

[157]:
```
history_1 = model_1.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))

history_2 = model_2.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 3s 76ms/step - loss: 0.5439 - accuracy:
0.7541 - val_loss: 0.4034 - val_accuracy: 0.8453
Epoch 2/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3021 - accuracy:
0.9047 - val_loss: 0.3501 - val_accuracy: 0.8569
```

```
Epoch 3/20
30/30 [==============================] - 1s 21ms/step - loss: 0.2168 - accuracy:
0.9280 - val_loss: 0.3633 - val_accuracy: 0.8541
Epoch 4/20
30/30 [==============================] - 1s 21ms/step - loss: 0.1627 - accuracy:
0.9471 - val_loss: 0.3552 - val_accuracy: 0.8654
Epoch 5/20
30/30 [==============================] - 1s 22ms/step - loss: 0.1346 - accuracy:
0.9557 - val_loss: 0.3249 - val_accuracy: 0.8744
Epoch 6/20
30/30 [==============================] - 1s 20ms/step - loss: 0.1094 - accuracy:
0.9668 - val_loss: 0.3214 - val_accuracy: 0.8822
Epoch 7/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0888 - accuracy:
0.9732 - val_loss: 0.3395 - val_accuracy: 0.8804
Epoch 8/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0688 - accuracy:
0.9811 - val_loss: 0.3799 - val_accuracy: 0.8782
Epoch 9/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0643 - accuracy:
0.9822 - val_loss: 0.4152 - val_accuracy: 0.8771
Epoch 10/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0533 - accuracy:
0.9860 - val_loss: 0.4221 - val_accuracy: 0.8769
Epoch 11/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0374 - accuracy:
0.9917 - val_loss: 0.5415 - val_accuracy: 0.8642
Epoch 12/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0254 - accuracy:
0.9967 - val_loss: 0.6101 - val_accuracy: 0.8609
Epoch 13/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0265 - accuracy:
0.9943 - val_loss: 0.5272 - val_accuracy: 0.8730
Epoch 14/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0277 - accuracy:
0.9927 - val_loss: 0.5554 - val_accuracy: 0.8690
Epoch 15/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0118 - accuracy:
0.9986 - val_loss: 0.6204 - val_accuracy: 0.8689
Epoch 16/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0185 - accuracy:
0.9952 - val_loss: 0.6552 - val_accuracy: 0.8684
Epoch 17/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0066 - accuracy:
0.9991 - val_loss: 0.7529 - val_accuracy: 0.8661
Epoch 18/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0209 - accuracy:
0.9947 - val_loss: 0.7473 - val_accuracy: 0.8661
```

```
Epoch 19/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0031 - accuracy:
0.9996 - val_loss: 0.7709 - val_accuracy: 0.8658
Epoch 20/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0161 - accuracy:
0.9954 - val_loss: 0.8077 - val_accuracy: 0.8643
Epoch 1/20
30/30 [==============================] - 2s 34ms/step - loss: 0.5036 - accuracy:
0.7919 - val_loss: 0.3921 - val_accuracy: 0.8660
Epoch 2/20
30/30 [==============================] - 1s 22ms/step - loss: 0.3197 - accuracy:
0.9040 - val_loss: 0.3221 - val_accuracy: 0.8832
Epoch 3/20
30/30 [==============================] - 1s 23ms/step - loss: 0.2477 - accuracy:
0.9235 - val_loss: 0.2945 - val_accuracy: 0.8888
Epoch 4/20
30/30 [==============================] - 1s 24ms/step - loss: 0.2026 - accuracy:
0.9376 - val_loss: 0.2783 - val_accuracy: 0.8917
Epoch 5/20
30/30 [==============================] - 1s 21ms/step - loss: 0.1732 - accuracy:
0.9454 - val_loss: 0.2790 - val_accuracy: 0.8892
Epoch 6/20
30/30 [==============================] - 1s 20ms/step - loss: 0.1491 - accuracy:
0.9559 - val_loss: 0.2754 - val_accuracy: 0.8901
Epoch 7/20
30/30 [==============================] - 1s 21ms/step - loss: 0.1311 - accuracy:
0.9619 - val_loss: 0.2857 - val_accuracy: 0.8844
Epoch 8/20
30/30 [==============================] - 1s 21ms/step - loss: 0.1153 - accuracy:
0.9681 - val_loss: 0.2870 - val_accuracy: 0.8864
Epoch 9/20
30/30 [==============================] - 1s 21ms/step - loss: 0.1010 - accuracy:
0.9736 - val_loss: 0.3017 - val_accuracy: 0.8867
Epoch 10/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0903 - accuracy:
0.9769 - val_loss: 0.3122 - val_accuracy: 0.8848
Epoch 11/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0806 - accuracy:
0.9795 - val_loss: 0.3174 - val_accuracy: 0.8836
Epoch 12/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0703 - accuracy:
0.9835 - val_loss: 0.3344 - val_accuracy: 0.8835
Epoch 13/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0615 - accuracy:
0.9871 - val_loss: 0.3497 - val_accuracy: 0.8768
Epoch 14/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0551 - accuracy:
0.9878 - val_loss: 0.3610 - val_accuracy: 0.8780
```

```
Epoch 15/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0487 - accuracy:
0.9901 - val_loss: 0.3786 - val_accuracy: 0.8790
Epoch 16/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0427 - accuracy:
0.9923 - val_loss: 0.4054 - val_accuracy: 0.8694
Epoch 17/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0376 - accuracy:
0.9935 - val_loss: 0.4157 - val_accuracy: 0.8724
Epoch 18/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0326 - accuracy:
0.9949 - val_loss: 0.4305 - val_accuracy: 0.8755
Epoch 19/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0293 - accuracy:
0.9957 - val_loss: 0.4481 - val_accuracy: 0.8722
Epoch 20/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0251 - accuracy:
0.9970 - val_loss: 0.4673 - val_accuracy: 0.8711
```

[158]:
```
model_1.summary()
model_2.summary()
```

```
Model: "sequential_18"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_54 (Dense)            (None, 16)                160016

 dense_55 (Dense)            (None, 16)                272

 dense_56 (Dense)            (None, 16)                272

 dense_57 (Dense)            (None, 1)                 17

=================================================================
Total params: 160,577
Trainable params: 160,577
Non-trainable params: 0
_________________________________________________________________
Model: "sequential_19"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_58 (Dense)            (None, 16)                160016

 dense_59 (Dense)            (None, 1)                 17

=================================================================
```

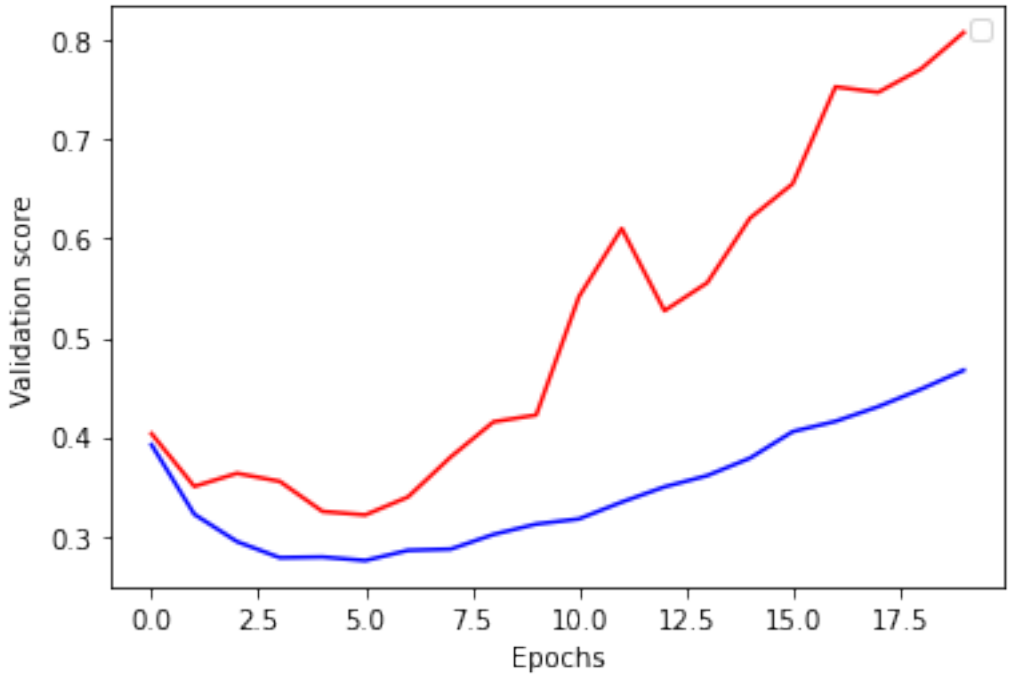
```
Total params: 160,033
Trainable params: 160,033
Non-trainable params: 0
_________________________________________________________________
```

[159]:
```
# Plotting the training and validation loss
```

[160]:
```
history_dict_1 = history_1.history
history_dict_2 = history_2.history

plt.plot(history_1.history['val_loss'], 'r', history_2.history['val_loss'], 'b')
plt.xlabel('Epochs')
plt.ylabel('Validation score')
plt.legend()
```

```
No handles with labels found to put in legend.
```

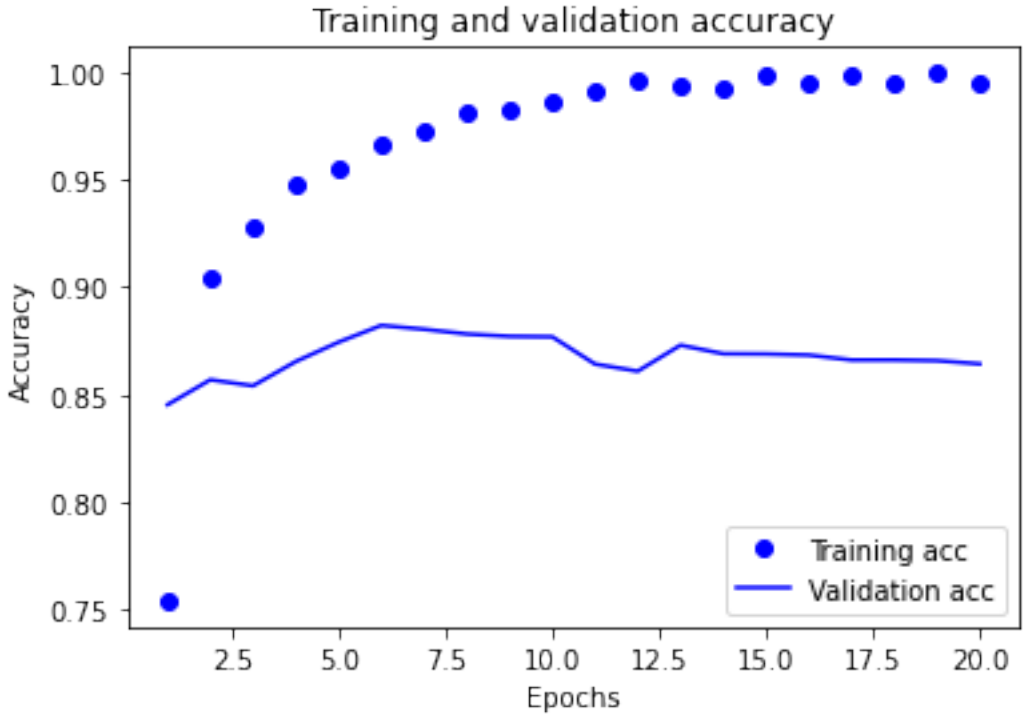
[160]:
```
<matplotlib.legend.Legend at 0x193c1002df0>
```




[161]:
```
# Plotting Training and Validation accuracy
```

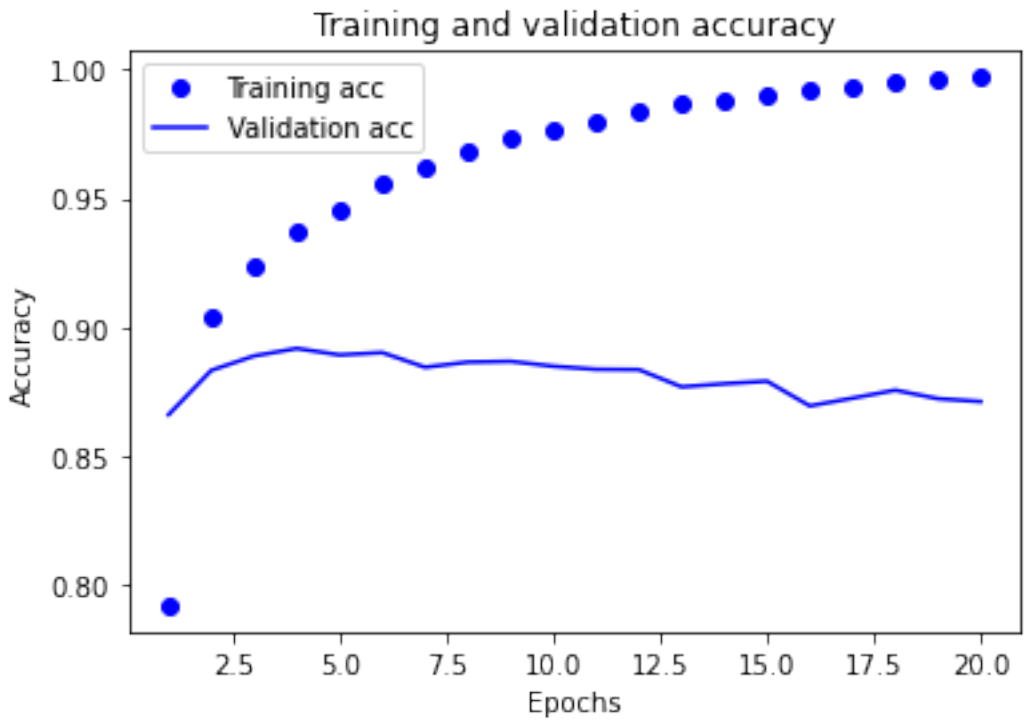
[162]:
```
plt.clf()
acc = history_dict_1["accuracy"]
val_acc = history_dict_1["val_accuracy"]
plt.plot(epochs, acc, "bo", label="Training acc")
```

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




[163]:
```
# plot_loss
# (history_dict_1.history['loss'], history_dict_1.history['val_loss'])
```

[164]:
```
plt.clf()
acc = history_dict_2["accuracy"]
val_acc = history_dict_2["val_accuracy"]
plt.plot(epochs, acc, "bo", label="Training acc")
plt.plot(epochs, val_acc, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```




[165]:
```
# 2.Try using layers with more hidden units or fewer hidden units: 32 units, 64␣
↪units, and so on.
```

[166]:
```
model_3 = keras.Sequential([
    layers.Dense(32, activation="relu"),
    layers.Dense(64, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

[167]:
```
model_3.compile(optimizer="rmsprop",
                loss="binary_crossentropy",
                metrics=["accuracy"])
```

[168]:
```
history_3 = model_3.fit(partial_x_train,
                        partial_y_train,
                        epochs=20,
                        batch_size=512,
                        validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 2s 37ms/step - loss: 0.4925 - accuracy:
0.7789 - val_loss: 0.3674 - val_accuracy: 0.8545
Epoch 2/20
30/30 [==============================] - 1s 31ms/step - loss: 0.2745 - accuracy:
```

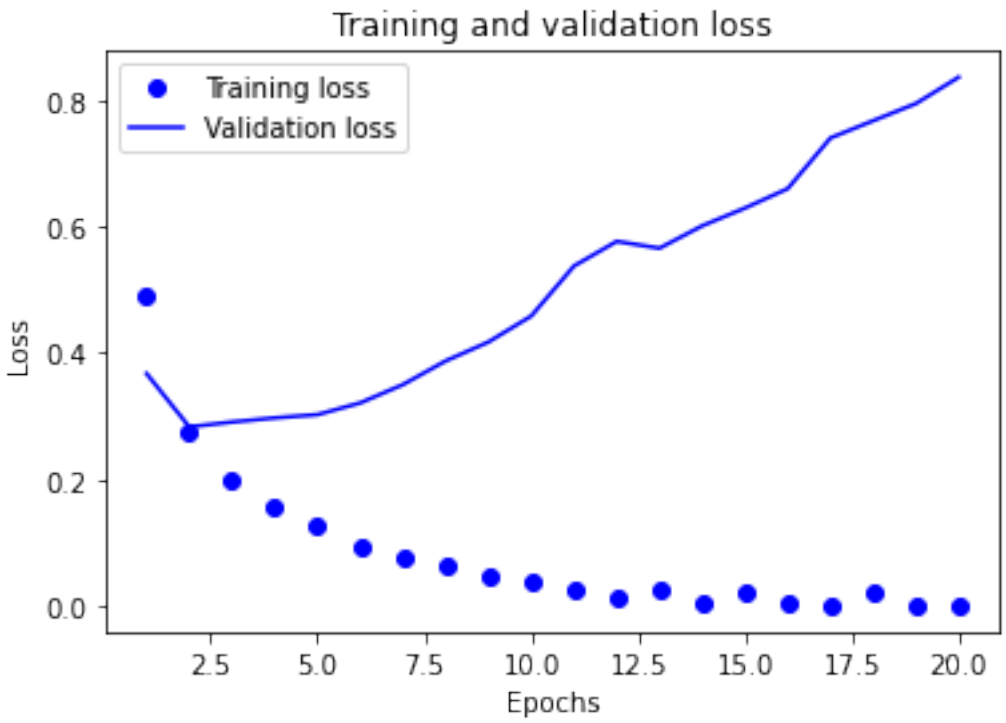
```
0.9039 - val_loss: 0.2831 - val_accuracy: 0.8924
Epoch 3/20
30/30 [==============================] - 1s 23ms/step - loss: 0.1991 - accuracy:
0.9304 - val_loss: 0.2907 - val_accuracy: 0.8840
Epoch 4/20
30/30 [==============================] - 1s 28ms/step - loss: 0.1546 - accuracy:
0.9467 - val_loss: 0.2973 - val_accuracy: 0.8829
Epoch 5/20
30/30 [==============================] - 1s 33ms/step - loss: 0.1258 - accuracy:
0.9565 - val_loss: 0.3024 - val_accuracy: 0.8858
Epoch 6/20
30/30 [==============================] - 1s 31ms/step - loss: 0.0943 - accuracy:
0.9707 - val_loss: 0.3209 - val_accuracy: 0.8830
Epoch 7/20
30/30 [==============================] - 1s 44ms/step - loss: 0.0768 - accuracy:
0.9774 - val_loss: 0.3499 - val_accuracy: 0.8803
Epoch 8/20
30/30 [==============================] - 1s 32ms/step - loss: 0.0622 - accuracy:
0.9813 - val_loss: 0.3874 - val_accuracy: 0.8802
Epoch 9/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0457 - accuracy:
0.9873 - val_loss: 0.4173 - val_accuracy: 0.8782
Epoch 10/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0382 - accuracy:
0.9883 - val_loss: 0.4583 - val_accuracy: 0.8743
Epoch 11/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0253 - accuracy:
0.9943 - val_loss: 0.5374 - val_accuracy: 0.8628
Epoch 12/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0140 - accuracy:
0.9982 - val_loss: 0.5763 - val_accuracy: 0.8662
Epoch 13/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0259 - accuracy:
0.9916 - val_loss: 0.5655 - val_accuracy: 0.8716
Epoch 14/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0053 - accuracy:
0.9999 - val_loss: 0.6009 - val_accuracy: 0.8723
Epoch 15/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0228 - accuracy:
0.9933 - val_loss: 0.6289 - val_accuracy: 0.8692
Epoch 16/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0029 - accuracy:
0.9999 - val_loss: 0.6596 - val_accuracy: 0.8703
Epoch 17/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0021 - accuracy:
0.9999 - val_loss: 0.7398 - val_accuracy: 0.8666
Epoch 18/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0211 - accuracy:
```

```
0.9948 - val_loss: 0.7671 - val_accuracy: 0.8675
Epoch 19/20
30/30 [==============================] - 1s 30ms/step - loss: 9.4256e-04 -
accuracy: 0.9999 - val_loss: 0.7942 - val_accuracy: 0.8680
Epoch 20/20
30/30 [==============================] - 1s 24ms/step - loss: 7.2426e-04 -
accuracy: 0.9999 - val_loss: 0.8358 - val_accuracy: 0.8663
```

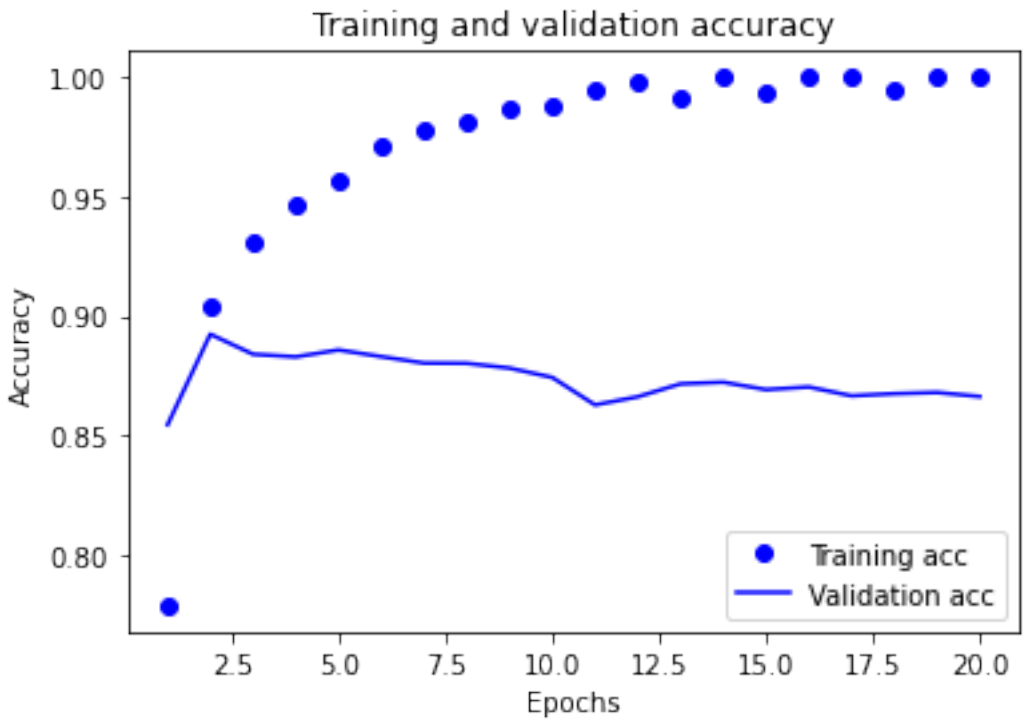
[169]:
```
model_3.summary()
```

```
Model: "sequential_20"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_60 (Dense)            (None, 32)                320032

 dense_61 (Dense)            (None, 64)                2112

 dense_62 (Dense)            (None, 1)                 65

=================================================================
Total params: 322,209
Trainable params: 322,209
Non-trainable params: 0
_________________________________________________________________
```

[170]:
```
history_dict_3 = history_3.history
loss_values = history_dict_3["loss"]
val_loss_values = history_dict_3["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()
```




[171]:
```
plt.clf()
acc = history_dict_3["accuracy"]
val_acc = history_dict_3["val_accuracy"]
plt.plot(epochs, acc, "bo", label="Training acc")
plt.plot(epochs, val_acc, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```




[172]:
```
# 3.Try using the 'mse' loss function instead of 'binary_crossentropy
```

[173]:
```
model_4 = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

[174]:
```
model_4.compile(optimizer="rmsprop",
                loss="mse",
                metrics=["accuracy"])
```

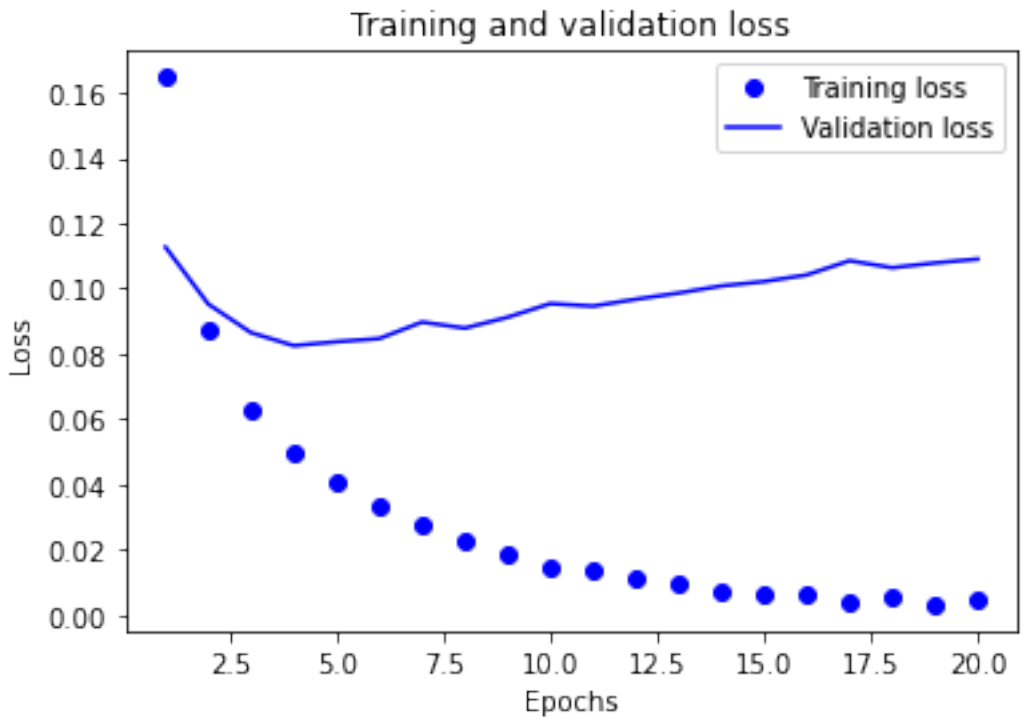
[175]:
```
# Training your model

history_4 = model_4.fit(partial_x_train,
                        partial_y_train,
                        epochs=20,
                        batch_size=512,
                        validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 2s 36ms/step - loss: 0.1648 - accuracy:
0.7859 - val_loss: 0.1126 - val_accuracy: 0.8724
Epoch 2/20
```

```
30/30 [==============================] - 1s 25ms/step - loss: 0.0873 - accuracy:
0.9042 - val_loss: 0.0952 - val_accuracy: 0.8814
Epoch 3/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0627 - accuracy:
0.9317 - val_loss: 0.0864 - val_accuracy: 0.8877
Epoch 4/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0495 - accuracy:
0.9463 - val_loss: 0.0824 - val_accuracy: 0.8891
Epoch 5/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0405 - accuracy:
0.9571 - val_loss: 0.0836 - val_accuracy: 0.8864
Epoch 6/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0332 - accuracy:
0.9661 - val_loss: 0.0846 - val_accuracy: 0.8842
Epoch 7/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0279 - accuracy:
0.9735 - val_loss: 0.0897 - val_accuracy: 0.8764
Epoch 8/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0223 - accuracy:
0.9801 - val_loss: 0.0878 - val_accuracy: 0.8818
Epoch 9/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0187 - accuracy:
0.9841 - val_loss: 0.0911 - val_accuracy: 0.8805
Epoch 10/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0146 - accuracy:
0.9886 - val_loss: 0.0953 - val_accuracy: 0.8726
Epoch 11/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0134 - accuracy:
0.9896 - val_loss: 0.0946 - val_accuracy: 0.8779
Epoch 12/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0110 - accuracy:
0.9913 - val_loss: 0.0967 - val_accuracy: 0.8768
Epoch 13/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0096 - accuracy:
0.9923 - val_loss: 0.0986 - val_accuracy: 0.8745
Epoch 14/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0072 - accuracy:
0.9947 - val_loss: 0.1007 - val_accuracy: 0.8703
Epoch 15/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0067 - accuracy:
0.9948 - val_loss: 0.1021 - val_accuracy: 0.8723
Epoch 16/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0061 - accuracy:
0.9950 - val_loss: 0.1041 - val_accuracy: 0.8680
Epoch 17/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0039 - accuracy:
0.9971 - val_loss: 0.1085 - val_accuracy: 0.8672
Epoch 18/20
```

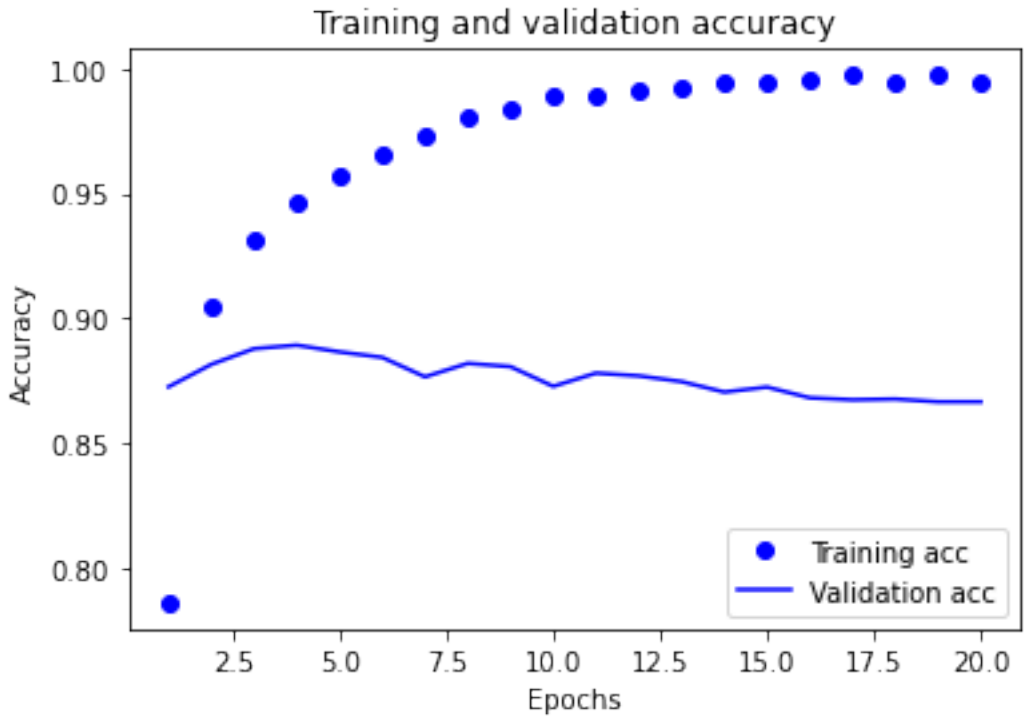
```
30/30 [==============================] - 1s 28ms/step - loss: 0.0054 - accuracy:
0.9949 - val_loss: 0.1063 - val_accuracy: 0.8675
Epoch 19/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0031 - accuracy:
0.9974 - val_loss: 0.1078 - val_accuracy: 0.8664
Epoch 20/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0048 - accuracy:
0.9949 - val_loss: 0.1090 - val_accuracy: 0.8664
```

[176]:
```
history_dict_4 = history_4.history
loss_values = history_dict_4["loss"]
val_loss_values = history_dict_4["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()
```




[ ]:
```
# Plotting the training and validation accuracy
```

[177]:
```
plt.clf()
acc = history_dict_4["accuracy"]
val_acc = history_dict_4["val_accuracy"]
plt.plot(epochs, acc, "bo", label="Training acc")
plt.plot(epochs, val_acc, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```




[178]:
```
# 4.Try using the tanh activation (an activation that was popular in the early␣
↪days of neural networks) instead of 'relu'
```

[179]:
```
model_5 = keras.Sequential([
    layers.Dense(16, activation="tanh"),
    layers.Dense(16, activation="tanh"),
    layers.Dense(1, activation="sigmoid")
])
```

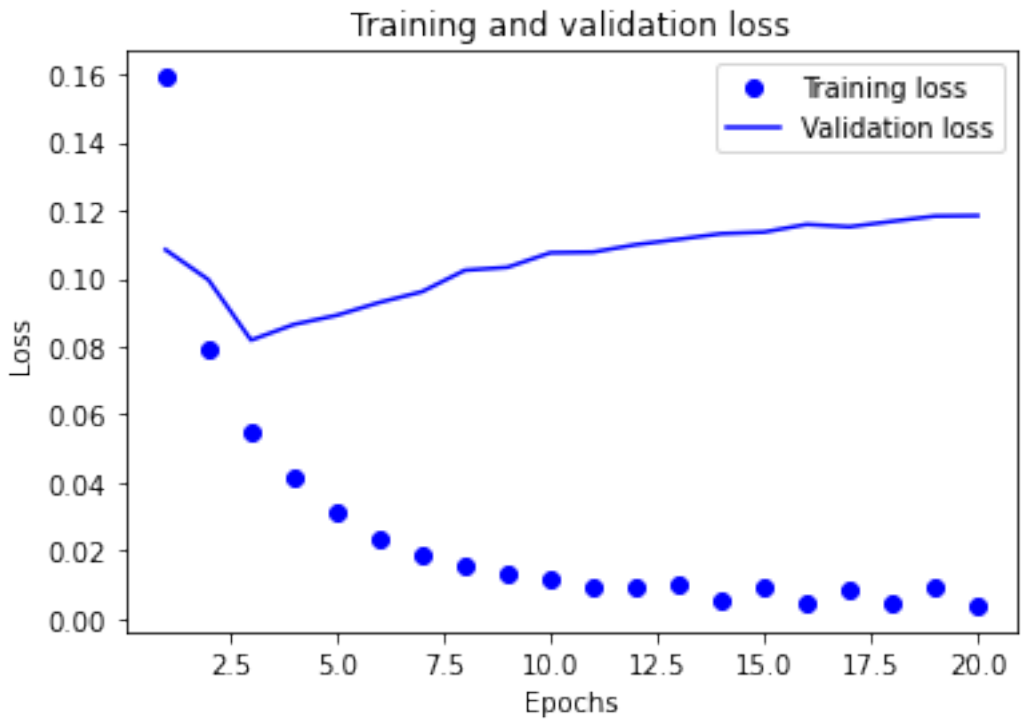
[180]:
```
model_5.compile(optimizer="rmsprop",
                loss="mse",
                metrics=["accuracy"])
```

[181]:
```
history_5 = model_5.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))
```

```
Epoch 1/20
30/30 [==============================] - 2s 46ms/step - loss: 0.1591 - accuracy:
0.7988 - val_loss: 0.1084 - val_accuracy: 0.8737
Epoch 2/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0791 - accuracy:
0.9123 - val_loss: 0.0996 - val_accuracy: 0.8643
Epoch 3/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0549 - accuracy:
0.9363 - val_loss: 0.0819 - val_accuracy: 0.8910
Epoch 4/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0413 - accuracy:
0.9527 - val_loss: 0.0865 - val_accuracy: 0.8824
Epoch 5/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0311 - accuracy:
0.9643 - val_loss: 0.0891 - val_accuracy: 0.8789
Epoch 6/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0233 - accuracy:
0.9743 - val_loss: 0.0929 - val_accuracy: 0.8805
Epoch 7/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0189 - accuracy:
0.9797 - val_loss: 0.0961 - val_accuracy: 0.8770
Epoch 8/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0153 - accuracy:
0.9839 - val_loss: 0.1024 - val_accuracy: 0.8750
Epoch 9/20
30/30 [==============================] - 1s 22ms/step - loss: 0.0135 - accuracy:
0.9856 - val_loss: 0.1032 - val_accuracy: 0.8756
Epoch 10/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0115 - accuracy:
0.9881 - val_loss: 0.1075 - val_accuracy: 0.8722
Epoch 11/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0093 - accuracy:
0.9901 - val_loss: 0.1077 - val_accuracy: 0.8733
Epoch 12/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0091 - accuracy:
0.9901 - val_loss: 0.1099 - val_accuracy: 0.8712
Epoch 13/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0101 - accuracy:
0.9885 - val_loss: 0.1115 - val_accuracy: 0.8711
Epoch 14/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0054 - accuracy:
0.9947 - val_loss: 0.1131 - val_accuracy: 0.8700
```

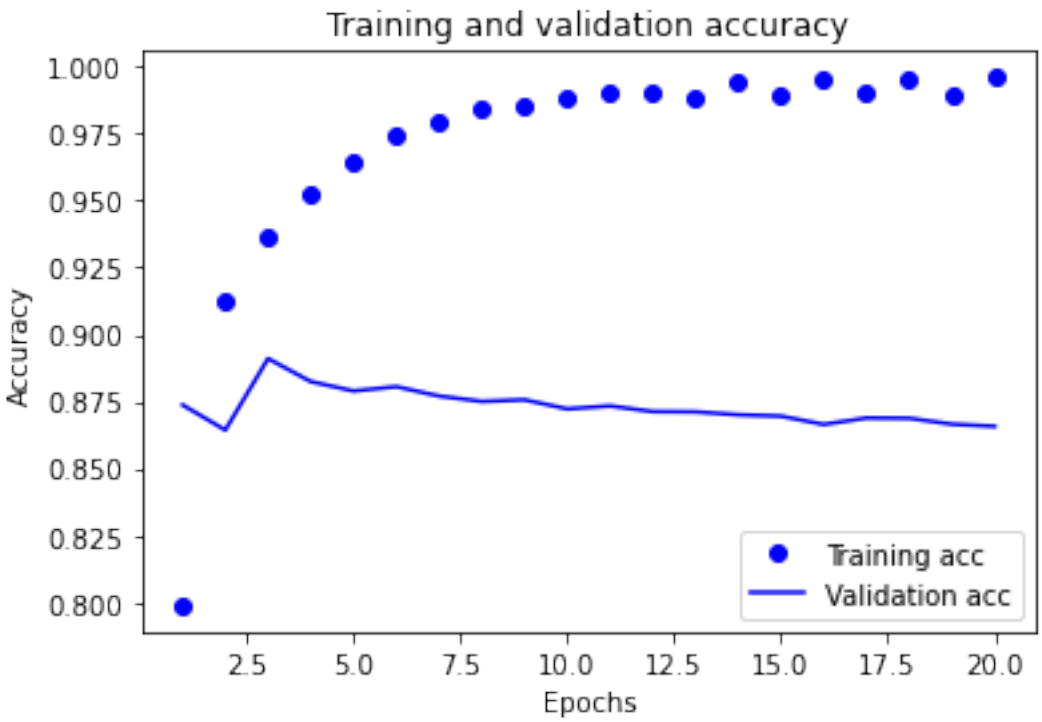
```
Epoch 15/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0091 - accuracy:
0.9892 - val_loss: 0.1136 - val_accuracy: 0.8695
Epoch 16/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0049 - accuracy:
0.9952 - val_loss: 0.1158 - val_accuracy: 0.8664
Epoch 17/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0088 - accuracy:
0.9901 - val_loss: 0.1151 - val_accuracy: 0.8688
Epoch 18/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0043 - accuracy:
0.9957 - val_loss: 0.1167 - val_accuracy: 0.8687
Epoch 19/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0091 - accuracy:
0.9897 - val_loss: 0.1183 - val_accuracy: 0.8665
Epoch 20/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0039 - accuracy:
0.9961 - val_loss: 0.1184 - val_accuracy: 0.8657
```

[182]:
```
history_dict_5 = history_5.history
loss_values = history_dict_5["loss"]
val_loss_values = history_dict_5["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()
```




[183]:
```
# Plotting the training and validation accuracy
```

[184]:
```
plt.clf()
acc = history_dict_5["accuracy"]
val_acc = history_dict_5["val_accuracy"]
plt.plot(epochs, acc, "bo", label="Training acc")
plt.plot(epochs, val_acc, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```




[185]:
```
# 5.Use any technique we studied in class, and these include regularization,␣
↪dropout, etc., to get your model to perform better on validation.
```

[186]:
```
model_6 = keras.Sequential([
    #layers.Dropout(0.2),
    layers.Dense(20, activation="relu"),
    layers.Dropout(0.2),
    layers.Dense(15, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

[187]:
```
# Model completion
```

[188]:
```
model_6.compile(optimizer="rmsprop",
                loss="binary_crossentropy",
                metrics=["accuracy"])
```

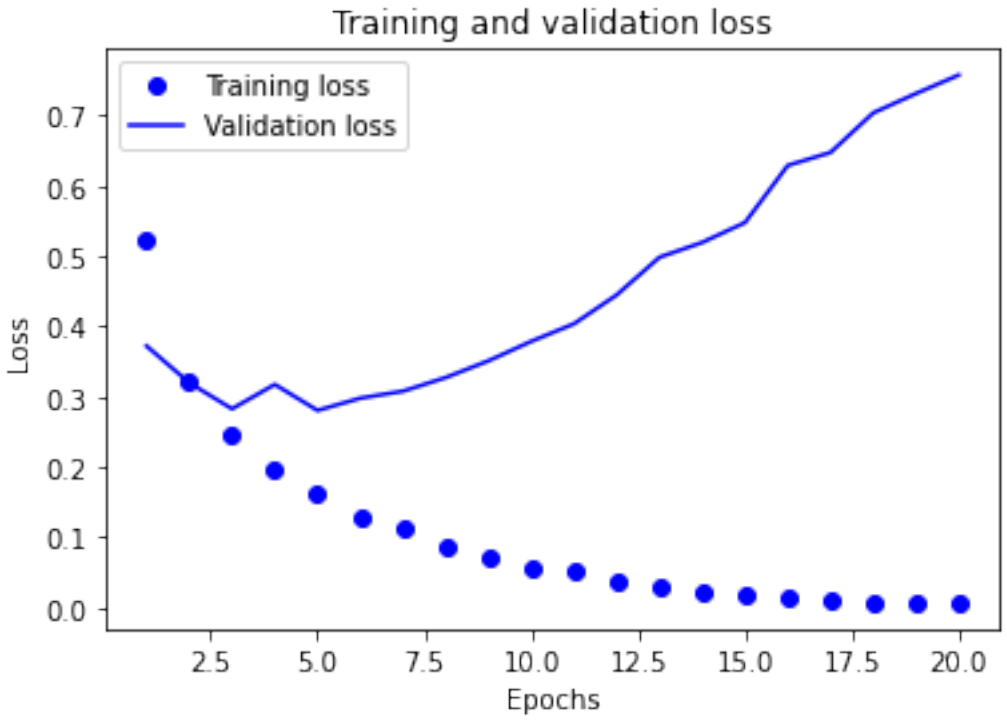
[189]:
```
history_6 = model_6.fit(partial_x_train,
                        partial_y_train,
                        epochs=20,
                        batch_size=512,
                        validation_data=(x_val, y_val))
```

```
Epoch 1/20
```

```
30/30 [==============================] - 2s 34ms/step - loss: 0.5248 - accuracy:
0.7677 - val_loss: 0.3721 - val_accuracy: 0.8752
Epoch 2/20
30/30 [==============================] - 1s 23ms/step - loss: 0.3218 - accuracy:
0.8903 - val_loss: 0.3197 - val_accuracy: 0.8767
Epoch 3/20
30/30 [==============================] - 1s 31ms/step - loss: 0.2458 - accuracy:
0.9128 - val_loss: 0.2827 - val_accuracy: 0.8882
Epoch 4/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1965 - accuracy:
0.9333 - val_loss: 0.3174 - val_accuracy: 0.8735
Epoch 5/20
30/30 [==============================] - 1s 23ms/step - loss: 0.1611 - accuracy:
0.9455 - val_loss: 0.2805 - val_accuracy: 0.8880
Epoch 6/20
30/30 [==============================] - 1s 23ms/step - loss: 0.1294 - accuracy:
0.9569 - val_loss: 0.2979 - val_accuracy: 0.8861
Epoch 7/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1122 - accuracy:
0.9627 - val_loss: 0.3078 - val_accuracy: 0.8854
Epoch 8/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0870 - accuracy:
0.9725 - val_loss: 0.3272 - val_accuracy: 0.8843
Epoch 9/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0724 - accuracy:
0.9793 - val_loss: 0.3511 - val_accuracy: 0.8823
Epoch 10/20
30/30 [==============================] - 1s 23ms/step - loss: 0.0579 - accuracy:
0.9821 - val_loss: 0.3787 - val_accuracy: 0.8785
Epoch 11/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0513 - accuracy:
0.9851 - val_loss: 0.4036 - val_accuracy: 0.8810
Epoch 12/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0380 - accuracy:
0.9903 - val_loss: 0.4448 - val_accuracy: 0.8745
Epoch 13/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0294 - accuracy:
0.9924 - val_loss: 0.4983 - val_accuracy: 0.8741
Epoch 14/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0232 - accuracy:
0.9955 - val_loss: 0.5193 - val_accuracy: 0.8722
Epoch 15/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0201 - accuracy:
0.9955 - val_loss: 0.5475 - val_accuracy: 0.8748
Epoch 16/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0135 - accuracy:
0.9976 - val_loss: 0.6287 - val_accuracy: 0.8613
Epoch 17/20
```

```
30/30 [==============================] - 1s 25ms/step - loss: 0.0099 - accuracy:
0.9987 - val_loss: 0.6472 - val_accuracy: 0.8707
Epoch 18/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0084 - accuracy:
0.9985 - val_loss: 0.7035 - val_accuracy: 0.8725
Epoch 19/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0079 - accuracy:
0.9986 - val_loss: 0.7310 - val_accuracy: 0.8735
Epoch 20/20
30/30 [==============================] - 1s 24ms/step - loss: 0.0072 - accuracy:
0.9985 - val_loss: 0.7573 - val_accuracy: 0.8720
```

[190]:
```
history_dict_6 = history_6.history
loss_values = history_dict_6["loss"]
val_loss_values = history_dict_6["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()
```



[191]:
```
# Plotting the training and validation accuracy
```

[192]:
```
plt.clf()
acc = history_dict_6["accuracy"]
val_acc = history_dict_6["val_accuracy"]
plt.plot(epochs, acc, "bo", label="Training acc")
plt.plot(epochs, val_acc, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```