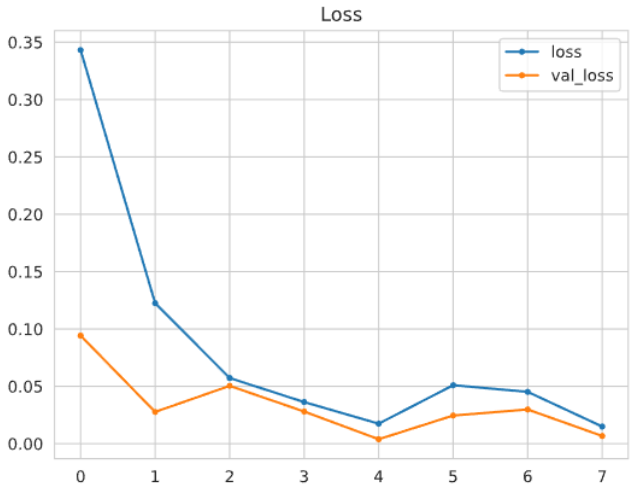
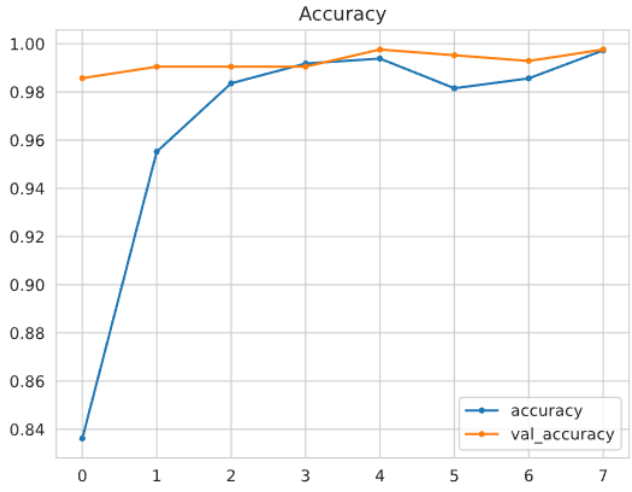
v2:

```
def create_tf_model():
    model = Sequential()

    model.add(Conv2D(filters=32, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=64, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=64, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())
    model.add(Dense(128, activation = 'relu'))

    model.add(Dropout(0.5))
    model.add(Dense(1, activation = 'sigmoid'))

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

    return model
```

```
early_stop = EarlyStopping(monitor='val_loss',patience=3)
```






```
    evaluation = model.evaluate(test_set)
```

```
43/43 [==============================] - 16s 364ms/step - loss: 0.0039 - accuracy: 0.9976
```

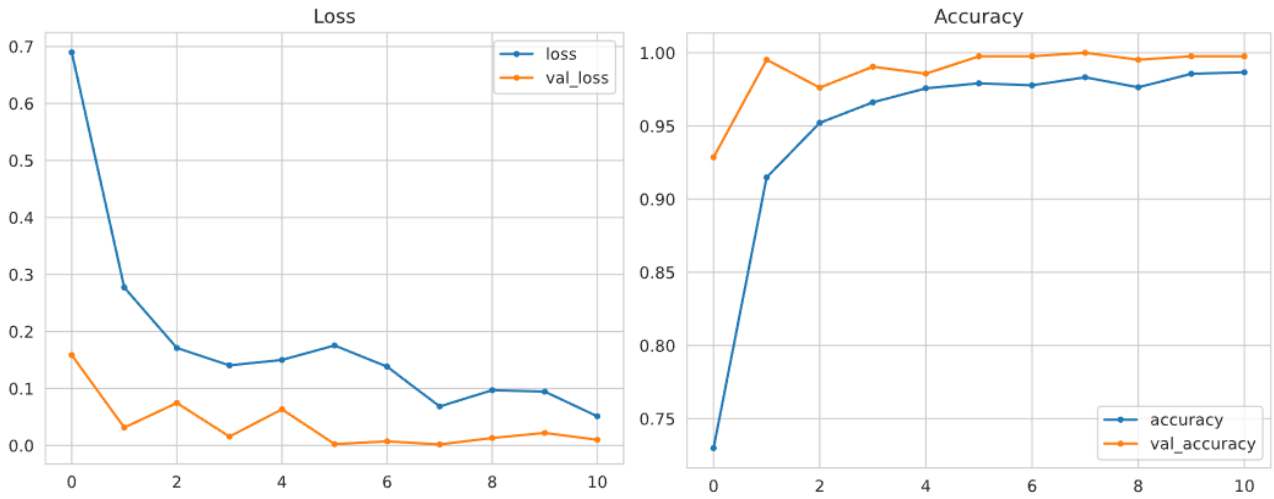
```
0.9999896963781794
healthy
```

Model accuracy plot suggests slight overfitting may occur if further epochs were run as the accuracy and val_accuracy are the same at the end of the curve.

v3:

```
def create_tf_model():
    model = Sequential()

    model.add(Conv2D(filters=32, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=64, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=64, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())
    model.add(Dense(128, activation = 'relu'))

    model.add(Dropout(0.5))
    model.add(Dense(1, activation = 'sigmoid'))

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

    return model
```

```
early_stop = EarlyStopping(monitor='val_loss',patience=3)
```




```
evaluation = model.evaluate(test_set)
```

```
43/43 [==============================] - 17s 382ms/step - loss: 0.0441 - accuracy: 0.9953
```

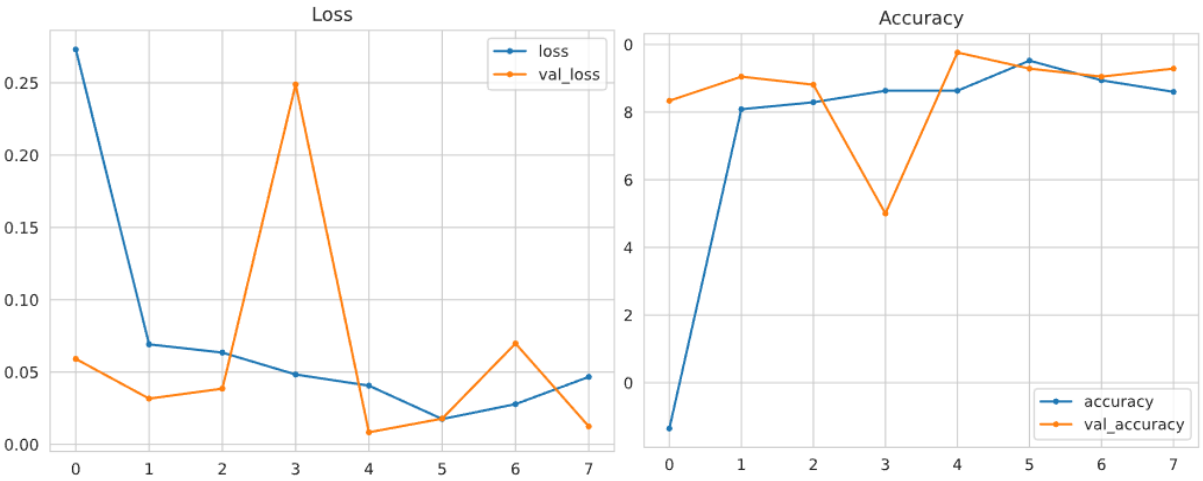
```
0.9999999817082621
healthy
```

Model accuracy plot suggests a normal fit. The loss models plot suggests that better learning is achievable.

v4:

```
def create_tf_model():
    model = Sequential()

    model.add(Conv2D(filters=32, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=64, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=64, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())
    model.add(Dense(128, activation = 'relu'))

    model.add(Dropout(0.5))
    model.add(Dense(2, activation = 'softmax'))

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

    return model
```

```
early_stop = EarlyStopping(monitor='val_loss',patience=3)
```




```
    evaluation = model.evaluate(test_set)
```

```
43/43 [==============================] - 15s 348ms/step - loss: 0.0116 - accuracy: 0.9964
```

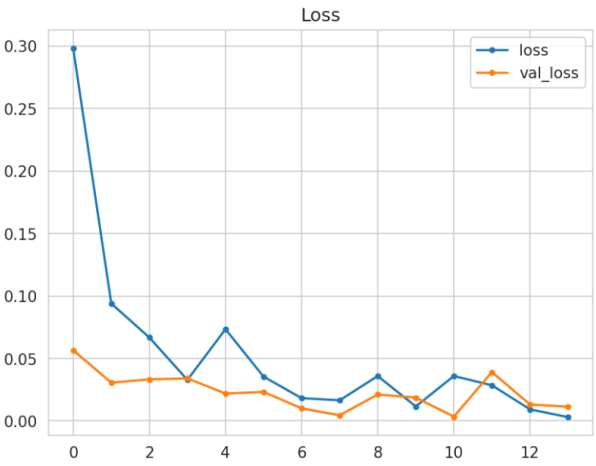
```
0.99985063
powdery_mildew
```

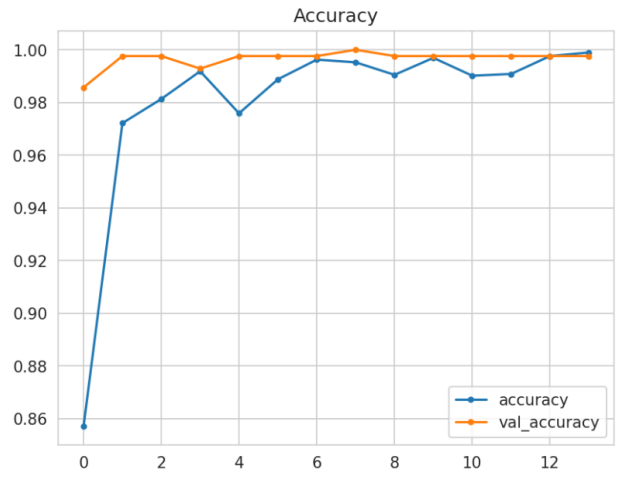
The model plots suggest poor learning. As the prediction was incorrect there is a large possibility that the code was incorrect for the labels. However, there were issues with fitting this model and saving/committing the code in CodeAnywhere and as such the code ran cannot be reviewed.

V5:

```
def create_tf_model():
    model = Sequential()

    model.add(Conv2D(filters=16, kernel_size=(5,5),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=32, kernel_size=(4,4),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=64, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Conv2D(filters=32, kernel_size=(3,3),input_shape=image_shape, activation='relu',))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())
    model.add(Dense(64, activation = 'relu'))

    model.add(Dropout(0.3))
    model.add(Dense(1, activation = 'sigmoid'))

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

    return model
```

```
early_stop = EarlyStopping(monitor='val_loss',patience=3)
```






```
evaluation = model.evaluate(test_set)
```

```
27/27 [==============================] - 13s 472ms/step - loss: 1.8114e-05 - accuracy: 1.0000
```

```
0.9999972517478
healthy
```

Model accuracy plot suggests slight overfitting as the loss and accuracy lines have crossed the corresponding val_loss and val_accuracy lines.