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

```
WARNING:tensorflow:From c:\Users\Supri\AppData\Local\Programs\Python\Python39\lib\site-packages\keras\src\losses.py:2976: The name tf.lc

Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb.npz
17464789/17464789 [==============================] - 1s 0us/step
```

To retrain the model, I've chosen three epochs in this instance

```
train_data[0]
```

```
[1,
 14,
 22,
 16,
 43,
 530,
 973,
 1622,
 1385,
 65,
 458,
 4468,
 66,
 3941,
 4,
 173,
 36,
 256,
 5,
 25,
 100,
 43,
 838,
 112,
 50,
 670,
 2,
 9,
 35,
 480,
 284,
 5,
 150,
 4,
 172,
 112,
 167,
 2,
 336,
 385,
 39,
 4,
 172,
 4536,
 1111,
 17,
 546,
 38,
 13,
 447,
 4,
 192,
 50,
 16,
 6,
 147,
 2025,
 19,
```

Checking the Data

```
train_labels[0]
```

```
max([max(sequence) for sequence in train_data])
```

```
9999
```

```
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]])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb_word_index.json
1641221/1641221 [==============================] - 0s 0us/step
```

Preparing the Data

Utilizing the multi hot encoding to encode the integer sequences

```
import numpy as np
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
x_train = vectorize_sequences(train_data)
x_test = vectorize_sequences(test_data)
```

```
x_train[0]
```

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

Building the Model

Definition of the Model

```
from tensorflow import keras
from tensorflow.keras import layers
# # In this case, I'm using two hidden layers, each with sixteen nodes, and one output layer node for either a +ve or -ve output. Hidden is
model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

```
WARNING:tensorflow:From c:\Users\Supri\AppData\Local\Programs\Python\Python39\lib\site-packages\keras\src\backend.py:873: The name tf.ge
```

Compiling the model

The Loss function is the binary crossentropy and the Adam serves as the optimizer

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

```
WARNING:tensorflow:From c:\Users\Supri\AppData\Local\Programs\Python\Python39\lib\site-packages\keras\src\optimizers\__init__.py:309: Th
```

Verifying the approach

Considering a validation set aside

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

Training the Model

We are using 512 batches and 20 epochs to train the model

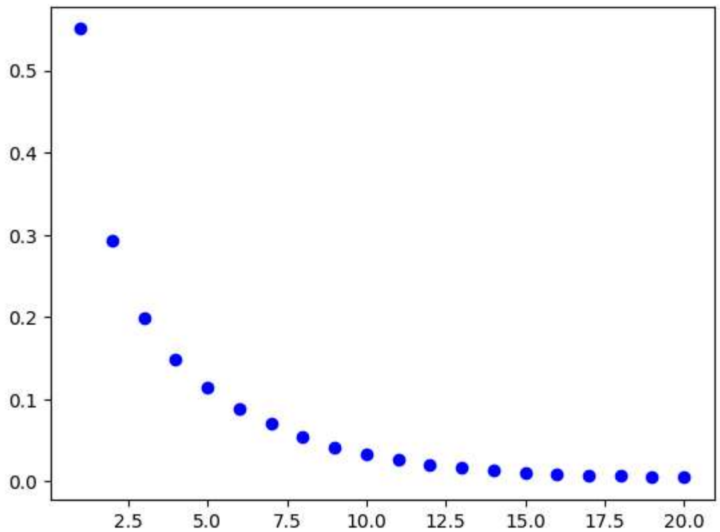
```
history = model.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))

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

```
Epoch 1/20
WARNING:tensorflow:From c:\Users\Supri\AppData\Local\Programs\Python\Python39\lib\site-packages\keras\src\utils\tf_utils.py:492: The nam

WARNING:tensorflow:From c:\Users\Supri\AppData\Local\Programs\Python\Python39\lib\site-packages\keras\src\engine\base_layer_utils.py:384

30/30 [==============================] - 2s 33ms/step - loss: 0.5506 - accuracy: 0.7566 - val_loss: 0.3902 - val_accuracy: 0.8599
Epoch 2/20
30/30 [==============================] - 0s 9ms/step - loss: 0.2939 - accuracy: 0.9015 - val_loss: 0.2930 - val_accuracy: 0.8854
Epoch 3/20
30/30 [==============================] - 0s 9ms/step - loss: 0.1990 - accuracy: 0.9345 - val_loss: 0.2799 - val_accuracy: 0.8904
Epoch 4/20
30/30 [==============================] - 0s 9ms/step - loss: 0.1488 - accuracy: 0.9542 - val_loss: 0.2819 - val_accuracy: 0.8861
Epoch 5/20
30/30 [==============================] - 0s 9ms/step - loss: 0.1143 - accuracy: 0.9675 - val_loss: 0.2965 - val_accuracy: 0.8823
Epoch 6/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0891 - accuracy: 0.9780 - val_loss: 0.3150 - val_accuracy: 0.8807
Epoch 7/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0698 - accuracy: 0.9855 - val_loss: 0.3410 - val_accuracy: 0.8787
Epoch 8/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0541 - accuracy: 0.9904 - val_loss: 0.3675 - val_accuracy: 0.8780
Epoch 9/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0419 - accuracy: 0.9941 - val_loss: 0.3939 - val_accuracy: 0.8777
Epoch 10/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0324 - accuracy: 0.9963 - val_loss: 0.4228 - val_accuracy: 0.8753
Epoch 11/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0260 - accuracy: 0.9976 - val_loss: 0.4547 - val_accuracy: 0.8713
Epoch 12/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0210 - accuracy: 0.9991 - val_loss: 0.4776 - val_accuracy: 0.8723
Epoch 13/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0168 - accuracy: 0.9996 - val_loss: 0.5030 - val_accuracy: 0.8707
Epoch 14/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0133 - accuracy: 0.9997 - val_loss: 0.5276 - val_accuracy: 0.8688
Epoch 15/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0108 - accuracy: 0.9998 - val_loss: 0.5495 - val_accuracy: 0.8681
Epoch 16/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0089 - accuracy: 0.9999 - val_loss: 0.5708 - val_accuracy: 0.8680
Epoch 17/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0076 - accuracy: 0.9999 - val_loss: 0.5916 - val_accuracy: 0.8669
Epoch 18/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0064 - accuracy: 0.9999 - val_loss: 0.6101 - val_accuracy: 0.8676
Epoch 19/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0055 - accuracy: 0.9999 - val_loss: 0.6287 - val_accuracy: 0.8665
Epoch 20/20
30/30 [==============================] - 0s 7ms/step - loss: 0.0048 - accuracy: 0.9999 - val_loss: 0.6454 - val_accuracy: 0.8670
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

Plotting the training and the validation loss

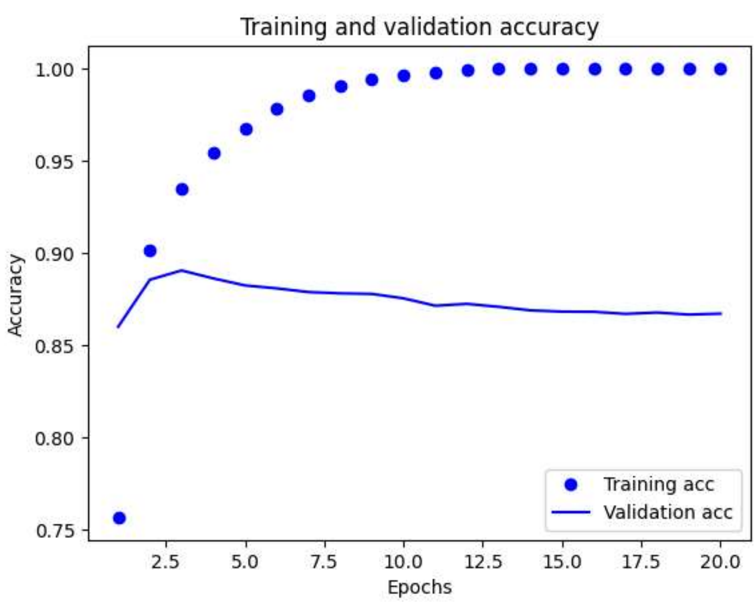
```
import matplotlib.pyplot as plt
history_dict = history.history
loss_values = history_dict["loss"]
val_loss_values = history_dict["val_loss"]
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values, "bo", label="Training loss")
```

```
[<matplotlib.lines.Line2D at 0x295d5eaf4c0>]
```

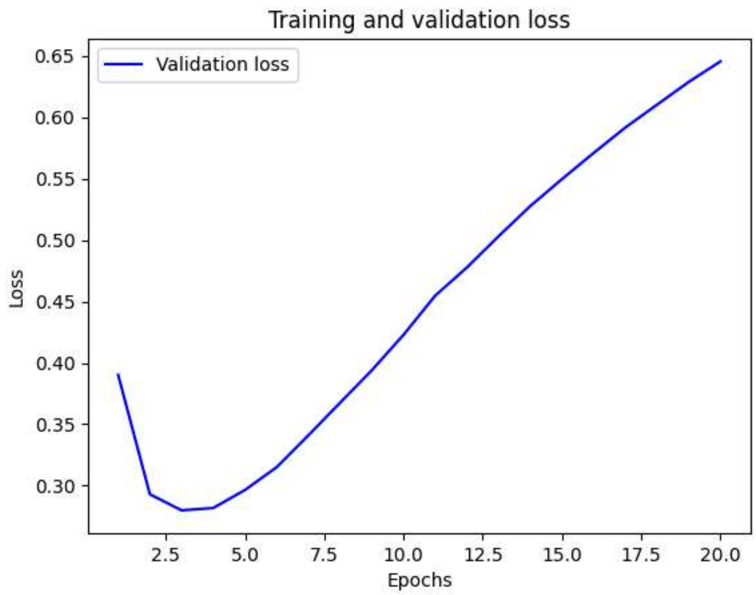


Plotting the training and the validation accuracy

```
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()
```



```
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()
```



Retraining the model from the beginning

```
model = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
#To retrain the model, I've chosen three epochs in this instance.
model.compile(optimizer="adam",
              loss="binary_crossentropy",
              metrics=["accuracy"])
model.fit(x_train, y_train, epochs=4, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/4
49/49 [==============================] - 1s 7ms/step - loss: 0.4462 - accuracy: 0.8172
Epoch 2/4
49/49 [==============================] - 0s 6ms/step - loss: 0.2246 - accuracy: 0.9191
Epoch 3/4
49/49 [==============================] - 0s 5ms/step - loss: 0.1687 - accuracy: 0.9407
Epoch 4/4
49/49 [==============================] - 0s 5ms/step - loss: 0.1342 - accuracy: 0.9541
782/782 [==============================] - 1s 2ms/step - loss: 0.3264 - accuracy: 0.8776
[0.32635897397994995, 0.877560019493103]
```

Constructing the Model

1 utilizing two or three hidden layers and observe how it affects the validation and test accuracy

```
model1_1 = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])
```

```
model1_3 = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])


model1_1.compile(optimizer="adam",
              loss="binary_crossentropy",
              metrics=["accuracy"])

model1_3.compile(optimizer="adam",
              loss="binary_crossentropy",
              metrics=["accuracy"])
```

Model fitting with 20 epochs and 512 batch size

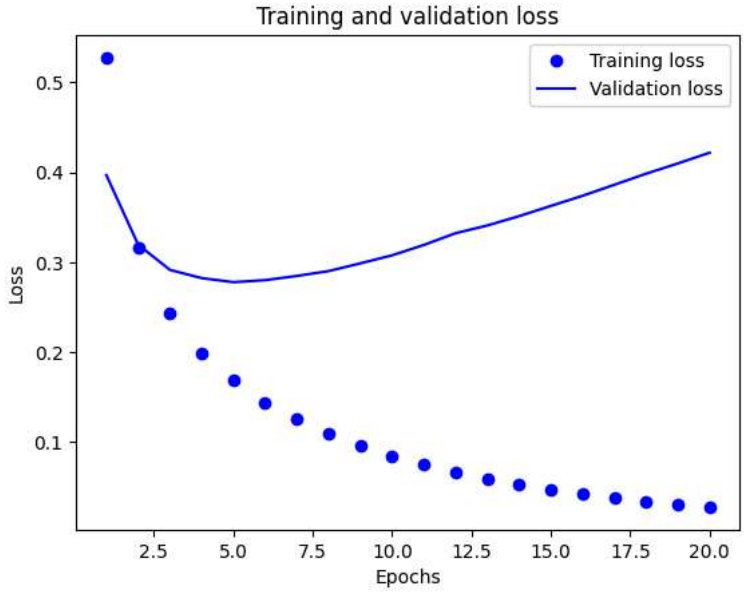
```
history1_1 = model1_1.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))

history1_3 = model1_3.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))
```

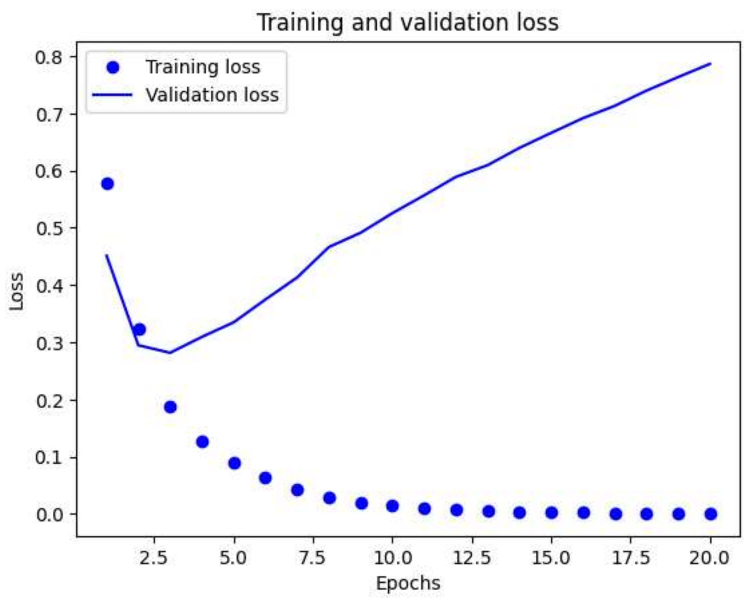
```
Epoch 1/20
30/30 [==============================] - 1s 24ms/step - loss: 0.5275 - accuracy: 0.7851 - val_loss: 0.3968 - val_accuracy: 0.8628
Epoch 2/20
30/30 [==============================] - 0s 9ms/step - loss: 0.3163 - accuracy: 0.8991 - val_loss: 0.3196 - val_accuracy: 0.8842
Epoch 3/20
30/30 [==============================] - 0s 8ms/step - loss: 0.2432 - accuracy: 0.9241 - val_loss: 0.2915 - val_accuracy: 0.8892
Epoch 4/20
30/30 [==============================] - 0s 9ms/step - loss: 0.1993 - accuracy: 0.9396 - val_loss: 0.2823 - val_accuracy: 0.8905
Epoch 5/20
30/30 [==============================] - 0s 9ms/step - loss: 0.1682 - accuracy: 0.9516 - val_loss: 0.2778 - val_accuracy: 0.8891
Epoch 6/20
30/30 [==============================] - 0s 9ms/step - loss: 0.1441 - accuracy: 0.9597 - val_loss: 0.2800 - val_accuracy: 0.8880
Epoch 7/20
30/30 [==============================] - 0s 8ms/step - loss: 0.1250 - accuracy: 0.9672 - val_loss: 0.2847 - val_accuracy: 0.8852
Epoch 8/20
30/30 [==============================] - 0s 9ms/step - loss: 0.1090 - accuracy: 0.9739 - val_loss: 0.2901 - val_accuracy: 0.8843
Epoch 9/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0956 - accuracy: 0.9787 - val_loss: 0.2987 - val_accuracy: 0.8833
Epoch 10/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0844 - accuracy: 0.9822 - val_loss: 0.3076 - val_accuracy: 0.8810
Epoch 11/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0747 - accuracy: 0.9870 - val_loss: 0.3192 - val_accuracy: 0.8801
Epoch 12/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0668 - accuracy: 0.9899 - val_loss: 0.3322 - val_accuracy: 0.8780
Epoch 13/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0590 - accuracy: 0.9904 - val_loss: 0.3408 - val_accuracy: 0.8780
Epoch 14/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0525 - accuracy: 0.9930 - val_loss: 0.3512 - val_accuracy: 0.8777
Epoch 15/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0469 - accuracy: 0.9937 - val_loss: 0.3626 - val_accuracy: 0.8768
Epoch 16/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0419 - accuracy: 0.9956 - val_loss: 0.3739 - val_accuracy: 0.8762
Epoch 17/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0376 - accuracy: 0.9964 - val_loss: 0.3861 - val_accuracy: 0.8753
Epoch 18/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0342 - accuracy: 0.9975 - val_loss: 0.3984 - val_accuracy: 0.8739
Epoch 19/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0306 - accuracy: 0.9978 - val_loss: 0.4098 - val_accuracy: 0.8743
Epoch 20/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0276 - accuracy: 0.9987 - val_loss: 0.4218 - val_accuracy: 0.8723
Epoch 1/20
30/30 [==============================] - 2s 29ms/step - loss: 0.5774 - accuracy: 0.7115 - val_loss: 0.4511 - val_accuracy: 0.8552
Epoch 2/20
30/30 [==============================] - 0s 10ms/step - loss: 0.3231 - accuracy: 0.9061 - val_loss: 0.2947 - val_accuracy: 0.8894
Epoch 3/20
30/30 [==============================] - 0s 11ms/step - loss: 0.1874 - accuracy: 0.9425 - val_loss: 0.2815 - val_accuracy: 0.8882
Epoch 4/20
30/30 [==============================] - 0s 11ms/step - loss: 0.1264 - accuracy: 0.9625 - val_loss: 0.3092 - val_accuracy: 0.8829
Epoch 5/20
```

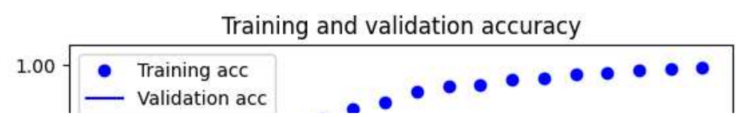
```
30/30 [==============================] - 0s 11ms/step - loss: 0.0887 - accuracy: 0.9771 - val_loss: 0.3346 - val_accuracy: 0.8805
Epoch 6/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0629 - accuracy: 0.9871 - val_loss: 0.3745 - val_accuracy: 0.8776
Epoch 7/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0422 - accuracy: 0.9933 - val_loss: 0.4130 - val_accuracy: 0.8750
Epoch 8/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0288 - accuracy: 0.9971 - val_loss: 0.4662 - val_accuracy: 0.8721
Epoch 9/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0198 - accuracy: 0.9983 - val_loss: 0.4912 - val_accuracy: 0.8709
```

Plotting the training Vs validation data

```
historyp1_1 = history1_1.history
historyp1_1.keys()

historyp1_3 = history1_1.history
historyp1_3.keys()

historyp1_1 = history1_1.history
loss_values1 = historyp1_1["loss"]
val_loss_values1 = historyp1_1["val_loss"]
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values1, "bo", label="Training loss")
plt.plot(epochs, val_loss_values1, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()

historyp1_3 = history1_3.history
loss_values3 = historyp1_3["loss"]
val_loss_values3 = historyp1_3["val_loss"]
epochs = range(1, len(loss_values) + 1)
plt.plot(epochs, loss_values3, "bo", label="Training loss")
plt.plot(epochs, val_loss_values3, "b", label="Validation loss")
plt.title("Training and validation loss")
plt.xlabel("Epochs")
plt.ylabel("Loss")
plt.legend()
plt.show()

plt.clf()
acc1 = historyp1_1["accuracy"]
val_acc1 = historyp1_1["val_accuracy"]
plt.plot(epochs, acc1, "bo", label="Training acc")
plt.plot(epochs, val_acc1, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()

plt.clf()
acc3 = historyp1_3["accuracy"]
val_acc3 = historyp1_3["val_accuracy"]
plt.plot(epochs, acc3, "bo", label="Training acc")
plt.plot(epochs, val_acc3, "b", label="Validation acc")
plt.title("Training and validation accuracy")
plt.xlabel("Epochs")
plt.ylabel("Accuracy")
plt.legend()
plt.show()
```


Training and validation loss
Training loss
Validation loss
Loss
Epochs


Training and validation loss
Training loss
Validation loss
Loss
Epochs


Training and validation accuracy
Training acc
Validation acc

2 for the hidden layers iam using 32 node units and 64 units

```
model2 = keras.Sequential([
    layers.Dense(32, activation="relu"),
    layers.Dense(64, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])

model2.compile(optimizer="adam",
              loss="binary_crossentropy",
              metrics=["accuracy"])

hist2 = model2.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))

histp2 = hist2.history
loss_values = histp2["loss"]
val_loss_values = histp2["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()

plt.clf()
acc = histp2["accuracy"]
val_acc = histp2["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()
```

```
Epoch 1/20
30/30 [==============================] - 2s 32ms/step - loss: 0.5163 - accuracy: 0.78
Epoch 2/20
30/30 [==============================] - 0s 13ms/step - loss: 0.2378 - accuracy: 0.91
Epoch 3/20
30/30 [==============================] - 0s 14ms/step - loss: 0.1462 - accuracy: 0.94
Epoch 4/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0970 - accuracy: 0.97
Epoch 5/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0638 - accuracy: 0.98
Epoch 6/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0426 - accuracy: 0.99
Epoch 7/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0251 - accuracy: 0.99
Epoch 8/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0146 - accuracy: 0.99
Epoch 9/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0091 - accuracy: 0.99
Epoch 10/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0060 - accuracy: 0.99
Epoch 11/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0043 - accuracy: 0.99
Epoch 12/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0032 - accuracy: 0.99
Epoch 13/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0025 - accuracy: 0.99
Epoch 14/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0020 - accuracy: 0.99
Epoch 15/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0016 - accuracy: 0.99
Epoch 16/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0013 - accuracy: 1.00
Epoch 17/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0011 - accuracy: 1.00
Epoch 18/20
30/30 [==============================] - 0s 11ms/step - loss: 9.4539e-04 - accuracy:
Epoch 19/20
30/30 [==============================] - 0s 11ms/step - loss: 8.1724e-04 - accuracy:
Epoch 20/20
30/30 [==============================] - 0s 11ms/step - loss: 7.1635e-04 - accuracy:
```

3 using the MSE loss function instead of the binary_crossentropy

```
model3 = keras.Sequential([
    layers.Dense(16, activation="relu"),
    layers.Dense(16, activation="relu"),
    layers.Dense(1, activation="sigmoid")
])


model3.compile(optimizer="adam",
              loss="mse",
              metrics=["accuracy"])

hist3 = model3.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))

histp3 = hist3.history
loss_values = histp3["loss"]
val_loss_values = histp3["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()

plt.clf()
acc = histp3["accuracy"]
val_acc = histp3["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()
```

```
Epoch 1/20
30/30 [==============================] - 1s 30ms/step - loss: 0.1745 - accuracy: 0.79
Epoch 2/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0836 - accuracy: 0.905
Epoch 3/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0549 - accuracy: 0.937
Epoch 4/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0392 - accuracy: 0.959
Epoch 5/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0298 - accuracy: 0.973
Epoch 6/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0226 - accuracy: 0.981
Epoch 7/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0168 - accuracy: 0.98
Epoch 8/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0132 - accuracy: 0.990
Epoch 9/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0103 - accuracy: 0.99
Epoch 10/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0085 - accuracy: 0.994
Epoch 11/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0070 - accuracy: 0.995
Epoch 12/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0061 - accuracy: 0.99
Epoch 13/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0055 - accuracy: 0.995
Epoch 14/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0050 - accuracy: 0.996
Epoch 15/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0046 - accuracy: 0.996
Epoch 16/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0043 - accuracy: 0.996
Epoch 17/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0041 - accuracy: 0.996
Epoch 18/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0040 - accuracy: 0.996
Epoch 19/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0038 - accuracy: 0.996
Epoch 20/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0037 - accuracy: 0.996
```

Using the tanh activation instead of relu

```
model4 = keras.Sequential([
    layers.Dense(16, activation="tanh"),
    layers.Dense(16, activation="tanh"),
    layers.Dense(1, activation="sigmoid")
])

model4.compile(optimizer="adam",
              loss="mse",
              metrics=["accuracy"])

hist4 = model4.fit(partial_x_train,
                    partial_y_train,
                    epochs=20,
                    batch_size=512,
                    validation_data=(x_val, y_val))

histp4 = hist4.history
loss_values = histp4["loss"]
val_loss_values = histp4["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()

plt.clf()
acc = histp4["accuracy"]
val_acc = histp4["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()
```

```
Epoch 1/20
30/30 [==============================] - 1s 26ms/step - loss: 0.1708 - accuracy: 0.78
Epoch 2/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0797 - accuracy: 0.908
Epoch 3/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0520 - accuracy: 0.942
Epoch 4/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0373 - accuracy: 0.963
Epoch 5/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0274 - accuracy: 0.97
Epoch 6/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0203 - accuracy: 0.983
Epoch 7/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0154 - accuracy: 0.988
Epoch 8/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0121 - accuracy: 0.991
Epoch 9/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0097 - accuracy: 0.992
Epoch 10/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0086 - accuracy: 0.993
Epoch 11/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0074 - accuracy: 0.994
Epoch 12/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0067 - accuracy: 0.994
Epoch 13/20
30/30 [==============================] - 0s 8ms/step - loss: 0.0060 - accuracy: 0.995
Epoch 14/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0055 - accuracy: 0.995
Epoch 15/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0052 - accuracy: 0.995
Epoch 16/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0050 - accuracy: 0.995
Epoch 17/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0047 - accuracy: 0.995
Epoch 18/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0045 - accuracy: 0.996
Epoch 19/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0042 - accuracy: 0.996
Epoch 20/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0040 - accuracy: 0.996
```