In [1]:
```
# first neural network with keras tutorial

from keras.models import Sequential
from keras.layers import Dense
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score
import numpy as np
from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import KNNImputer
from keras.optimizers import SGD
```

```
Using TensorFlow backend.
```

In [2]:
```
# load the dataset
df_empty = np.genfromtxt('framingham.csv', delimiter=",")
print(df_empty.shape)

print(df_empty)
```

```
(4240, 16)
[[  1.  39.   4. ...  80.  77.   0.]
 [  0.  46.   2. ...  95.  76.   0.]
 [  1.  48.   1. ...  75.  70.   0.]
 ...
 [  0.  52.   2. ...  80. 107.   0.]
 [  1.  40.   3. ...  67.  72.   0.]
 [  0.  39.   3. ...  85.  80.   0.]]
```

In [3]:
```
imputer = KNNImputer(n_neighbors=3, weights="uniform")
df = imputer.fit_transform(df_empty)
```

In [4]:
```
# split into input (X) and output (y) variables
X = df[:,0:15]
y = df[:,15]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, r
```

In [5]:
```
# define the keras model
model = Sequential()
model.add(Dense(25, input_dim=15, activation='relu'))
model.add(Dense(15, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

```
WARNING:tensorflow:From /Users/dmitryshribak/.conda/envs/PycharmProjects/
lib/python3.7/site-packages/keras/backend/tensorflow_backend.py:74: The n
ame tf.get_default_graph is deprecated. Please use tf.compat.v1.get_defau
lt_graph instead.

WARNING:tensorflow:From /Users/dmitryshribak/.conda/envs/PycharmProjects/
lib/python3.7/site-packages/keras/backend/tensorflow_backend.py:517: The
name tf.placeholder is deprecated. Please use tf.compat.v1.placeholder in
stead.

WARNING:tensorflow:From /Users/dmitryshribak/.conda/envs/PycharmProjects/
lib/python3.7/site-packages/keras/backend/tensorflow_backend.py:4138: The
name tf.random_uniform is deprecated. Please use tf.random.uniform instea
d.
```

In [6]:
```
# compile the keras model

opt = SGD(lr=0.0001)


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

```
WARNING:tensorflow:From /Users/dmitryshribak/.conda/envs/PycharmProjects/
lib/python3.7/site-packages/keras/optimizers.py:790: The name tf.train.Op
timizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From /Users/dmitryshribak/.conda/envs/PycharmProjects/
lib/python3.7/site-packages/keras/backend/tensorflow_backend.py:3376: The
name tf.log is deprecated. Please use tf.math.log instead.

WARNING:tensorflow:From /Users/dmitryshribak/.conda/envs/PycharmProjects/
lib/python3.7/site-packages/tensorflow_core/python/ops/nn_impl.py:183: wh
ere (from tensorflow.python.ops.array_ops) is deprecated and will be remo
ved in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
```

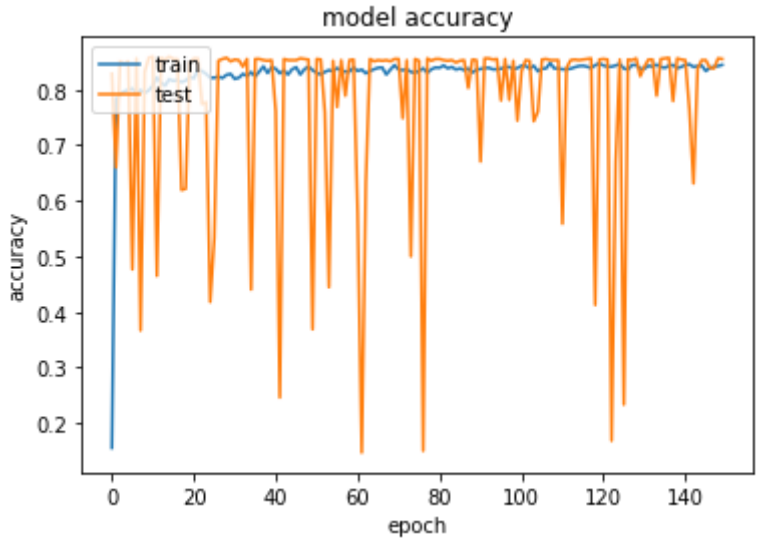
In [7]:
```
# Fit the model
history = model.fit(X_train, y_train, validation_split=0.33, epochs=150, ba
```

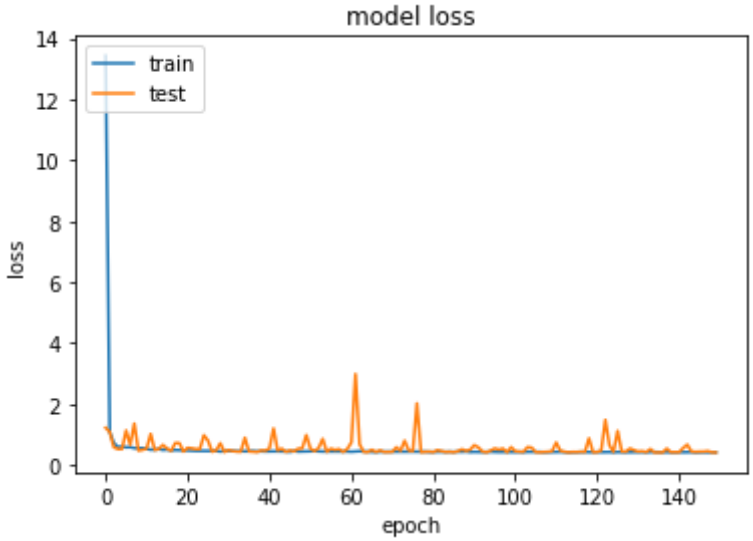
```
- acc: 0.8438 - val_loss: 0.5458 - val_acc: 0.8561
Epoch 130/150
1902/1902 [==============================] - 0s 232us/step - loss: 0.4299
- acc: 0.8375 - val_loss: 0.4912 - val_acc: 0.8241
Epoch 131/150
1902/1902 [==============================] - 0s 233us/step - loss: 0.4262
- acc: 0.8375 - val_loss: 0.4463 - val_acc: 0.8412
Epoch 132/150
1902/1902 [==============================] - 0s 240us/step - loss: 0.4284
- acc: 0.8428 - val_loss: 0.4563 - val_acc: 0.8539
Epoch 133/150
1902/1902 [==============================] - 0s 225us/step - loss: 0.4231
- acc: 0.8412 - val_loss: 0.4267 - val_acc: 0.8539
Epoch 134/150
1902/1902 [==============================] - 0s 207us/step - loss: 0.4282
- acc: 0.8449 - val_loss: 0.5262 - val_acc: 0.7889
Epoch 135/150
1902/1902 [==============================] - 0s 213us/step - loss: 0.4322
- acc: 0.8407 - val_loss: 0.4155 - val_acc: 0.8550
Epoch 136/150
```

In [8]:
```
history_dict = history.history
print(history_dict.keys())
```

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

In [9]:
```
# summarize history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
```





In [10]:
```
y_pred = model.predict_classes(X_test)
print(accuracy_score(y_test, y_pred))
```

```
0.8464285714285714
```

In [ ]:
```

```