Deep Learning IMDB Data

Hidden Layers	Neurons	Loss Function	Optimiser	Activate Function	Epoch(highest accuracy)	Validation Accuracy	Test Accuracy	Validation Accuracy with Dropout	Test Accuracy with Dropout
1	16	MSE	Adam	Tanh	4	0.9474	0.8843		
1	32	MSE	Adam	Tanh	4	0.9547	0.8813	0.9539	0.8804
1	64	MSE	Adam	Tanh	2	0.9292	0.8844		
1	128	MSE	Adam	Tanh	2	0.9315	0.8800		
2	16	binary_crossentropy	rmsprop	relu	4	0.9425	0.8829		
2	16	MSE	Adam	Tanh	4	0.9630	0.8724	0.9588	0.8750
2	32	MSE	Adam	Tanh	2	0.9306	0.8804		
2	64	MSE	Adam	Tanh	2	0.9348	0.8788		
2	128	MSE	Adam	Tanh	3	0.9561	0.8687		
3	16	MSE	Adam	Tanh	3	0.9493	0.8744	0.9502	0.8787
3	32	MSE	Adam	Tanh	2	0.9307	0.879		
3	64	MSE	Adam	Tanh	2	0.9348	0.8766		
3	128	MSE	Adam	Tanh	2	0.9358	0.8736		

About the data:

The IMDB dataset

"IMDB dataset" is a set of 50,000 highly-polarized reviews from the Internet Movie Database. They are split into 25,000 reviews for training and 25,000 reviews for testing, each set consisting in 50% negative and 50% positive reviews.

The IMDB dataset comes packaged with Keras. It has already been preprocessed: the reviews (sequences of words) have been turned into sequences of integers, where each integer stands for a specific word in a dictionary.

Summary

1.we have used 3 hidden layers with 16,32,64,128 Neurons and we came up with optimal Epoch based on highest validation accuracy in the first case untill 4epoch we can see increasing trend then we can see downtrend that means it indicates that it going to be overfitting. In the first case we can see maximum vadialtion accuracy obtained at epoch 4 with test accuracy of 0.8843%

2.there is no much varaiation in changing active function, optimiser and loss funtion in test accuracy and validation accuracy

3.when we apply 1 ,2 or 3 hidden layers we have not seen much variation or particular trend among different applied neurons

4.when we keep increasing layers with neurons we can see that increase in validation accuracy but there is no much upgradtion in the test accuracy result

5.when we apply the regularisation model with dropout technique there is no much added advantage in validation accuracy and test accuracy.

Conclusion-

After considering above all the points with different layers with Neurons and changing functions does not give much change in test accuracy

In [49]:
```
train_data[0]
```

```
 16,
 43,
 530,
 973,
 1622,
 1385,
 65,
 458,
 4468,
 66,
 3941,
 4,
 173,
 36,
 256,
 5,
 25,
 100,
 43,
 838
```

In [50]:
```
train_labels[0]
```

Out[50]: 1

In [51]:
```
max([max(sequence) for sequence in train_data])
```

Out[51]: 9999

In [52]:
```
#Decoding reviews back to text
```

In [53]:
```
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]])
```

In [54]:
```
#Preparinf data set
```

In [55]:
```
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)
```

In [56]:
```
x_train[0]
```

Out[56]:
```
array([0., 1., 1., ..., 0., 0., 0.])
```

In [57]:
```
y_train = np.asarray(train_labels).astype("float32")
y_test = np.asarray(test_labels).astype("float32")
```

In [58]:
```
#Buliding Model
```

In [59]:
```
from tensorflow import keras
from tensorflow.keras import layers

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

In [60]:
```
model.compile(optimizer="adam",
              loss="binary_crossentropy",
              metrics=["accuracy"])
```

In [61]:
```
#Validating your approach
```

In [62]:
```
x_val = x_train[:10000]
partial_x_train = x_train[10000:]
y_val = y_train[:10000]
partial_y_train = y_train[10000:]
```

In [63]:
```
#Training  model
```

In [64]:
```
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 [==============================] - 1s 26ms/step - loss: 0.5302 - accu
racy: 0.7670 - val_loss: 0.3764 - val_accuracy: 0.8630
Epoch 2/20
30/30 [==============================] - 0s 12ms/step - loss: 0.2807 - accu
racy: 0.9041 - val_loss: 0.2902 - val_accuracy: 0.8869
Epoch 3/20
30/30 [==============================] - 0s 13ms/step - loss: 0.1936 - accu
racy: 0.9367 - val_loss: 0.2761 - val_accuracy: 0.8883
Epoch 4/20
30/30 [==============================] - 0s 14ms/step - loss: 0.1472 - accu
racy: 0.9541 - val_loss: 0.2855 - val_accuracy: 0.8861
Epoch 5/20
30/30 [==============================] - 0s 13ms/step - loss: 0.1116 - accu
racy: 0.9685 - val_loss: 0.3009 - val_accuracy: 0.8820
Epoch 6/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0861 - accu
racy: 0.9787 - val_loss: 0.3194 - val_accuracy: 0.8808
Epoch 7/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0641 - accu
racy: 0.9864 - val_loss: 0.3444 - val_accuracy: 0.8790
Epoch 8/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0475 - accu
racy: 0.9924 - val_loss: 0.3767 - val_accuracy: 0.8746
Epoch 9/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0342 - accu
racy: 0.9959 - val_loss: 0.4039 - val_accuracy: 0.8753
Epoch 10/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0252 - accu
racy: 0.9978 - val_loss: 0.4321 - val_accuracy: 0.8743
Epoch 11/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0182 - accu
racy: 0.9994 - val_loss: 0.4592 - val_accuracy: 0.8740
Epoch 12/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0132 - accu
racy: 0.9997 - val_loss: 0.4842 - val_accuracy: 0.8727
Epoch 13/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0099 - accu
racy: 0.9999 - val_loss: 0.5059 - val_accuracy: 0.8728
Epoch 14/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0078 - accu
racy: 0.9999 - val_loss: 0.5258 - val_accuracy: 0.8720
Epoch 15/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0063 - accu
racy: 0.9999 - val_loss: 0.5448 - val_accuracy: 0.8725
Epoch 16/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0051 - accu
racy: 0.9999 - val_loss: 0.5613 - val_accuracy: 0.8708
Epoch 17/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0043 - accu
racy: 0.9999 - val_loss: 0.5777 - val_accuracy: 0.8698
```

```
Epoch 18/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0037 - accu
racy: 0.9999 - val_loss: 0.5922 - val_accuracy: 0.8689
Epoch 19/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0031 - accu
racy: 0.9999 - val_loss: 0.6069 - val_accuracy: 0.8686
Epoch 20/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0027 - accu
racy: 0.9999 - val_loss: 0.6193 - val_accuracy: 0.8684
```

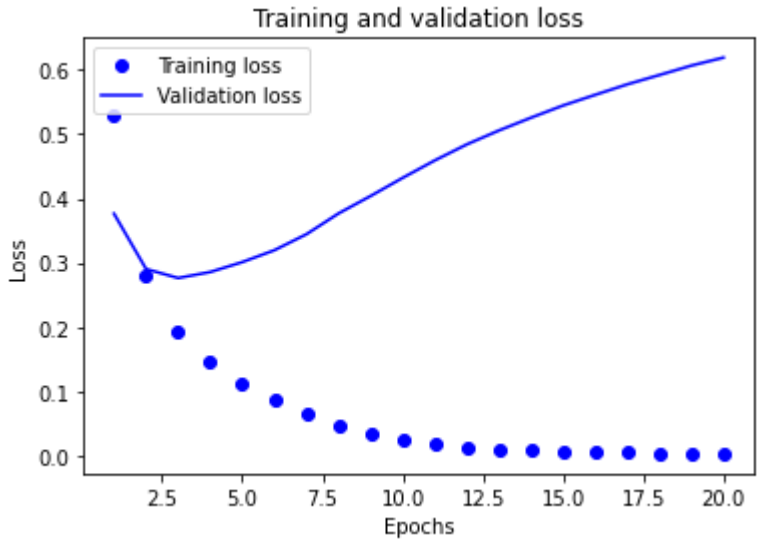
In [65]:
```
#Compiling the model
```

In [66]:
```
history_dict = history.history
history_dict.keys()
```

Out[66]:
```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

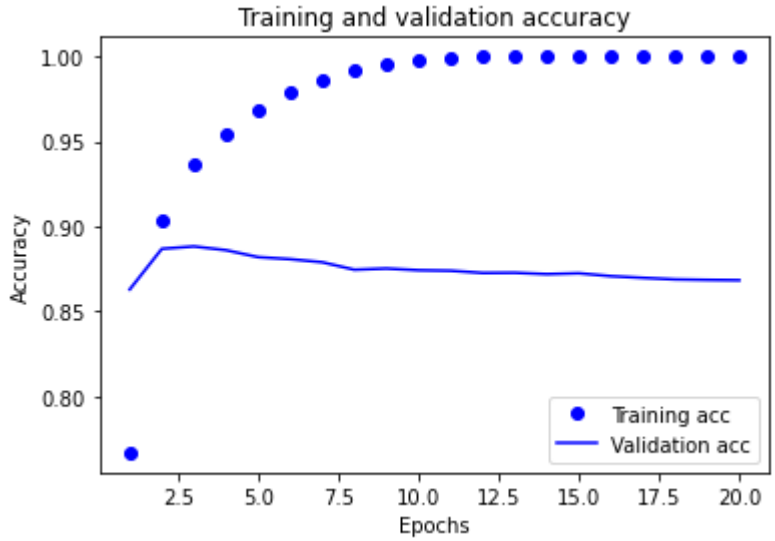
In [67]:
```
#Plotting the training and validation loss
```

In [68]:
```
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")
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()
```



In [69]:
```
#Plotting the training and accuracy
```

In [70]:
```
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()
```



In [71]:
```
#Retraining a model from scratch
```

In [72]:
```
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 [==============================] - 1s 8ms/step - loss: 0.4436 - accur
acy: 0.8260
Epoch 2/4
49/49 [==============================] - 0s 8ms/step - loss: 0.2558 - accur
acy: 0.9102
Epoch 3/4
49/49 [==============================] - 0s 9ms/step - loss: 0.1967 - accur
acy: 0.9304
Epoch 4/4
49/49 [==============================] - 0s 9ms/step - loss: 0.1640 - accur
acy: 0.9425
782/782 [==============================] - 1s 2ms/step - loss: 0.2977 - acc
uracy: 0.8830
```

In [73]:
```
results
```

Out[73]:
```
[0.2977439761161804, 0.8829600214958191]
```

In [74]:
```
#Using a trained model to generate predictions on new data
```

In [75]:
```
model.predict(x_test)
```

```
782/782 [==============================] - 1s 2ms/step
```

Out[75]:
```
array([[0.21102272],
       [0.9997975 ],
       [0.8735719 ],
       ...,
       [0.11094584],
       [0.05689529],
       [0.5809148 ]], dtype=float32)
```

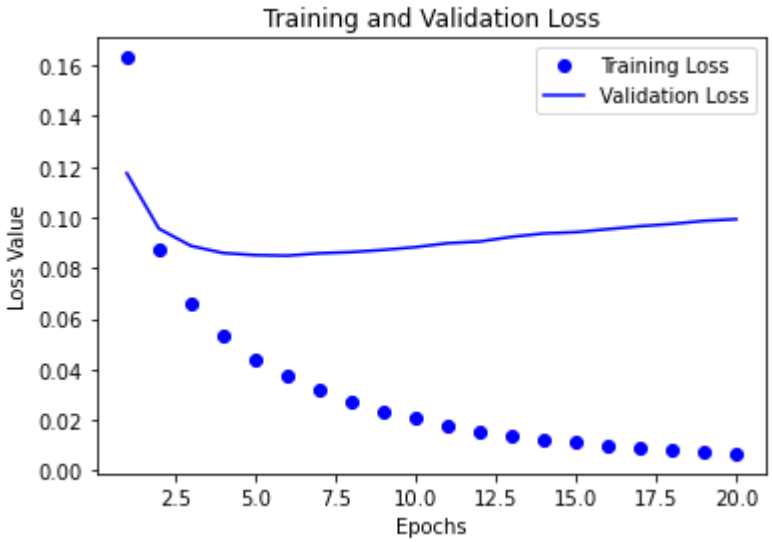
In [116]:

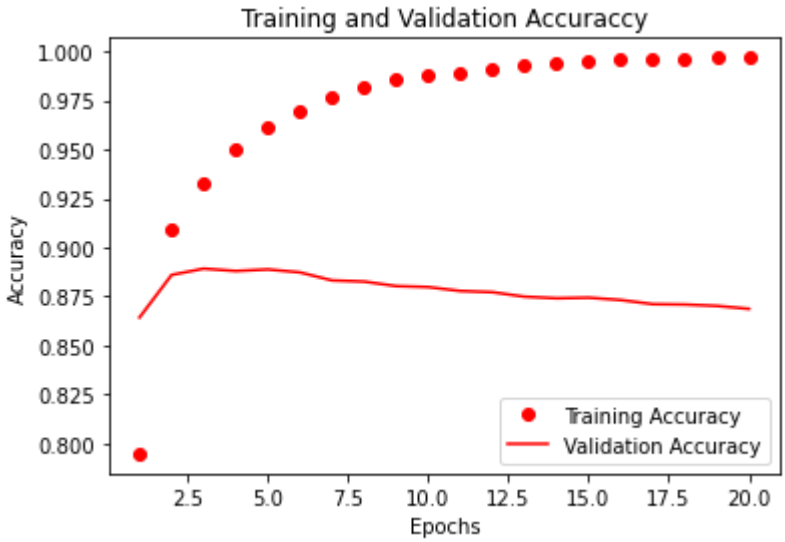
```
#implemented one hideden layer with 16 neurons and mse loss function

from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])



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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()

# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 20ms/step - loss: 0.1630 - bina
ry_accuracy: 0.7949 - val_loss: 0.1174 - val_binary_accuracy: 0.8643
Epoch 2/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0876 - bina
ry_accuracy: 0.9090 - val_loss: 0.0956 - val_binary_accuracy: 0.8860
Epoch 3/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0659 - bina
ry_accuracy: 0.9329 - val_loss: 0.0888 - val_binary_accuracy: 0.8892
Epoch 4/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0531 - bina
ry_accuracy: 0.9501 - val_loss: 0.0859 - val_binary_accuracy: 0.8880
Epoch 5/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0441 - bina
ry_accuracy: 0.9609 - val_loss: 0.0851 - val_binary_accuracy: 0.8888
Epoch 6/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0372 - bina
ry_accuracy: 0.9691 - val_loss: 0.0849 - val_binary_accuracy: 0.8873
Epoch 7/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0317 - bina
ry_accuracy: 0.9770 - val_loss: 0.0858 - val_binary_accuracy: 0.8832
Epoch 8/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0273 - bina
ry_accuracy: 0.9815 - val_loss: 0.0863 - val_binary_accuracy: 0.8826
Epoch 9/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0235 - bina
ry_accuracy: 0.9854 - val_loss: 0.0872 - val_binary_accuracy: 0.8803
Epoch 10/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0206 - bina
```

```
ry_accuracy: 0.9877 - val_loss: 0.0882 - val_binary_accuracy: 0.8798
Epoch 11/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0180 - bina
ry_accuracy: 0.9887 - val_loss: 0.0898 - val_binary_accuracy: 0.8778
Epoch 12/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0158 - bina
ry_accuracy: 0.9913 - val_loss: 0.0905 - val_binary_accuracy: 0.8772
Epoch 13/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0138 - bina
ry_accuracy: 0.9931 - val_loss: 0.0923 - val_binary_accuracy: 0.8749
Epoch 14/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0124 - bina
ry_accuracy: 0.9943 - val_loss: 0.0936 - val_binary_accuracy: 0.8741
Epoch 15/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0111 - bina
ry_accuracy: 0.9951 - val_loss: 0.0942 - val_binary_accuracy: 0.8744
Epoch 16/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0099 - bina
ry_accuracy: 0.9957 - val_loss: 0.0953 - val_binary_accuracy: 0.8732
Epoch 17/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0089 - bina
ry_accuracy: 0.9960 - val_loss: 0.0965 - val_binary_accuracy: 0.8711
Epoch 18/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0081 - bina
ry_accuracy: 0.9965 - val_loss: 0.0974 - val_binary_accuracy: 0.8709
Epoch 19/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0073 - bina
ry_accuracy: 0.9967 - val_loss: 0.0986 - val_binary_accuracy: 0.8702
Epoch 20/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0068 - bina
ry_accuracy: 0.9968 - val_loss: 0.0993 - val_binary_accuracy: 0.8687
```




Training and Validation Accuraccy
Accuracy
Epochs
1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Training Accuracy
Validation Accuracy

In [117]:
```
model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              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 8ms/step - loss: 0.1444 - accur
acy: 0.8234
Epoch 2/4
49/49 [==============================] - 0s 7ms/step - loss: 0.0786 - accur
acy: 0.9121
Epoch 3/4
49/49 [==============================] - 0s 7ms/step - loss: 0.0608 - accur
acy: 0.9342
Epoch 4/4
49/49 [==============================] - 0s 8ms/step - loss: 0.0503 - accur
acy: 0.9474
782/782 [==============================] - 2s 2ms/step - loss: 0.0872 - acc
uracy: 0.8843
```

Out[117]:
```
[0.08715371787548065, 0.8843200206756592]
```

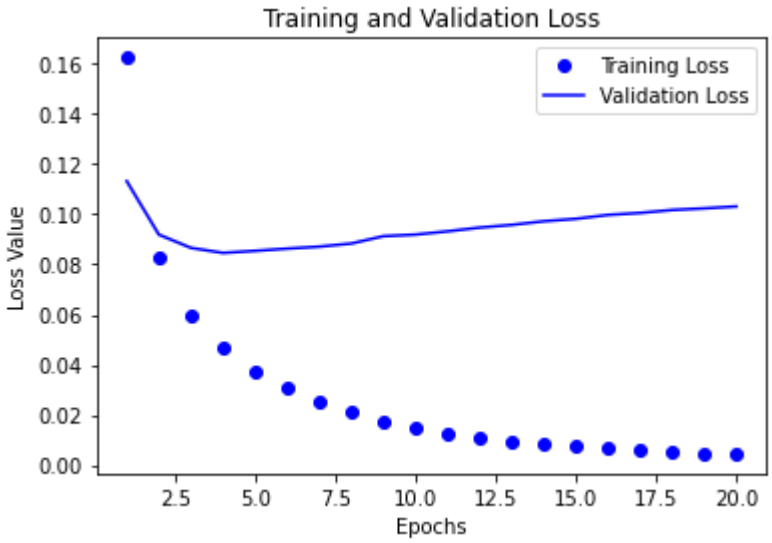
In [119]:

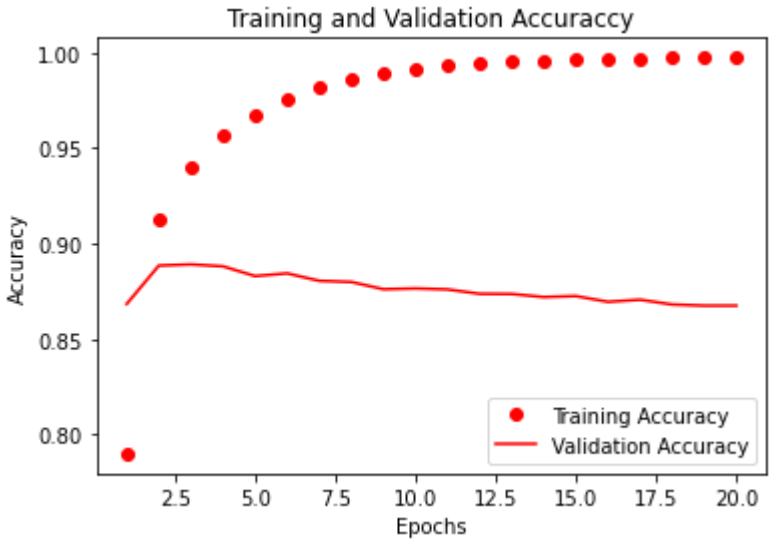
```
#implemented one hideden layer with 32 neurons and mse loss function

from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
```

```
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 24ms/step - loss: 0.1625 - bi
nary_accuracy: 0.7899 - val_loss: 0.1132 - val_binary_accuracy: 0.8684
Epoch 2/20
30/30 [==============================] - 0s 16ms/step - loss: 0.0829 - bi
nary_accuracy: 0.9126 - val_loss: 0.0919 - val_binary_accuracy: 0.8885
Epoch 3/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0597 - bi
nary_accuracy: 0.9398 - val_loss: 0.0866 - val_binary_accuracy: 0.8891
Epoch 4/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0465 - bi
nary_accuracy: 0.9563 - val_loss: 0.0846 - val_binary_accuracy: 0.8881
Epoch 5/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0373 - bi
nary_accuracy: 0.9678 - val_loss: 0.0854 - val_binary_accuracy: 0.8830
Epoch 6/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0305 - bi
nary_accuracy: 0.9759 - val_loss: 0.0863 - val_binary_accuracy: 0.8844
Epoch 7/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0251 - bi
nary_accuracy: 0.9821 - val_loss: 0.0871 - val_binary_accuracy: 0.8805
Epoch 8/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0210 - bi
nary_accuracy: 0.9862 - val_loss: 0.0884 - val_binary_accuracy: 0.8799
Epoch 9/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0175 - bi
nary_accuracy: 0.9897 - val_loss: 0.0913 - val_binary_accuracy: 0.8761
Epoch 10/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0148 - bi
nary_accuracy: 0.9915 - val_loss: 0.0919 - val_binary_accuracy: 0.8765
Epoch 11/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0126 - bi
nary_accuracy: 0.9935 - val_loss: 0.0932 - val_binary_accuracy: 0.8760
Epoch 12/20
```

```
30/30 [==============================] - 0s 14ms/step - loss: 0.0109 - bi
nary_accuracy: 0.9945 - val_loss: 0.0947 - val_binary_accuracy: 0.8737
Epoch 13/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0095 - bi
nary_accuracy: 0.9953 - val_loss: 0.0958 - val_binary_accuracy: 0.8736
Epoch 14/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0084 - bi
nary_accuracy: 0.9957 - val_loss: 0.0972 - val_binary_accuracy: 0.8720
Epoch 15/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0074 - bi
nary_accuracy: 0.9962 - val_loss: 0.0982 - val_binary_accuracy: 0.8725
Epoch 16/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0066 - bi
nary_accuracy: 0.9965 - val_loss: 0.0998 - val_binary_accuracy: 0.8695
Epoch 17/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0060 - bi
nary_accuracy: 0.9968 - val_loss: 0.1005 - val_binary_accuracy: 0.8706
Epoch 18/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0053 - bi
nary_accuracy: 0.9972 - val_loss: 0.1017 - val_binary_accuracy: 0.8681
Epoch 19/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0049 - bi
nary_accuracy: 0.9974 - val_loss: 0.1024 - val_binary_accuracy: 0.8675
Epoch 20/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0046 - bi
nary_accuracy: 0.9975 - val_loss: 0.1031 - val_binary_accuracy: 0.8675
```





In [120]:
```
model = models.Sequential()
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              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 [==============================] - 2s 10ms/step - loss: 0.1349 - accu
racy: 0.8322
Epoch 2/4
49/49 [==============================] - 0s 9ms/step - loss: 0.0698 - accur
acy: 0.9197
Epoch 3/4
49/49 [==============================] - 0s 9ms/step - loss: 0.0536 - accur
acy: 0.9404
Epoch 4/4
49/49 [==============================] - 0s 9ms/step - loss: 0.0432 - accur
acy: 0.9547
782/782 [==============================] - 2s 2ms/step - loss: 0.0872 - acc
uracy: 0.8813
```

Out[120]: [0.08720527589321136, 0.8813199996948242]

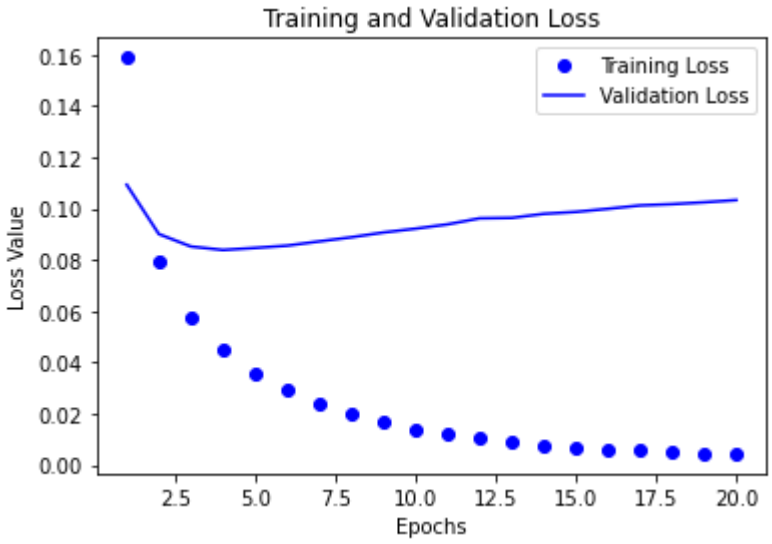
In [144]:

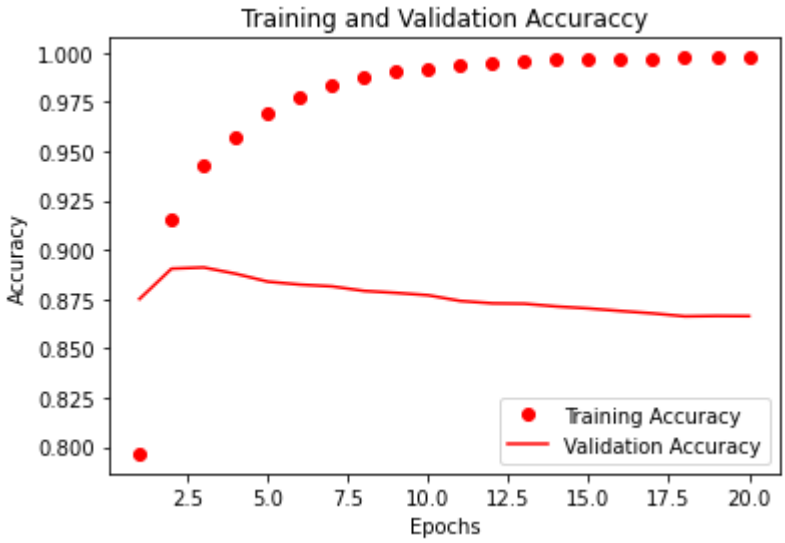
```
#implemented one hideden layer with 32 neurons and mse loss function with dro

from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
```

```
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 21ms/step - loss: 0.1589 - bina
ry_accuracy: 0.7965 - val_loss: 0.1093 - val_binary_accuracy: 0.8750
Epoch 2/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0793 - bina
ry_accuracy: 0.9157 - val_loss: 0.0901 - val_binary_accuracy: 0.8904
Epoch 3/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0573 - bina
ry_accuracy: 0.9427 - val_loss: 0.0852 - val_binary_accuracy: 0.8910
Epoch 4/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0447 - bina
ry_accuracy: 0.9573 - val_loss: 0.0839 - val_binary_accuracy: 0.8878
Epoch 5/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0358 - bina
ry_accuracy: 0.9695 - val_loss: 0.0847 - val_binary_accuracy: 0.8838
Epoch 6/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0292 - bina
ry_accuracy: 0.9778 - val_loss: 0.0856 - val_binary_accuracy: 0.8823
Epoch 7/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0240 - bina
ry_accuracy: 0.9835 - val_loss: 0.0872 - val_binary_accuracy: 0.8814
Epoch 8/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0197 - bina
ry_accuracy: 0.9877 - val_loss: 0.0888 - val_binary_accuracy: 0.8791
Epoch 9/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0165 - bina
ry_accuracy: 0.9904 - val_loss: 0.0907 - val_binary_accuracy: 0.8781
Epoch 10/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0140 - bina
ry_accuracy: 0.9921 - val_loss: 0.0922 - val_binary_accuracy: 0.8769
Epoch 11/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0118 - bina
ry_accuracy: 0.9938 - val_loss: 0.0939 - val_binary_accuracy: 0.8740
```

```
Epoch 12/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0102 - bina
ry_accuracy: 0.9951 - val_loss: 0.0962 - val_binary_accuracy: 0.8728
Epoch 13/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0088 - bina
ry_accuracy: 0.9957 - val_loss: 0.0964 - val_binary_accuracy: 0.8726
Epoch 14/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0077 - bina
ry_accuracy: 0.9964 - val_loss: 0.0980 - val_binary_accuracy: 0.8712
Epoch 15/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0069 - bina
ry_accuracy: 0.9967 - val_loss: 0.0987 - val_binary_accuracy: 0.8702
Epoch 16/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0062 - binar
y_accuracy: 0.9968 - val_loss: 0.0999 - val_binary_accuracy: 0.8689
Epoch 17/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0056 - binar
y_accuracy: 0.9971 - val_loss: 0.1012 - val_binary_accuracy: 0.8677
Epoch 18/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0051 - binar
y_accuracy: 0.9973 - val_loss: 0.1017 - val_binary_accuracy: 0.8662
Epoch 19/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0047 - bina
ry_accuracy: 0.9973 - val_loss: 0.1025 - val_binary_accuracy: 0.8664
Epoch 20/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0044 - bina
ry_accuracy: 0.9975 - val_loss: 0.1033 - val_binary_accuracy: 0.8663
```





In [145]:
```
model = models.Sequential()
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
layers.Dropout(0.5),
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              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: 0.1398 - accur
acy: 0.8262
Epoch 2/4
49/49 [==============================] - 0s 7ms/step - loss: 0.0716 - accur
acy: 0.9189
Epoch 3/4
49/49 [==============================] - 0s 7ms/step - loss: 0.0539 - accur
acy: 0.9402
Epoch 4/4
49/49 [==============================] - 0s 7ms/step - loss: 0.0433 - accur
acy: 0.9539
782/782 [==============================] - 1s 865us/step - loss: 0.0877 - a
ccuracy: 0.8805
```

Out[145]: [0.08766432851552963, 0.8804799914360046]

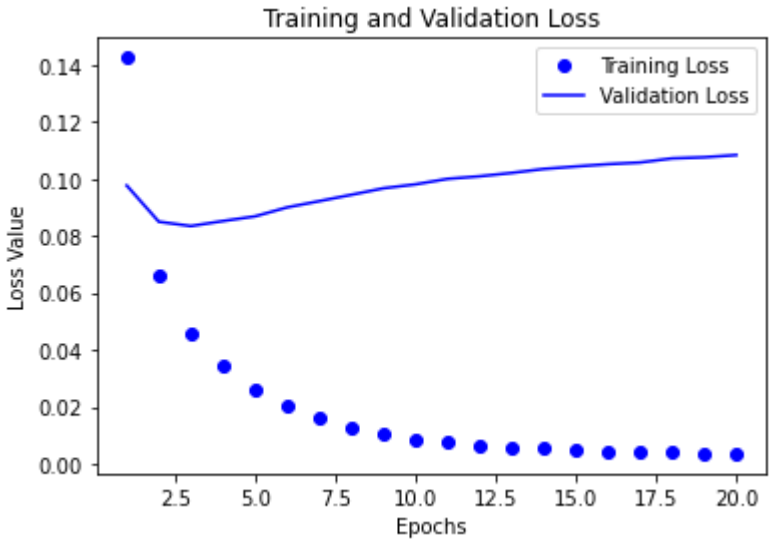
In [123]:

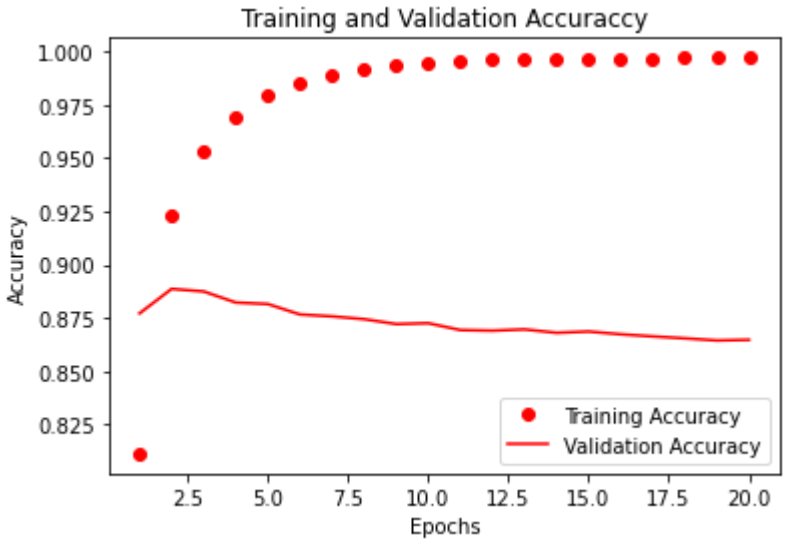
```
#implemented one hideden layer with 64 neurons and mse loss function

from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
```

```
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 29ms/step - loss: 0.1425 - bina
ry_accuracy: 0.8114 - val_loss: 0.0976 - val_binary_accuracy: 0.8772
Epoch 2/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0663 - bina
ry_accuracy: 0.9229 - val_loss: 0.0849 - val_binary_accuracy: 0.8887
Epoch 3/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0459 - bina
ry_accuracy: 0.9529 - val_loss: 0.0834 - val_binary_accuracy: 0.8875
Epoch 4/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0341 - bina
ry_accuracy: 0.9694 - val_loss: 0.0852 - val_binary_accuracy: 0.8823
Epoch 5/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0261 - bina
ry_accuracy: 0.9792 - val_loss: 0.0868 - val_binary_accuracy: 0.8816
Epoch 6/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0204 - bina
ry_accuracy: 0.9851 - val_loss: 0.0900 - val_binary_accuracy: 0.8767
Epoch 7/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0160 - bina
ry_accuracy: 0.9894 - val_loss: 0.0921 - val_binary_accuracy: 0.8758
Epoch 8/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0129 - bina
ry_accuracy: 0.9920 - val_loss: 0.0944 - val_binary_accuracy: 0.8745
Epoch 9/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0107 - bina
ry_accuracy: 0.9941 - val_loss: 0.0967 - val_binary_accuracy: 0.8722
Epoch 10/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0087 - bina
ry_accuracy: 0.9951 - val_loss: 0.0980 - val_binary_accuracy: 0.8726
Epoch 11/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0074 - bina
```

```
ry_accuracy: 0.9959 - val_loss: 0.1000 - val_binary_accuracy: 0.8694
Epoch 12/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0064 - bina
ry_accuracy: 0.9961 - val_loss: 0.1009 - val_binary_accuracy: 0.8691
Epoch 13/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0057 - bina
ry_accuracy: 0.9964 - val_loss: 0.1021 - val_binary_accuracy: 0.8697
Epoch 14/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0052 - bina
ry_accuracy: 0.9965 - val_loss: 0.1034 - val_binary_accuracy: 0.8681
Epoch 15/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0048 - bina
ry_accuracy: 0.9966 - val_loss: 0.1043 - val_binary_accuracy: 0.8687
Epoch 16/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0044 - bina
ry_accuracy: 0.9967 - val_loss: 0.1051 - val_binary_accuracy: 0.8674
Epoch 17/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0042 - bina
ry_accuracy: 0.9969 - val_loss: 0.1057 - val_binary_accuracy: 0.8664
Epoch 18/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0039 - bina
ry_accuracy: 0.9970 - val_loss: 0.1071 - val_binary_accuracy: 0.8655
Epoch 19/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0038 - bina
ry_accuracy: 0.9971 - val_loss: 0.1075 - val_binary_accuracy: 0.8645
Epoch 20/20
30/30 [==============================] - 1s 18ms/step - loss: 0.0035 - bina
ry_accuracy: 0.9972 - val_loss: 0.1083 - val_binary_accuracy: 0.8648
```





In [124]:

```
model = models.Sequential()
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=2, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/2
49/49 [==============================] - 1s 14ms/step - loss: 0.1195 - accu
racy: 0.8475
Epoch 2/2
49/49 [==============================] - 1s 15ms/step - loss: 0.0598 - accu
racy: 0.9292
782/782 [==============================] - 2s 2ms/step - loss: 0.0863 - acc
uracy: 0.8844
```

Out[124]:

```
[0.08631864935159683, 0.8844000101089478]
```

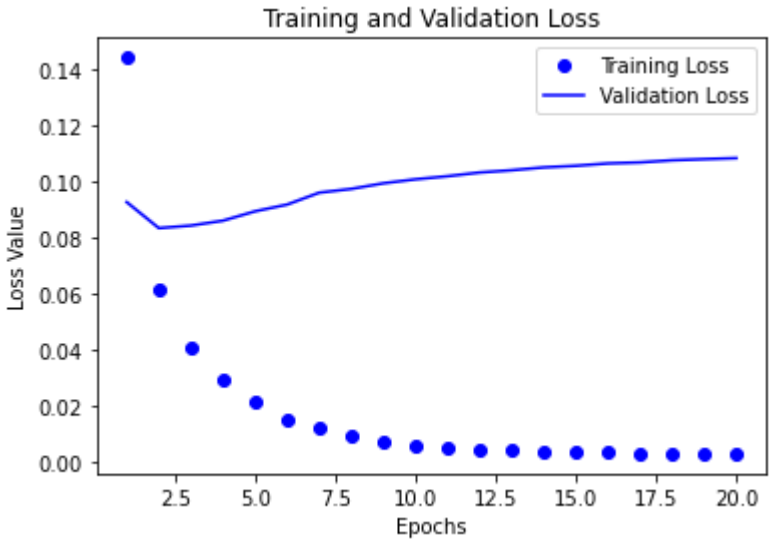
In [125]:

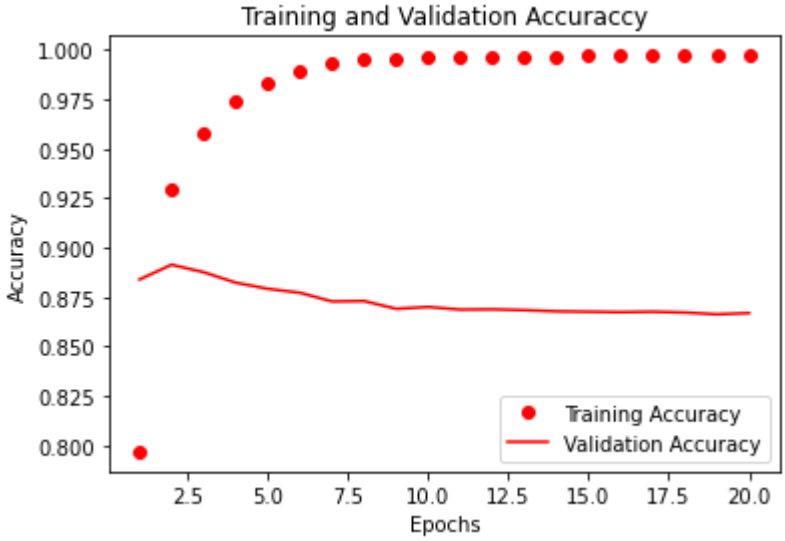
```
#implemented one hideden layer with 128 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 2s 35ms/step - loss: 0.1442 - bi
nary_accuracy: 0.7968 - val_loss: 0.0928 - val_binary_accuracy: 0.8839
Epoch 2/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0614 - bi
nary_accuracy: 0.9293 - val_loss: 0.0835 - val_binary_accuracy: 0.8914
Epoch 3/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0412 - bi
nary_accuracy: 0.9575 - val_loss: 0.0844 - val_binary_accuracy: 0.8876
Epoch 4/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0293 - bi
nary_accuracy: 0.9744 - val_loss: 0.0862 - val_binary_accuracy: 0.8823
Epoch 5/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0215 - bi
nary_accuracy: 0.9832 - val_loss: 0.0895 - val_binary_accuracy: 0.8792
Epoch 6/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0156 - bi
nary_accuracy: 0.9893 - val_loss: 0.0919 - val_binary_accuracy: 0.8772
Epoch 7/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0122 - bi
nary_accuracy: 0.9932 - val_loss: 0.0961 - val_binary_accuracy: 0.8728
Epoch 8/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0093 - bi
nary_accuracy: 0.9950 - val_loss: 0.0975 - val_binary_accuracy: 0.8730
Epoch 9/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0074 - bi
nary_accuracy: 0.9959 - val_loss: 0.0995 - val_binary_accuracy: 0.8691
Epoch 10/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0061 - bi
nary_accuracy: 0.9963 - val_loss: 0.1009 - val_binary_accuracy: 0.8700
Epoch 11/20
```

```
30/30 [==============================] - 1s 26ms/step - loss: 0.0053 - bi
nary_accuracy: 0.9966 - val_loss: 0.1020 - val_binary_accuracy: 0.8687
Epoch 12/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0047 - bi
nary_accuracy: 0.9968 - val_loss: 0.1033 - val_binary_accuracy: 0.8688
Epoch 13/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0044 - bi
nary_accuracy: 0.9969 - val_loss: 0.1041 - val_binary_accuracy: 0.8684
Epoch 14/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0041 - bi
nary_accuracy: 0.9969 - val_loss: 0.1051 - val_binary_accuracy: 0.8678
Epoch 15/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0038 - bi
nary_accuracy: 0.9971 - val_loss: 0.1057 - val_binary_accuracy: 0.8676
Epoch 16/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0036 - bi
nary_accuracy: 0.9971 - val_loss: 0.1066 - val_binary_accuracy: 0.8674
Epoch 17/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0035 - bi
nary_accuracy: 0.9971 - val_loss: 0.1069 - val_binary_accuracy: 0.8676
Epoch 18/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0034 - bi
nary_accuracy: 0.9972 - val_loss: 0.1077 - val_binary_accuracy: 0.8672
Epoch 19/20
30/30 [==============================] - 1s 25ms/step - loss: 0.0033 - bi
nary_accuracy: 0.9972 - val_loss: 0.1081 - val_binary_accuracy: 0.8663
Epoch 20/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0032 - bi
nary_accuracy: 0.9972 - val_loss: 0.1084 - val_binary_accuracy: 0.8669
```




Training and Validation Accuraccy
Accuracy
1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs
Training Accuracy
Validation Accuracy

In [126]:
```
model = models.Sequential()
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=2, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/2
49/49 [==============================] - 1s 19ms/step - loss: 0.1196 - accu
racy: 0.8381
Epoch 2/2
49/49 [==============================] - 1s 19ms/step - loss: 0.0567 - accu
racy: 0.9315
782/782 [==============================] - 2s 2ms/step - loss: 0.0883 - acc
uracy: 0.8800
```

Out[126]:
```
[0.08834024518728256, 0.8800399899482727]
```

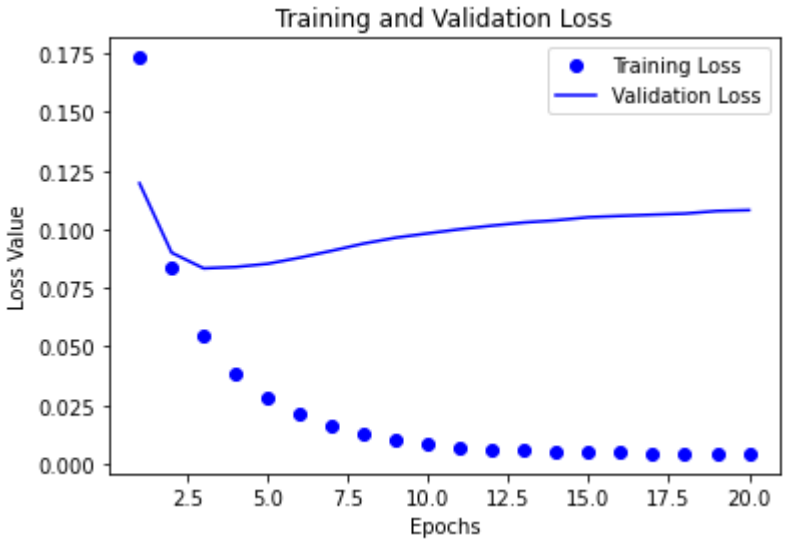
In [127]:

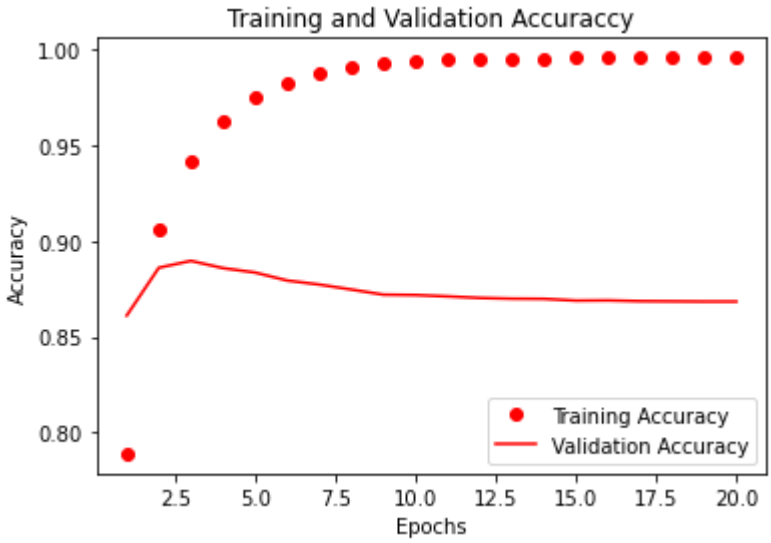
```
#implemented two hideden layer with 16 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 20ms/step - loss: 0.1732 - bi
nary_accuracy: 0.7888 - val_loss: 0.1196 - val_binary_accuracy: 0.8611
Epoch 2/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0838 - bi
nary_accuracy: 0.9064 - val_loss: 0.0900 - val_binary_accuracy: 0.8861
Epoch 3/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0545 - bi
nary_accuracy: 0.9414 - val_loss: 0.0833 - val_binary_accuracy: 0.8897
Epoch 4/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0386 - bi
nary_accuracy: 0.9625 - val_loss: 0.0839 - val_binary_accuracy: 0.8859
Epoch 5/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0285 - bi
nary_accuracy: 0.9759 - val_loss: 0.0853 - val_binary_accuracy: 0.8836
Epoch 6/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0212 - bi
nary_accuracy: 0.9830 - val_loss: 0.0879 - val_binary_accuracy: 0.8794
Epoch 7/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0165 - bi
nary_accuracy: 0.9877 - val_loss: 0.0909 - val_binary_accuracy: 0.8773
Epoch 8/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0127 - bi
nary_accuracy: 0.9909 - val_loss: 0.0939 - val_binary_accuracy: 0.8747
Epoch 9/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0101 - bi
nary_accuracy: 0.9928 - val_loss: 0.0964 - val_binary_accuracy: 0.8720
Epoch 10/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0083 - bi
nary_accuracy: 0.9938 - val_loss: 0.0983 - val_binary_accuracy: 0.8718
Epoch 11/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0070 - bi
```

```
nary_accuracy: 0.9948 - val_loss: 0.1000 - val_binary_accuracy: 0.8711
Epoch 12/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0063 - bi
nary_accuracy: 0.9953 - val_loss: 0.1016 - val_binary_accuracy: 0.8703
Epoch 13/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0056 - bi
nary_accuracy: 0.9957 - val_loss: 0.1029 - val_binary_accuracy: 0.8699
Epoch 14/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0053 - bi
nary_accuracy: 0.9957 - val_loss: 0.1038 - val_binary_accuracy: 0.8698
Epoch 15/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0050 - bi
nary_accuracy: 0.9958 - val_loss: 0.1051 - val_binary_accuracy: 0.8689
Epoch 16/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0047 - bi
nary_accuracy: 0.9959 - val_loss: 0.1057 - val_binary_accuracy: 0.8690
Epoch 17/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0046 - bi
nary_accuracy: 0.9959 - val_loss: 0.1062 - val_binary_accuracy: 0.8686
Epoch 18/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0045 - bi
nary_accuracy: 0.9960 - val_loss: 0.1067 - val_binary_accuracy: 0.8685
Epoch 19/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0044 - bi
nary_accuracy: 0.9961 - val_loss: 0.1078 - val_binary_accuracy: 0.8684
Epoch 20/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0043 - bi
nary_accuracy: 0.9961 - val_loss: 0.1082 - val_binary_accuracy: 0.8684
```




Training and Validation Accuraccy
1.00
0.95
0.90
0.85
0.80
Accuracy
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs
Training Accuracy
Validation Accuracy

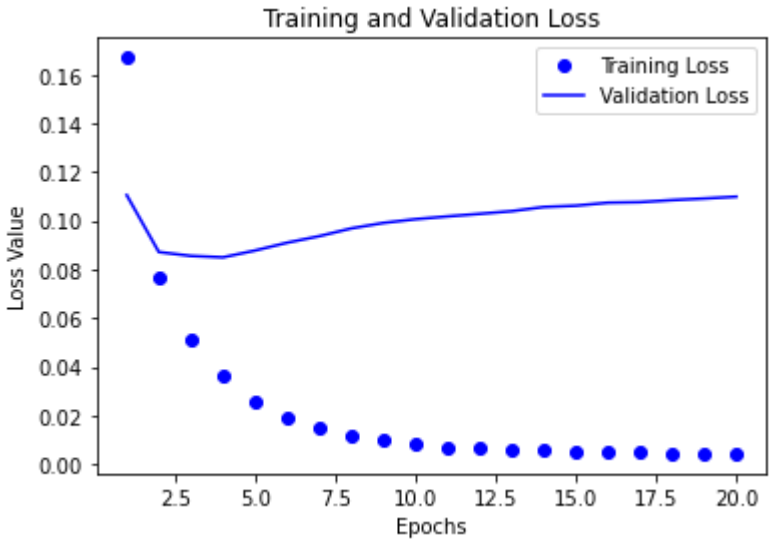
In [146]:

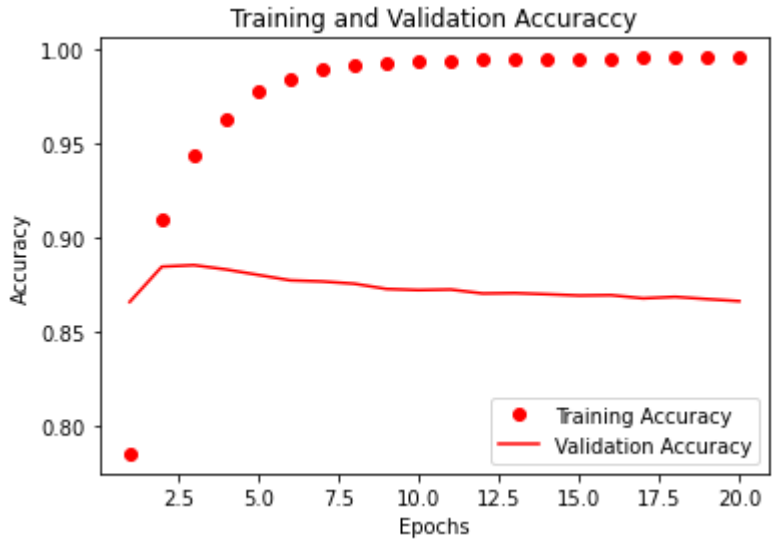
```
#implemented two hideden layer with 16 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 20ms/step - loss: 0.1670 - bi
nary_accuracy: 0.7849 - val_loss: 0.1105 - val_binary_accuracy: 0.8658
Epoch 2/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0770 - bi
nary_accuracy: 0.9101 - val_loss: 0.0871 - val_binary_accuracy: 0.8846
Epoch 3/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0508 - bi
nary_accuracy: 0.9439 - val_loss: 0.0855 - val_binary_accuracy: 0.8854
Epoch 4/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0360 - bi
nary_accuracy: 0.9634 - val_loss: 0.0850 - val_binary_accuracy: 0.8831
Epoch 5/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0260 - bi
nary_accuracy: 0.9776 - val_loss: 0.0878 - val_binary_accuracy: 0.8802
Epoch 6/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0192 - bi
nary_accuracy: 0.9850 - val_loss: 0.0910 - val_binary_accuracy: 0.8773
Epoch 7/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0147 - bi
nary_accuracy: 0.9897 - val_loss: 0.0937 - val_binary_accuracy: 0.8767
Epoch 8/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0117 - bi
nary_accuracy: 0.9917 - val_loss: 0.0969 - val_binary_accuracy: 0.8755
Epoch 9/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0097 - bi
nary_accuracy: 0.9928 - val_loss: 0.0991 - val_binary_accuracy: 0.8727
Epoch 10/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0081 - bin
ary_accuracy: 0.9939 - val_loss: 0.1006 - val_binary_accuracy: 0.8721
Epoch 11/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0071 - bi
```

```
nary_accuracy: 0.9945 - val_loss: 0.1018 - val_binary_accuracy: 0.8724
Epoch 12/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0065 - bi
nary_accuracy: 0.9949 - val_loss: 0.1028 - val_binary_accuracy: 0.8703
Epoch 13/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0060 - bin
ary_accuracy: 0.9951 - val_loss: 0.1039 - val_binary_accuracy: 0.8705
Epoch 14/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0055 - bi
nary_accuracy: 0.9953 - val_loss: 0.1056 - val_binary_accuracy: 0.8699
Epoch 15/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0052 - bi
nary_accuracy: 0.9955 - val_loss: 0.1062 - val_binary_accuracy: 0.8692
Epoch 16/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0050 - bi
nary_accuracy: 0.9955 - val_loss: 0.1074 - val_binary_accuracy: 0.8694
Epoch 17/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0047 - bi
nary_accuracy: 0.9958 - val_loss: 0.1076 - val_binary_accuracy: 0.8677
Epoch 18/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0045 - bin
ary_accuracy: 0.9959 - val_loss: 0.1085 - val_binary_accuracy: 0.8685
Epoch 19/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0044 - bin
ary_accuracy: 0.9959 - val_loss: 0.1091 - val_binary_accuracy: 0.8673
Epoch 20/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0043 - bin
ary_accuracy: 0.9960 - val_loss: 0.1099 - val_binary_accuracy: 0.8662
```





In [129]:
```
model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              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 9ms/step - loss: 0.1320 - accur
acy: 0.8261
Epoch 2/4
49/49 [==============================] - 0s 9ms/step - loss: 0.0622 - accur
acy: 0.9232
Epoch 3/4
49/49 [==============================] - 0s 9ms/step - loss: 0.0450 - accur
acy: 0.9473
Epoch 4/4
49/49 [==============================] - 0s 9ms/step - loss: 0.0347 - accur
acy: 0.9630
782/782 [==============================] - 1s 2ms/step - loss: 0.0949 - acc
uracy: 0.8724
```

Out[129]: [0.09486528486013412, 0.872439980506897]

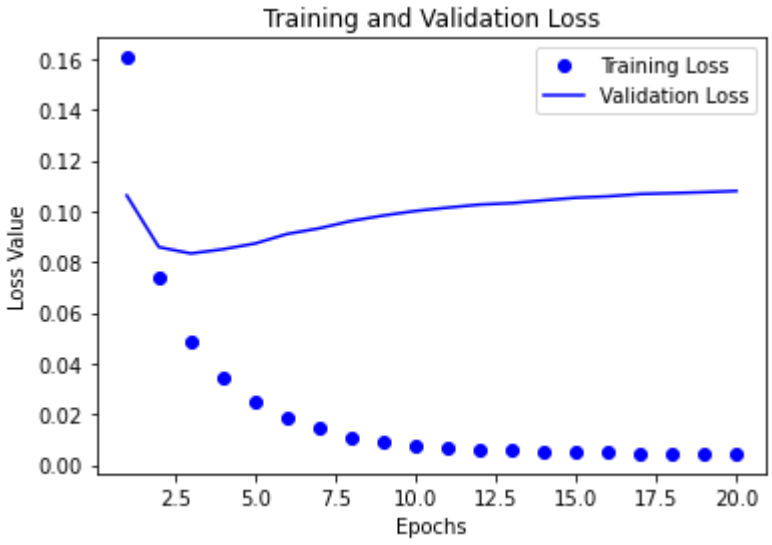
In [147]:

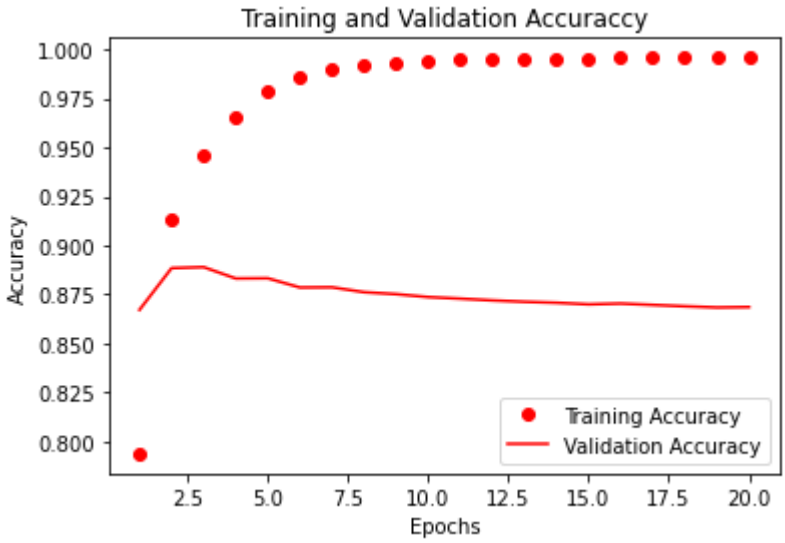
```
#implemented two hideden layer with 32 neurons and mse loss function with dro

from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 3s 85ms/step - loss: 0.1606 - bina
ry_accuracy: 0.7937 - val_loss: 0.1062 - val_binary_accuracy: 0.8671
Epoch 2/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0740 - bina
ry_accuracy: 0.9131 - val_loss: 0.0859 - val_binary_accuracy: 0.8884
Epoch 3/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0486 - bina
ry_accuracy: 0.9456 - val_loss: 0.0834 - val_binary_accuracy: 0.8889
Epoch 4/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0345 - bina
ry_accuracy: 0.9659 - val_loss: 0.0850 - val_binary_accuracy: 0.8831
Epoch 5/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0249 - bina
ry_accuracy: 0.9786 - val_loss: 0.0873 - val_binary_accuracy: 0.8832
Epoch 6/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0186 - bina
ry_accuracy: 0.9857 - val_loss: 0.0911 - val_binary_accuracy: 0.8785
Epoch 7/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0142 - bina
ry_accuracy: 0.9897 - val_loss: 0.0933 - val_binary_accuracy: 0.8786
Epoch 8/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0109 - bina
ry_accuracy: 0.9921 - val_loss: 0.0962 - val_binary_accuracy: 0.8761
Epoch 9/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0088 - bina
ry_accuracy: 0.9935 - val_loss: 0.0983 - val_binary_accuracy: 0.8751
Epoch 10/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0074 - bina
ry_accuracy: 0.9943 - val_loss: 0.1001 - val_binary_accuracy: 0.8736
Epoch 11/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0066 - binar
y_accuracy: 0.9948 - val_loss: 0.1014 - val_binary_accuracy: 0.8728
```

```
Epoch 12/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0060 - binar
y_accuracy: 0.9951 - val_loss: 0.1026 - val_binary_accuracy: 0.8719
Epoch 13/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0055 - bina
ry_accuracy: 0.9954 - val_loss: 0.1032 - val_binary_accuracy: 0.8712
Epoch 14/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0053 - bina
ry_accuracy: 0.9955 - val_loss: 0.1043 - val_binary_accuracy: 0.8707
Epoch 15/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0050 - bina
ry_accuracy: 0.9955 - val_loss: 0.1053 - val_binary_accuracy: 0.8699
Epoch 16/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0048 - bina
ry_accuracy: 0.9957 - val_loss: 0.1059 - val_binary_accuracy: 0.8703
Epoch 17/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0047 - bina
ry_accuracy: 0.9958 - val_loss: 0.1068 - val_binary_accuracy: 0.8696
Epoch 18/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0046 - bina
ry_accuracy: 0.9958 - val_loss: 0.1071 - val_binary_accuracy: 0.8689
Epoch 19/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0045 - binar
y_accuracy: 0.9958 - val_loss: 0.1075 - val_binary_accuracy: 0.8683
Epoch 20/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0044 - bina
ry_accuracy: 0.9959 - val_loss: 0.1080 - val_binary_accuracy: 0.8685
```





In [148]:
```
model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
layers.Dropout(0.5),
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
layers.Dropout(0.5),
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              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 6ms/step - loss: 0.1439 - accur
acy: 0.8234
Epoch 2/4
49/49 [==============================] - 0s 6ms/step - loss: 0.0682 - accur
acy: 0.9171
Epoch 3/4
49/49 [==============================] - 0s 6ms/step - loss: 0.0483 - accur
acy: 0.9448
Epoch 4/4
49/49 [==============================] - 0s 6ms/step - loss: 0.0375 - accur
acy: 0.9588
782/782 [==============================] - 1s 1ms/step - loss: 0.0926 - acc
uracy: 0.8751
```

Out[148]: [0.09261167049407959, 0.8750799894332886]

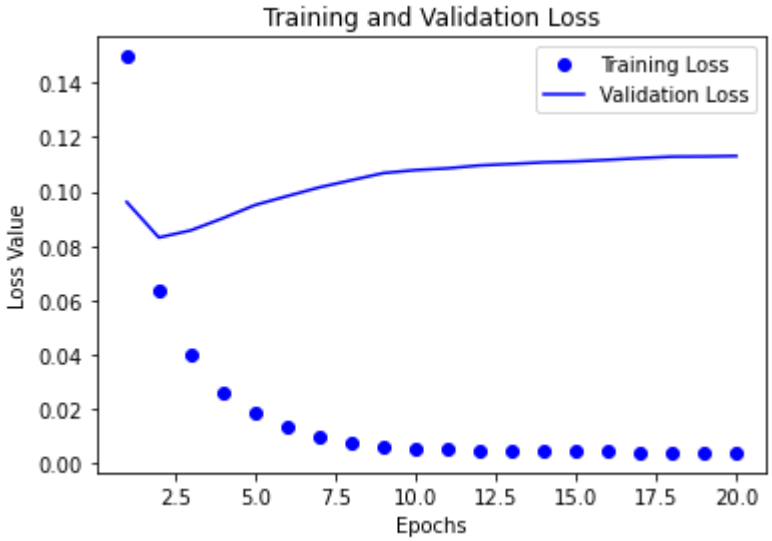
In [130]:

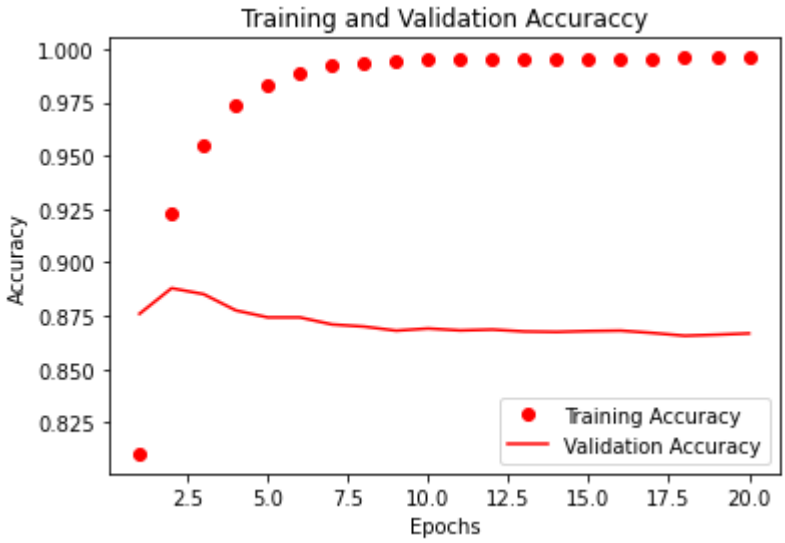
```
#implemented two hideden layer with 32 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 4s 25ms/step - loss: 0.1495 - bina
ry_accuracy: 0.8103 - val_loss: 0.0961 - val_binary_accuracy: 0.8759
Epoch 2/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0633 - bina
ry_accuracy: 0.9232 - val_loss: 0.0831 - val_binary_accuracy: 0.8879
Epoch 3/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0401 - bina
ry_accuracy: 0.9545 - val_loss: 0.0857 - val_binary_accuracy: 0.8851
Epoch 4/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0265 - bina
ry_accuracy: 0.9734 - val_loss: 0.0902 - val_binary_accuracy: 0.8775
Epoch 5/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0190 - bina
ry_accuracy: 0.9829 - val_loss: 0.0950 - val_binary_accuracy: 0.8742
Epoch 6/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0134 - bina
ry_accuracy: 0.9889 - val_loss: 0.0983 - val_binary_accuracy: 0.8742
Epoch 7/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0097 - bina
ry_accuracy: 0.9925 - val_loss: 0.1015 - val_binary_accuracy: 0.8709
Epoch 8/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0076 - bina
ry_accuracy: 0.9936 - val_loss: 0.1041 - val_binary_accuracy: 0.8699
Epoch 9/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0065 - bina
ry_accuracy: 0.9945 - val_loss: 0.1067 - val_binary_accuracy: 0.8680
Epoch 10/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0059 - bina
ry_accuracy: 0.9950 - val_loss: 0.1078 - val_binary_accuracy: 0.8690
Epoch 11/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0053 - bina
```

```
ry_accuracy: 0.9952 - val_loss: 0.1085 - val_binary_accuracy: 0.8681
Epoch 12/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0050 - bina
ry_accuracy: 0.9953 - val_loss: 0.1095 - val_binary_accuracy: 0.8685
Epoch 13/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0048 - bina
ry_accuracy: 0.9954 - val_loss: 0.1101 - val_binary_accuracy: 0.8676
Epoch 14/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0047 - bina
ry_accuracy: 0.9955 - val_loss: 0.1107 - val_binary_accuracy: 0.8674
Epoch 15/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0046 - bina
ry_accuracy: 0.9956 - val_loss: 0.1110 - val_binary_accuracy: 0.8678
Epoch 16/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0045 - bina
ry_accuracy: 0.9956 - val_loss: 0.1116 - val_binary_accuracy: 0.8680
Epoch 17/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0044 - bina
ry_accuracy: 0.9957 - val_loss: 0.1122 - val_binary_accuracy: 0.8669
Epoch 18/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0043 - bina
ry_accuracy: 0.9958 - val_loss: 0.1128 - val_binary_accuracy: 0.8656
Epoch 19/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0042 - bina
ry_accuracy: 0.9959 - val_loss: 0.1128 - val_binary_accuracy: 0.8661
Epoch 20/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0041 - bina
ry_accuracy: 0.9960 - val_loss: 0.1130 - val_binary_accuracy: 0.8667
```





In [131]:
```
model = models.Sequential()
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=2, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/2
49/49 [==============================] - 1s 9ms/step - loss: 0.1170 - accur
acy: 0.8422
Epoch 2/2
49/49 [==============================] - 0s 9ms/step - loss: 0.0552 - accur
acy: 0.9306
782/782 [==============================] - 2s 2ms/step - loss: 0.0889 - acc
uracy: 0.8805
```

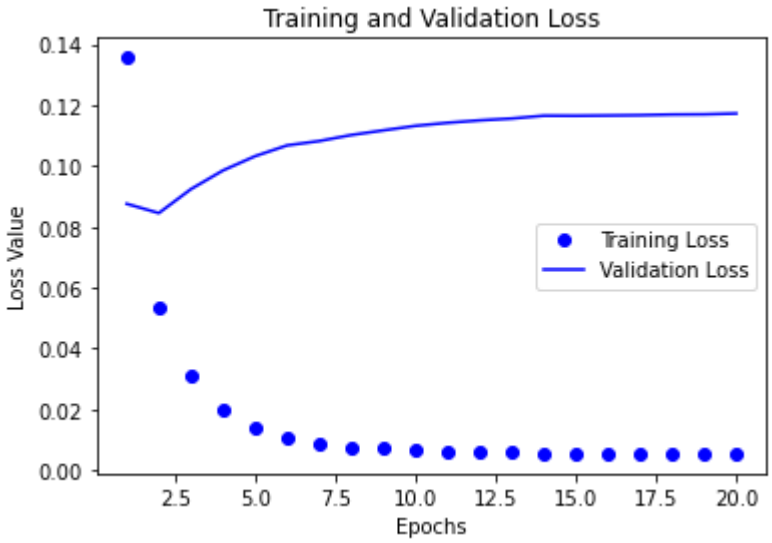
Out[131]:
```
[0.08891916275024414, 0.8804799914360046]
```

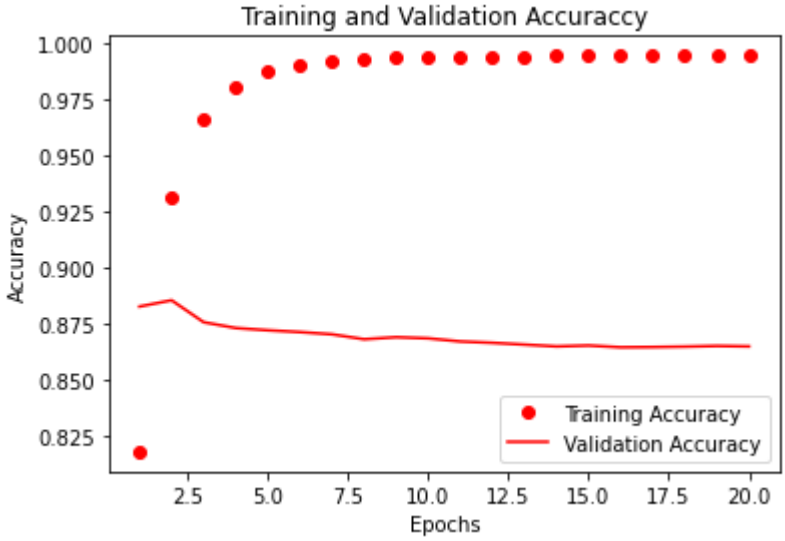
In [132]:
```
#implemented two hideden layer with 64 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 27ms/step - loss: 0.1356 - bi
nary_accuracy: 0.8178 - val_loss: 0.0875 - val_binary_accuracy: 0.8826
Epoch 2/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0532 - bi
nary_accuracy: 0.9316 - val_loss: 0.0845 - val_binary_accuracy: 0.8854
Epoch 3/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0311 - bi
nary_accuracy: 0.9661 - val_loss: 0.0924 - val_binary_accuracy: 0.8756
Epoch 4/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0198 - bi
nary_accuracy: 0.9803 - val_loss: 0.0985 - val_binary_accuracy: 0.8730
Epoch 5/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0135 - bi
nary_accuracy: 0.9879 - val_loss: 0.1032 - val_binary_accuracy: 0.8720
Epoch 6/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0104 - bi
nary_accuracy: 0.9907 - val_loss: 0.1068 - val_binary_accuracy: 0.8712
Epoch 7/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0087 - bi
nary_accuracy: 0.9920 - val_loss: 0.1082 - val_binary_accuracy: 0.8702
Epoch 8/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0076 - bi
nary_accuracy: 0.9931 - val_loss: 0.1102 - val_binary_accuracy: 0.8680
Epoch 9/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0071 - bi
nary_accuracy: 0.9936 - val_loss: 0.1117 - val_binary_accuracy: 0.8689
Epoch 10/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0063 - bi
nary_accuracy: 0.9941 - val_loss: 0.1132 - val_binary_accuracy: 0.8684
Epoch 11/20
```

```
30/30 [==============================] - 1s 19ms/step - loss: 0.0061 - bi
nary_accuracy: 0.9943 - val_loss: 0.1142 - val_binary_accuracy: 0.8670
Epoch 12/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0058 - bi
nary_accuracy: 0.9943 - val_loss: 0.1150 - val_binary_accuracy: 0.8664
Epoch 13/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0057 - bi
nary_accuracy: 0.9943 - val_loss: 0.1156 - val_binary_accuracy: 0.8656
Epoch 14/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0056 - bi
nary_accuracy: 0.9945 - val_loss: 0.1165 - val_binary_accuracy: 0.8648
Epoch 15/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0055 - bi
nary_accuracy: 0.9946 - val_loss: 0.1165 - val_binary_accuracy: 0.8652
Epoch 16/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0054 - bi
nary_accuracy: 0.9946 - val_loss: 0.1166 - val_binary_accuracy: 0.8644
Epoch 17/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0054 - bi
nary_accuracy: 0.9946 - val_loss: 0.1167 - val_binary_accuracy: 0.8645
Epoch 18/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0054 - bi
nary_accuracy: 0.9946 - val_loss: 0.1169 - val_binary_accuracy: 0.8647
Epoch 19/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0054 - bi
nary_accuracy: 0.9947 - val_loss: 0.1170 - val_binary_accuracy: 0.8650
Epoch 20/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0054 - bi
nary_accuracy: 0.9947 - val_loss: 0.1173 - val_binary_accuracy: 0.8648
```




Training and Validation Accuraccy
1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
Accuracy
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Epochs
Training Accuracy
Validation Accuracy

In [133]:
```
model = models.Sequential()
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=2, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/2
49/49 [==============================] - 1s 13ms/step - loss: 0.1132 - accu
racy: 0.8513
Epoch 2/2
49/49 [==============================] - 1s 13ms/step - loss: 0.0527 - accu
racy: 0.9348
782/782 [==============================] - 2s 2ms/step - loss: 0.0903 - acc
uracy: 0.8788
```

Out[133]:
```
[0.09031180292367935, 0.8788400292396545]
```

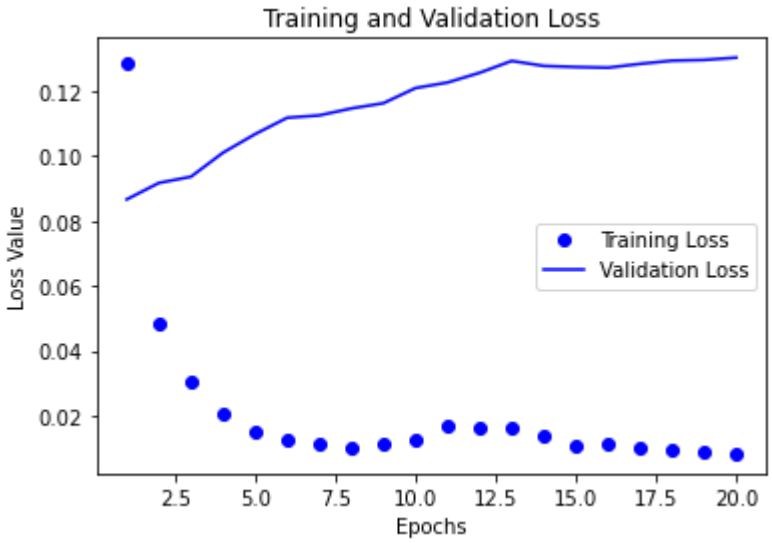
In [134]:

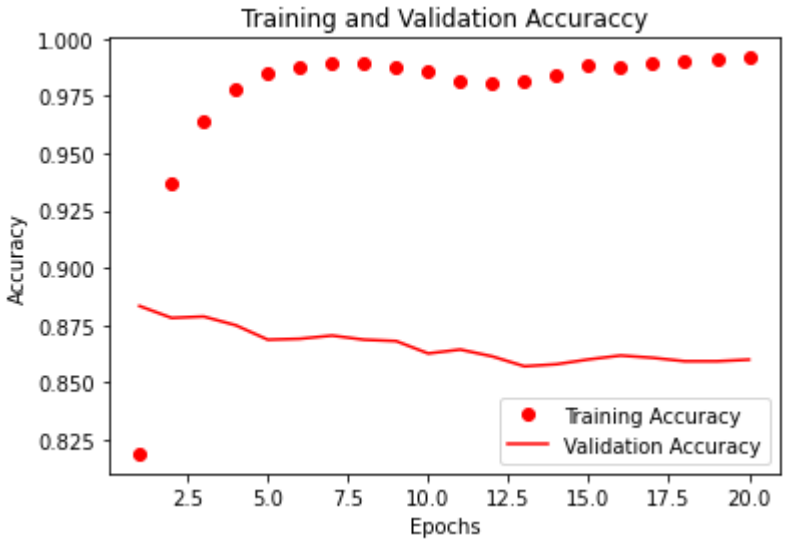
```
#implemented two hideden layer with 128 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 2s 36ms/step - loss: 0.1289 - bi
nary_accuracy: 0.8190 - val_loss: 0.0867 - val_binary_accuracy: 0.8834
Epoch 2/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0486 - bi
nary_accuracy: 0.9371 - val_loss: 0.0917 - val_binary_accuracy: 0.8782
Epoch 3/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0304 - bi
nary_accuracy: 0.9638 - val_loss: 0.0936 - val_binary_accuracy: 0.8788
Epoch 4/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0205 - bi
nary_accuracy: 0.9781 - val_loss: 0.1010 - val_binary_accuracy: 0.8750
Epoch 5/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0148 - bi
nary_accuracy: 0.9852 - val_loss: 0.1067 - val_binary_accuracy: 0.8687
Epoch 6/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0125 - bi
nary_accuracy: 0.9876 - val_loss: 0.1118 - val_binary_accuracy: 0.8691
Epoch 7/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0114 - bi
nary_accuracy: 0.9891 - val_loss: 0.1125 - val_binary_accuracy: 0.8705
Epoch 8/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0104 - bi
nary_accuracy: 0.9895 - val_loss: 0.1147 - val_binary_accuracy: 0.8687
Epoch 9/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0112 - bi
nary_accuracy: 0.9880 - val_loss: 0.1163 - val_binary_accuracy: 0.8681
Epoch 10/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0127 - bi
nary_accuracy: 0.9857 - val_loss: 0.1209 - val_binary_accuracy: 0.8627
Epoch 11/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0167 - bi
```

```
nary_accuracy: 0.9811 - val_loss: 0.1226 - val_binary_accuracy: 0.8644
Epoch 12/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0164 - bi
nary_accuracy: 0.9809 - val_loss: 0.1256 - val_binary_accuracy: 0.8614
Epoch 13/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0162 - bi
nary_accuracy: 0.9813 - val_loss: 0.1293 - val_binary_accuracy: 0.8571
Epoch 14/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0141 - bi
nary_accuracy: 0.9844 - val_loss: 0.1277 - val_binary_accuracy: 0.8580
Epoch 15/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0111 - bi
nary_accuracy: 0.9882 - val_loss: 0.1274 - val_binary_accuracy: 0.8601
Epoch 16/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0114 - bi
nary_accuracy: 0.9880 - val_loss: 0.1272 - val_binary_accuracy: 0.8618
Epoch 17/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0102 - bi
nary_accuracy: 0.9892 - val_loss: 0.1284 - val_binary_accuracy: 0.8608
Epoch 18/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0097 - bi
nary_accuracy: 0.9901 - val_loss: 0.1294 - val_binary_accuracy: 0.8593
Epoch 19/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0088 - bi
nary_accuracy: 0.9912 - val_loss: 0.1296 - val_binary_accuracy: 0.8593
Epoch 20/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0084 - bi
nary_accuracy: 0.9917 - val_loss: 0.1303 - val_binary_accuracy: 0.8600
```





In [135]:
```
model = models.Sequential()
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=3, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/3
49/49 [==============================] - 1s 20ms/step - loss: 0.1083 - accu
racy: 0.8507
Epoch 2/3
49/49 [==============================] - 1s 20ms/step - loss: 0.0528 - accu
racy: 0.9337
Epoch 3/3
49/49 [==============================] - 1s 22ms/step - loss: 0.0372 - accu
racy: 0.9561
782/782 [==============================] - 3s 3ms/step - loss: 0.1050 - acc
uracy: 0.8688
```

Out[135]:
```
[0.10499988496303558, 0.8687599897384644]
```

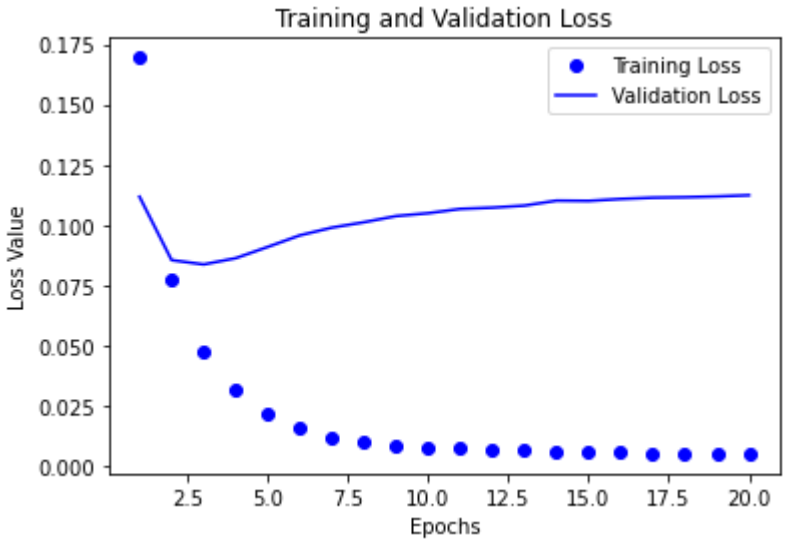
In [136]:

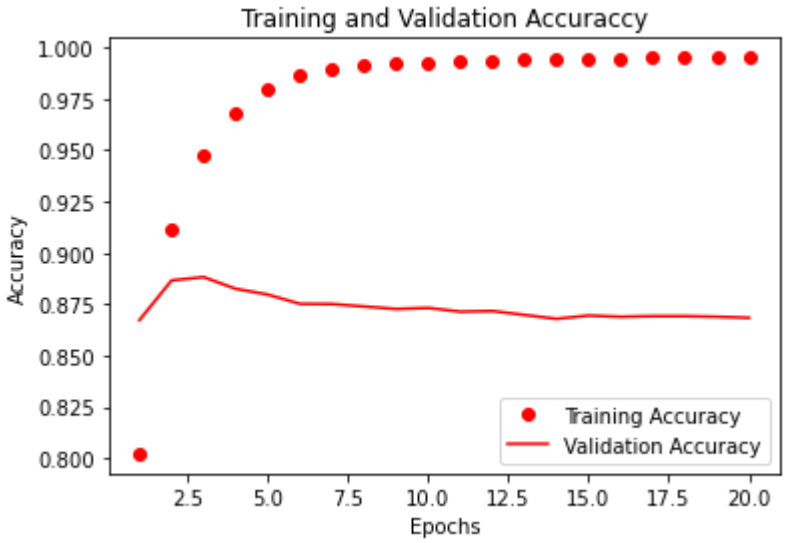
```
#implemented three hideden layer with 16 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 23ms/step - loss: 0.1694 - bina
ry_accuracy: 0.8023 - val_loss: 0.1117 - val_binary_accuracy: 0.8673
Epoch 2/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0772 - bina
ry_accuracy: 0.9112 - val_loss: 0.0854 - val_binary_accuracy: 0.8866
Epoch 3/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0475 - bina
ry_accuracy: 0.9471 - val_loss: 0.0836 - val_binary_accuracy: 0.8882
Epoch 4/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0313 - bina
ry_accuracy: 0.9682 - val_loss: 0.0862 - val_binary_accuracy: 0.8825
Epoch 5/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0216 - bina
ry_accuracy: 0.9800 - val_loss: 0.0909 - val_binary_accuracy: 0.8797
Epoch 6/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0156 - bina
ry_accuracy: 0.9865 - val_loss: 0.0957 - val_binary_accuracy: 0.8752
Epoch 7/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0119 - bina
ry_accuracy: 0.9898 - val_loss: 0.0989 - val_binary_accuracy: 0.8751
Epoch 8/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0103 - bina
ry_accuracy: 0.9911 - val_loss: 0.1011 - val_binary_accuracy: 0.8739
Epoch 9/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0088 - bina
ry_accuracy: 0.9921 - val_loss: 0.1037 - val_binary_accuracy: 0.8727
Epoch 10/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0079 - bina
ry_accuracy: 0.9925 - val_loss: 0.1049 - val_binary_accuracy: 0.8732
Epoch 11/20
```

```
30/30 [==============================] - 0s 12ms/step - loss: 0.0072 - bina
ry_accuracy: 0.9931 - val_loss: 0.1067 - val_binary_accuracy: 0.8714
Epoch 12/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0067 - bina
ry_accuracy: 0.9937 - val_loss: 0.1072 - val_binary_accuracy: 0.8717
Epoch 13/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0064 - bina
ry_accuracy: 0.9939 - val_loss: 0.1080 - val_binary_accuracy: 0.8698
Epoch 14/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0061 - bina
ry_accuracy: 0.9941 - val_loss: 0.1101 - val_binary_accuracy: 0.8679
Epoch 15/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0059 - bina
ry_accuracy: 0.9943 - val_loss: 0.1100 - val_binary_accuracy: 0.8695
Epoch 16/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0055 - bina
ry_accuracy: 0.9947 - val_loss: 0.1108 - val_binary_accuracy: 0.8689
Epoch 17/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0054 - bina
ry_accuracy: 0.9948 - val_loss: 0.1113 - val_binary_accuracy: 0.8692
Epoch 18/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0053 - bina
ry_accuracy: 0.9948 - val_loss: 0.1115 - val_binary_accuracy: 0.8692
Epoch 19/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0052 - bina
ry_accuracy: 0.9949 - val_loss: 0.1118 - val_binary_accuracy: 0.8689
Epoch 20/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0051 - bina
ry_accuracy: 0.9949 - val_loss: 0.1123 - val_binary_accuracy: 0.8684
```




Training and Validation Accuraccy
Accuracy
Epochs
1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0.800
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Training Accuracy
Validation Accuracy

In [99]:
```
model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=3, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/3
49/49 [==============================] - 1s 8ms/step - loss: 0.1415 - accur
acy: 0.8225
Epoch 2/3
49/49 [==============================] - 0s 8ms/step - loss: 0.0614 - accur
acy: 0.9236
Epoch 3/3
49/49 [==============================] - 0s 7ms/step - loss: 0.0431 - accur
acy: 0.9493
782/782 [==============================] - 2s 2ms/step - loss: 0.0937 - acc
uracy: 0.8745
```

Out[99]:
```
[0.09372340887784958, 0.8744800090789795]
```

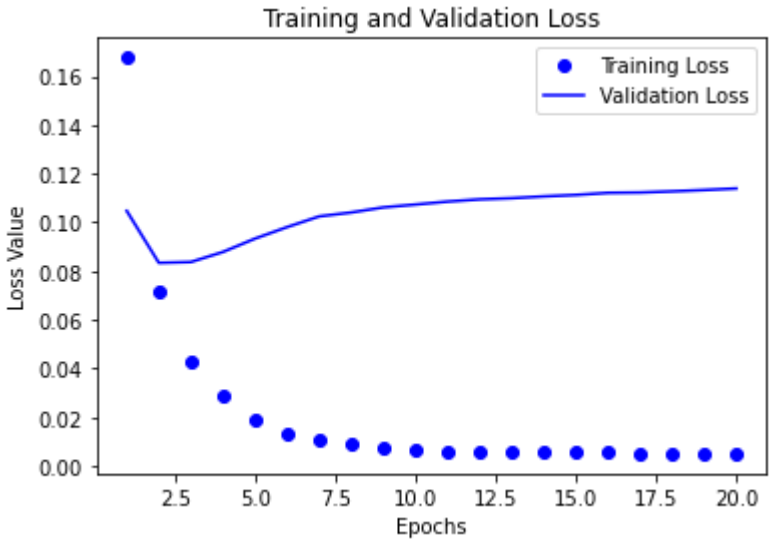
In [149]:

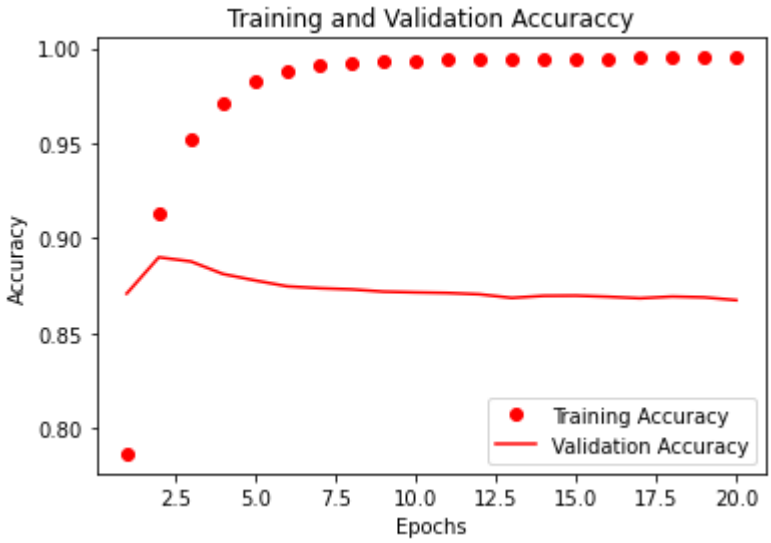
```
#implemented three hideden layer with 16 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 17ms/step - loss: 0.1676 - bi
nary_accuracy: 0.7865 - val_loss: 0.1047 - val_binary_accuracy: 0.8708
Epoch 2/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0712 - bi
nary_accuracy: 0.9133 - val_loss: 0.0834 - val_binary_accuracy: 0.8899
Epoch 3/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0428 - bi
nary_accuracy: 0.9517 - val_loss: 0.0837 - val_binary_accuracy: 0.8877
Epoch 4/20
30/30 [==============================] - 0s 11ms/step - loss: 0.0285 - bi
nary_accuracy: 0.9707 - val_loss: 0.0878 - val_binary_accuracy: 0.8811
Epoch 5/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0188 - bi
nary_accuracy: 0.9823 - val_loss: 0.0933 - val_binary_accuracy: 0.8777
Epoch 6/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0135 - bi
nary_accuracy: 0.9882 - val_loss: 0.0980 - val_binary_accuracy: 0.8746
Epoch 7/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0107 - bi
nary_accuracy: 0.9907 - val_loss: 0.1024 - val_binary_accuracy: 0.8736
Epoch 8/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0088 - bi
nary_accuracy: 0.9925 - val_loss: 0.1040 - val_binary_accuracy: 0.8730
Epoch 9/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0077 - bin
ary_accuracy: 0.9929 - val_loss: 0.1061 - val_binary_accuracy: 0.8718
Epoch 10/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0068 - bi
nary_accuracy: 0.9935 - val_loss: 0.1073 - val_binary_accuracy: 0.8714
Epoch 11/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0061 - bi
```

```
nary_accuracy: 0.9943 - val_loss: 0.1085 - val_binary_accuracy: 0.8711
Epoch 12/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0059 - bi
nary_accuracy: 0.9944 - val_loss: 0.1094 - val_binary_accuracy: 0.8704
Epoch 13/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0058 - bi
nary_accuracy: 0.9944 - val_loss: 0.1099 - val_binary_accuracy: 0.8685
Epoch 14/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0056 - bi
nary_accuracy: 0.9945 - val_loss: 0.1106 - val_binary_accuracy: 0.8696
Epoch 15/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0055 - bin
ary_accuracy: 0.9947 - val_loss: 0.1112 - val_binary_accuracy: 0.8697
Epoch 16/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0054 - bin
ary_accuracy: 0.9947 - val_loss: 0.1121 - val_binary_accuracy: 0.8691
Epoch 17/20
30/30 [==============================] - 0s 9ms/step - loss: 0.0053 - bin
ary_accuracy: 0.9948 - val_loss: 0.1122 - val_binary_accuracy: 0.8683
Epoch 18/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0051 - bi
nary_accuracy: 0.9949 - val_loss: 0.1127 - val_binary_accuracy: 0.8692
Epoch 19/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0051 - bi
nary_accuracy: 0.9950 - val_loss: 0.1133 - val_binary_accuracy: 0.8688
Epoch 20/20
30/30 [==============================] - 0s 10ms/step - loss: 0.0050 - bi
nary_accuracy: 0.9951 - val_loss: 0.1139 - val_binary_accuracy: 0.8673
```





In [150]:
```
model = models.Sequential()
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
layers.Dropout(0.5),
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
layers.Dropout(0.5),
model.add(layers.Dense(16, activation='tanh', input_shape=(10000,)))
layers.Dropout(0.5),
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=3, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/3
49/49 [==============================] - 1s 7ms/step - loss: 0.1406 - accur
acy: 0.8263
Epoch 2/3
49/49 [==============================] - 0s 8ms/step - loss: 0.0614 - accur
acy: 0.9235
Epoch 3/3
49/49 [==============================] - 0s 7ms/step - loss: 0.0422 - accur
acy: 0.9502
782/782 [==============================] - 1s 894us/step - loss: 0.0915 - a
ccuracy: 0.8787
```

Out[150]: [0.09145137667655945, 0.8787199854850769]

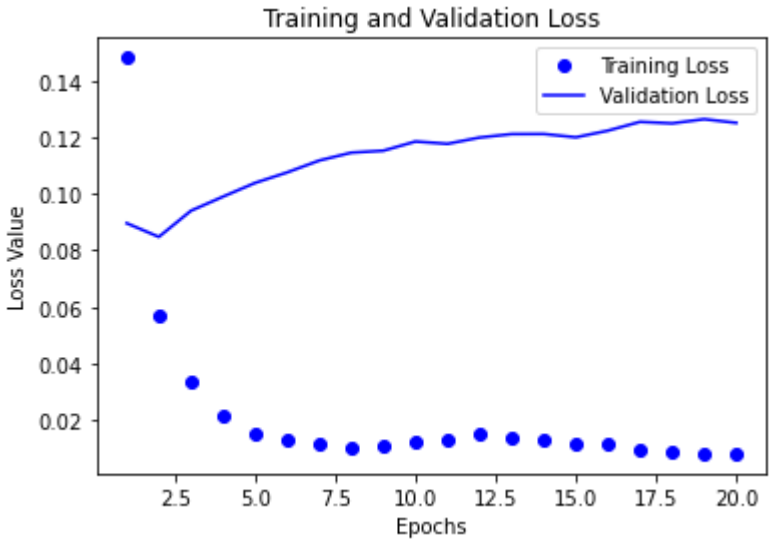
In [137]:

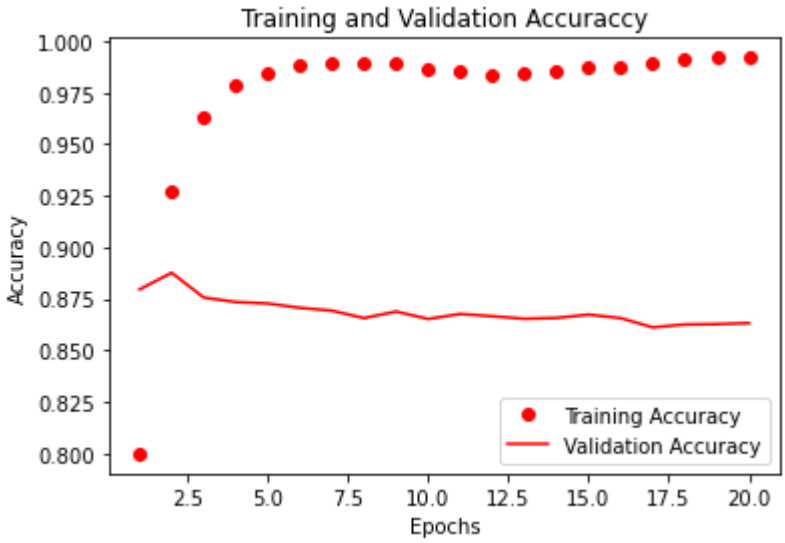
```
#implemented three hideden layer with 32 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 22ms/step - loss: 0.1481 - bina
ry_accuracy: 0.7999 - val_loss: 0.0896 - val_binary_accuracy: 0.8795
Epoch 2/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0570 - bina
ry_accuracy: 0.9267 - val_loss: 0.0848 - val_binary_accuracy: 0.8876
Epoch 3/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0333 - bina
ry_accuracy: 0.9628 - val_loss: 0.0940 - val_binary_accuracy: 0.8756
Epoch 4/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0216 - bina
ry_accuracy: 0.9785 - val_loss: 0.0990 - val_binary_accuracy: 0.8734
Epoch 5/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0155 - bina
ry_accuracy: 0.9845 - val_loss: 0.1038 - val_binary_accuracy: 0.8727
Epoch 6/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0128 - bina
ry_accuracy: 0.9879 - val_loss: 0.1075 - val_binary_accuracy: 0.8706
Epoch 7/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0114 - bina
ry_accuracy: 0.9893 - val_loss: 0.1117 - val_binary_accuracy: 0.8692
Epoch 8/20
30/30 [==============================] - 0s 15ms/step - loss: 0.0105 - bina
ry_accuracy: 0.9896 - val_loss: 0.1145 - val_binary_accuracy: 0.8656
Epoch 9/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0108 - bina
ry_accuracy: 0.9893 - val_loss: 0.1152 - val_binary_accuracy: 0.8688
Epoch 10/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0123 - bina
ry_accuracy: 0.9867 - val_loss: 0.1185 - val_binary_accuracy: 0.8652
Epoch 11/20
```

```
30/30 [==============================] - 0s 13ms/step - loss: 0.0133 - bina
ry_accuracy: 0.9853 - val_loss: 0.1176 - val_binary_accuracy: 0.8676
Epoch 12/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0149 - bina
ry_accuracy: 0.9834 - val_loss: 0.1199 - val_binary_accuracy: 0.8665
Epoch 13/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0140 - bina
ry_accuracy: 0.9841 - val_loss: 0.1211 - val_binary_accuracy: 0.8653
Epoch 14/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0130 - bina
ry_accuracy: 0.9853 - val_loss: 0.1211 - val_binary_accuracy: 0.8657
Epoch 15/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0120 - bina
ry_accuracy: 0.9873 - val_loss: 0.1199 - val_binary_accuracy: 0.8673
Epoch 16/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0113 - bina
ry_accuracy: 0.9877 - val_loss: 0.1223 - val_binary_accuracy: 0.8656
Epoch 17/20
30/30 [==============================] - 0s 12ms/step - loss: 0.0098 - bina
ry_accuracy: 0.9896 - val_loss: 0.1254 - val_binary_accuracy: 0.8611
Epoch 18/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0089 - bina
ry_accuracy: 0.9910 - val_loss: 0.1249 - val_binary_accuracy: 0.8625
Epoch 19/20
30/30 [==============================] - 0s 14ms/step - loss: 0.0083 - bina
ry_accuracy: 0.9919 - val_loss: 0.1264 - val_binary_accuracy: 0.8627
Epoch 20/20
30/30 [==============================] - 0s 13ms/step - loss: 0.0082 - bina
ry_accuracy: 0.9919 - val_loss: 0.1251 - val_binary_accuracy: 0.8632
```





In [138]:
```
model = models.Sequential()
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(32, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=2, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/2
49/49 [==============================] - 1s 9ms/step - loss: 0.1190 - accur
acy: 0.8395
Epoch 2/2
49/49 [==============================] - 0s 9ms/step - loss: 0.0547 - accur
acy: 0.9307
782/782 [==============================] - 2s 3ms/step - loss: 0.0904 - acc
uracy: 0.8791
```

Out[138]: [0.09038501232862473, 0.8790799975395203]

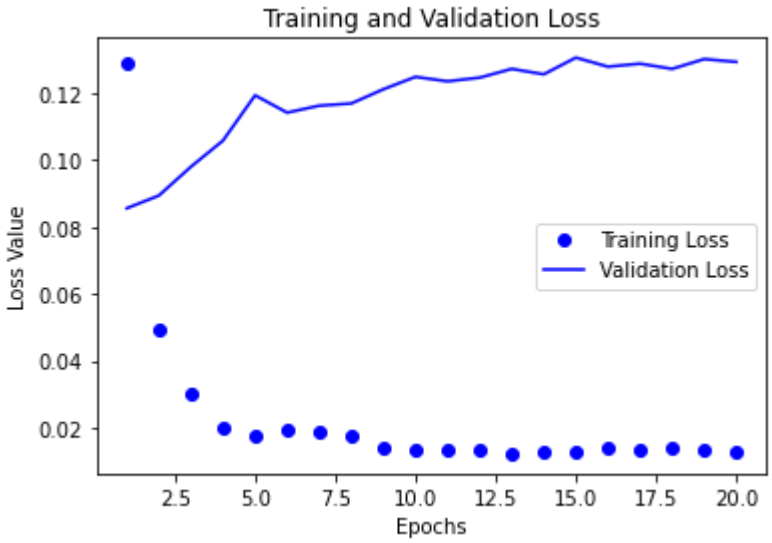
In [139]:

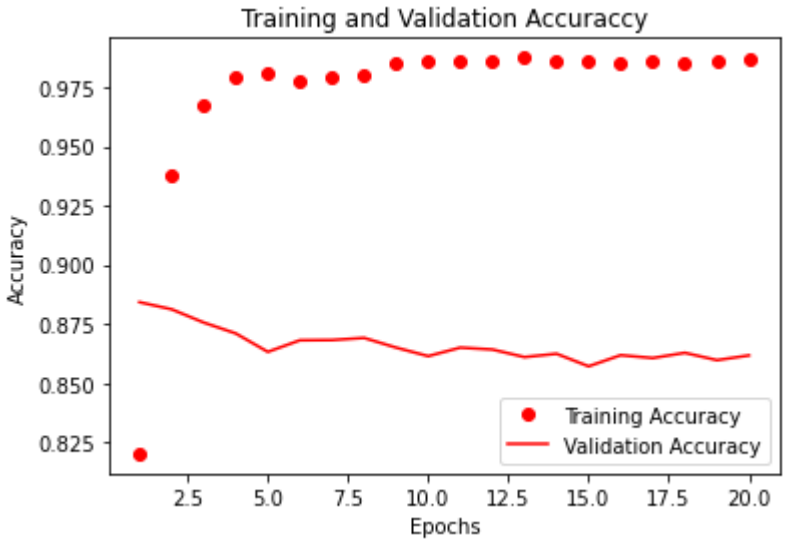
```
#implemented three hideden layer with 64 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 26ms/step - loss: 0.1289 - bi
nary_accuracy: 0.8202 - val_loss: 0.0856 - val_binary_accuracy: 0.8842
Epoch 2/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0494 - bi
nary_accuracy: 0.9375 - val_loss: 0.0894 - val_binary_accuracy: 0.8812
Epoch 3/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0299 - bi
nary_accuracy: 0.9673 - val_loss: 0.0980 - val_binary_accuracy: 0.8756
Epoch 4/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0198 - bi
nary_accuracy: 0.9789 - val_loss: 0.1059 - val_binary_accuracy: 0.8710
Epoch 5/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0177 - bi
nary_accuracy: 0.9809 - val_loss: 0.1194 - val_binary_accuracy: 0.8632
Epoch 6/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0197 - bi
nary_accuracy: 0.9776 - val_loss: 0.1142 - val_binary_accuracy: 0.8681
Epoch 7/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0186 - bi
nary_accuracy: 0.9789 - val_loss: 0.1163 - val_binary_accuracy: 0.8682
Epoch 8/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0175 - bi
nary_accuracy: 0.9800 - val_loss: 0.1170 - val_binary_accuracy: 0.8691
Epoch 9/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0142 - bi
nary_accuracy: 0.9848 - val_loss: 0.1212 - val_binary_accuracy: 0.8650
Epoch 10/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0134 - bi
nary_accuracy: 0.9860 - val_loss: 0.1249 - val_binary_accuracy: 0.8614
Epoch 11/20
```

```
30/30 [==============================] - 1s 19ms/step - loss: 0.0134 - bi
nary_accuracy: 0.9859 - val_loss: 0.1236 - val_binary_accuracy: 0.8650
Epoch 12/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0132 - bi
nary_accuracy: 0.9863 - val_loss: 0.1247 - val_binary_accuracy: 0.8642
Epoch 13/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0122 - bi
nary_accuracy: 0.9875 - val_loss: 0.1273 - val_binary_accuracy: 0.8610
Epoch 14/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0129 - bi
nary_accuracy: 0.9858 - val_loss: 0.1257 - val_binary_accuracy: 0.8624
Epoch 15/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0128 - bi
nary_accuracy: 0.9865 - val_loss: 0.1307 - val_binary_accuracy: 0.8571
Epoch 16/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0139 - bi
nary_accuracy: 0.9851 - val_loss: 0.1280 - val_binary_accuracy: 0.8618
Epoch 17/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0133 - bi
nary_accuracy: 0.9859 - val_loss: 0.1289 - val_binary_accuracy: 0.8606
Epoch 18/20
30/30 [==============================] - 1s 21ms/step - loss: 0.0139 - bi
nary_accuracy: 0.9852 - val_loss: 0.1273 - val_binary_accuracy: 0.8628
Epoch 19/20
30/30 [==============================] - 1s 20ms/step - loss: 0.0133 - bi
nary_accuracy: 0.9858 - val_loss: 0.1302 - val_binary_accuracy: 0.8597
Epoch 20/20
30/30 [==============================] - 1s 19ms/step - loss: 0.0126 - bi
nary_accuracy: 0.9867 - val_loss: 0.1294 - val_binary_accuracy: 0.8617
```





In [143]:
```
model = models.Sequential()
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(64, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=2, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/2
49/49 [==============================] - 3s 10ms/step - loss: 0.1111 - accu
racy: 0.8491
Epoch 2/2
49/49 [==============================] - 1s 11ms/step - loss: 0.0523 - accu
racy: 0.9348
782/782 [==============================] - 1s 1ms/step - loss: 0.0936 - acc
uracy: 0.8766
```

Out[143]: [0.09357214719057083, 0.8766400218009949]

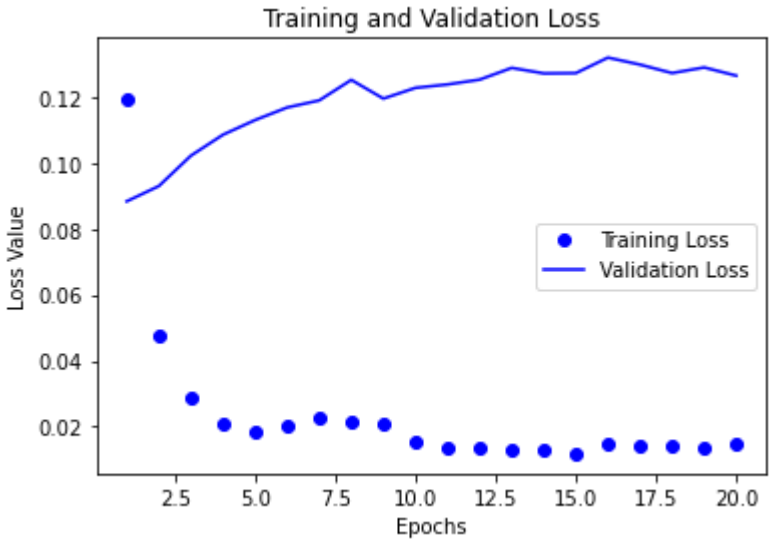
In [141]:

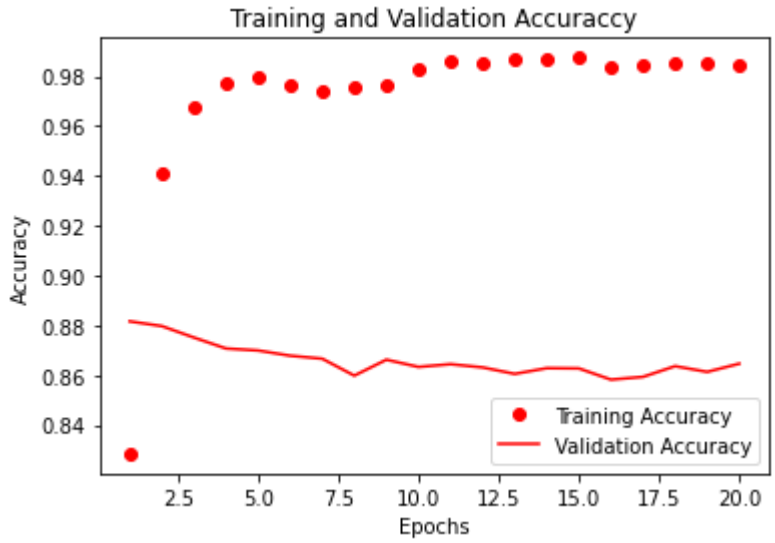
```
#implemented three hideden layer with 128 neurons and mse loss function


from keras import models
from tensorflow.keras import layers

model = models.Sequential()
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])

from keras import optimizers
from keras import losses
from keras import metrics

from tensorflow import keras
from keras import optimizers
from tensorflow.keras import optimizers
from tensorflow.keras import optimizers

model.compile(optimizer='adam',
              loss = losses.mse,
              metrics = [metrics.binary_accuracy])

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

y_val = y_train[:10000]
partial_y_train = y_train[10000:]

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()
# Plotting the training and validation loss

import matplotlib.pyplot as plt
%matplotlib inline

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 Value')
plt.legend()

plt.show()


# Plotting the training and validation accuracy
# Training and Validation Accuracy

acc_values = history_dict['binary_accuracy']
val_acc_values = history_dict['val_binary_accuracy']

epochs = range(1, len(loss_values) + 1)

plt.plot(epochs, acc_values, 'ro', label="Training Accuracy")
plt.plot(epochs, val_acc_values, 'r', label="Validation Accuracy")

plt.title('Training and Validation Accuraccy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.show()
```

```
Epoch 1/20
30/30 [==============================] - 1s 38ms/step - loss: 0.1192 - bi
nary_accuracy: 0.8285 - val_loss: 0.0884 - val_binary_accuracy: 0.8816
Epoch 2/20
30/30 [==============================] - 1s 30ms/step - loss: 0.0478 - bi
nary_accuracy: 0.9411 - val_loss: 0.0930 - val_binary_accuracy: 0.8797
Epoch 3/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0289 - bi
nary_accuracy: 0.9676 - val_loss: 0.1022 - val_binary_accuracy: 0.8751
Epoch 4/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0208 - bi
nary_accuracy: 0.9776 - val_loss: 0.1086 - val_binary_accuracy: 0.8707
Epoch 5/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0183 - bi
nary_accuracy: 0.9798 - val_loss: 0.1130 - val_binary_accuracy: 0.8699
Epoch 6/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0200 - bi
nary_accuracy: 0.9768 - val_loss: 0.1168 - val_binary_accuracy: 0.8678
Epoch 7/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0229 - bi
nary_accuracy: 0.9737 - val_loss: 0.1190 - val_binary_accuracy: 0.8666
Epoch 8/20
30/30 [==============================] - 1s 30ms/step - loss: 0.0214 - bi
nary_accuracy: 0.9754 - val_loss: 0.1252 - val_binary_accuracy: 0.8598
Epoch 9/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0206 - bi
nary_accuracy: 0.9763 - val_loss: 0.1195 - val_binary_accuracy: 0.8662
Epoch 10/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0152 - bi
nary_accuracy: 0.9831 - val_loss: 0.1228 - val_binary_accuracy: 0.8633
Epoch 11/20
```

```
30/30 [==============================] - 1s 26ms/step - loss: 0.0134 - bi
nary_accuracy: 0.9860 - val_loss: 0.1238 - val_binary_accuracy: 0.8644
Epoch 12/20
30/30 [==============================] - 1s 28ms/step - loss: 0.0135 - bi
nary_accuracy: 0.9851 - val_loss: 0.1253 - val_binary_accuracy: 0.8631
Epoch 13/20
30/30 [==============================] - 1s 29ms/step - loss: 0.0128 - bi
nary_accuracy: 0.9866 - val_loss: 0.1288 - val_binary_accuracy: 0.8605
Epoch 14/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0127 - bi
nary_accuracy: 0.9868 - val_loss: 0.1272 - val_binary_accuracy: 0.8628
Epoch 15/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0118 - bi
nary_accuracy: 0.9875 - val_loss: 0.1272 - val_binary_accuracy: 0.8627
Epoch 16/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0148 - bi
nary_accuracy: 0.9841 - val_loss: 0.1320 - val_binary_accuracy: 0.8582
Epoch 17/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0144 - bi
nary_accuracy: 0.9845 - val_loss: 0.1298 - val_binary_accuracy: 0.8593
Epoch 18/20
30/30 [==============================] - 1s 27ms/step - loss: 0.0139 - bi
nary_accuracy: 0.9855 - val_loss: 0.1273 - val_binary_accuracy: 0.8636
Epoch 19/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0135 - bi
nary_accuracy: 0.9854 - val_loss: 0.1290 - val_binary_accuracy: 0.8613
Epoch 20/20
30/30 [==============================] - 1s 26ms/step - loss: 0.0145 - bi
nary_accuracy: 0.9842 - val_loss: 0.1265 - val_binary_accuracy: 0.8646
```





In [142]:
```
model = models.Sequential()
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(128, activation='tanh', input_shape=(10000,)))
model.add(layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',
              loss='mse',
              metrics=['accuracy'])
model.fit(x_train, y_train, epochs=2, batch_size=512)
results = model.evaluate(x_test, y_test)

results
```

```
Epoch 1/2
49/49 [==============================] - 2s 24ms/step - loss: 0.1085 - accu
racy: 0.8490
Epoch 2/2
49/49 [==============================] - 1s 21ms/step - loss: 0.0513 - accu
racy: 0.9358
782/782 [==============================] - 2s 2ms/step - loss: 0.0963 - acc
uracy: 0.8737
```

Out[142]:
```
[0.0962989553809166, 0.8736799955368042]
```

In [ ]:
```

```