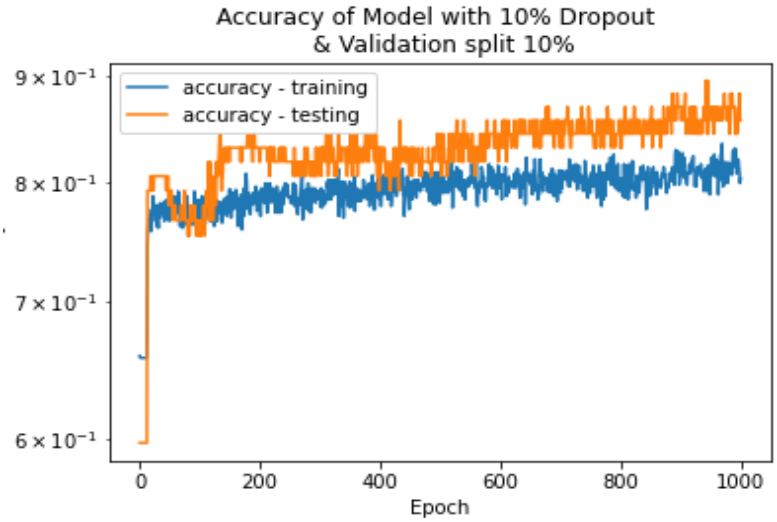
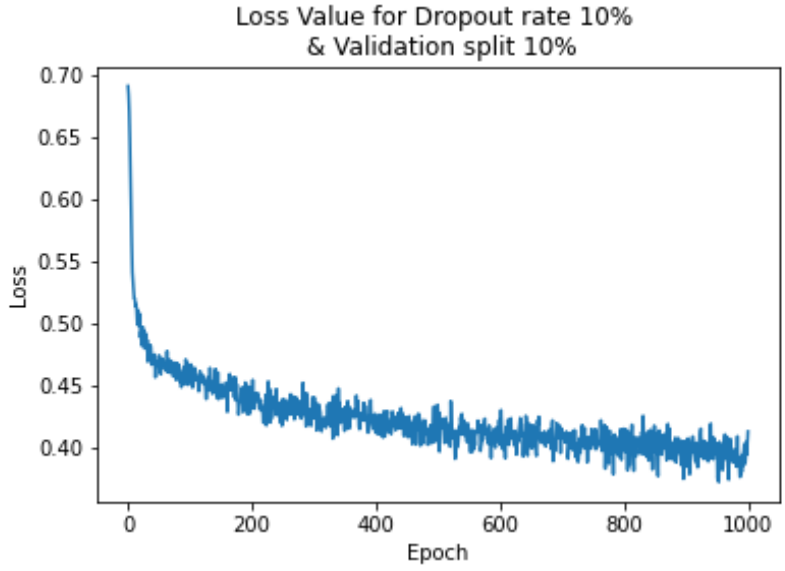
Steven Vasquez
04/25/2020

Dropout Rate/Validation Split Mini Report

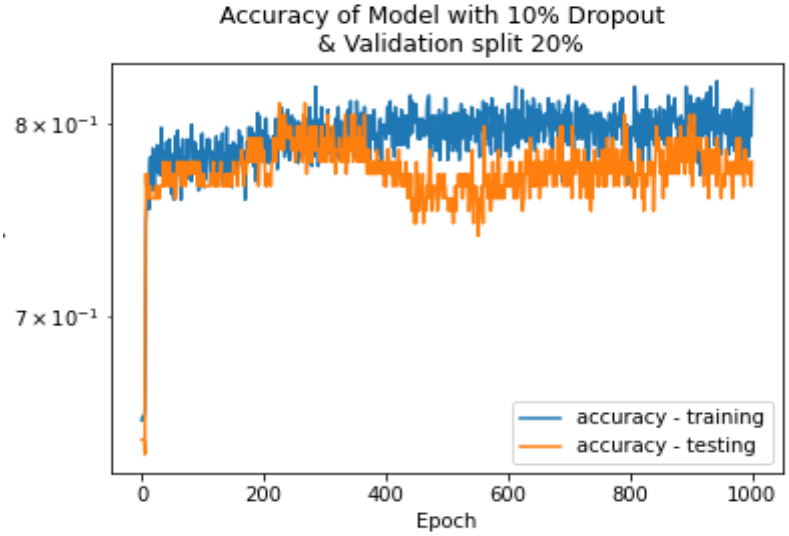
'Dropout' is one way of preventing or accommodating a neural network that is overfitting. Besides 'dropout' adding data to a model can help decrease the overfitting. The dropout technique is used to 'drop' links, weights, and neurons to help simplify the model for its optimal weights and biases. For this example different combination of dropout rate and validation split effect the accuracy and confusion matrix of a binary classification neural network. A separate notebook will be attached to show the code that was used for this demonstration. The main differences in the commands are the dropout rate and validation split. For the first 3 examples the dropout rate will be 10% while the validation split will be 10%, 20%, and 30% in that order.

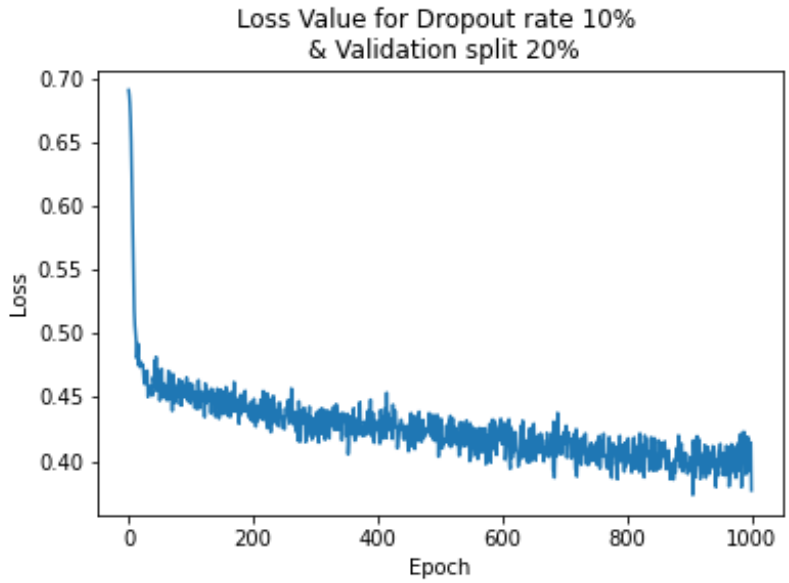
```
768/768 [==============================] - 0s 26us/step

accuracy: 83.72%
====================
Saved model to disk
====================
Confusion Matrix
================
True negatives:  447
False negatives:  72
False positives:  53
True positives:  196
```



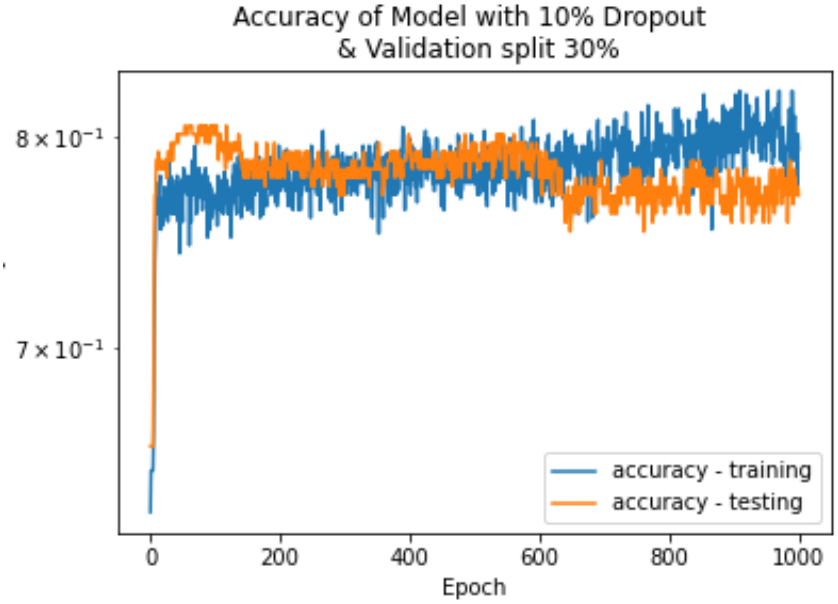


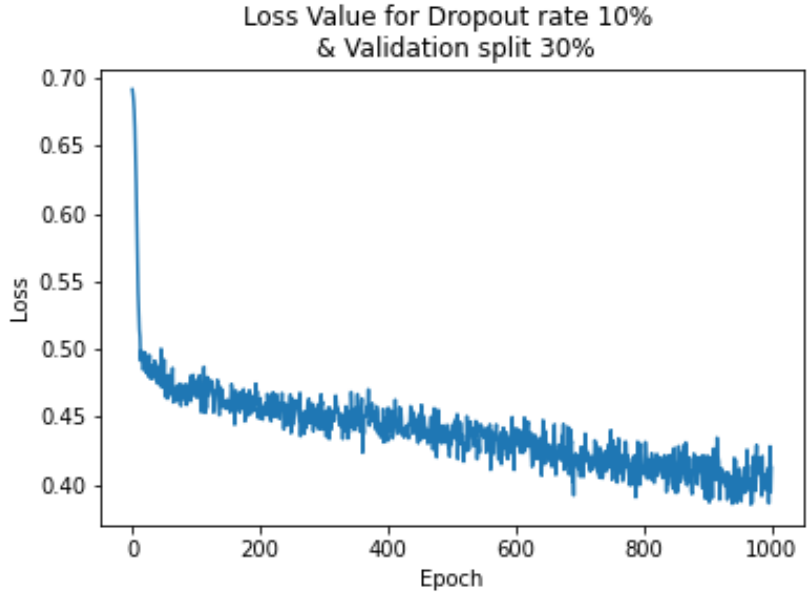
```
768/768 [==============================] - 0s 21us/step

accuracy: 81.77%
====================
Saved model to disk
====================
Confusion Matrix
================
True negatives:  449
False negatives:  89
False positives:  51
True positives:  179
```





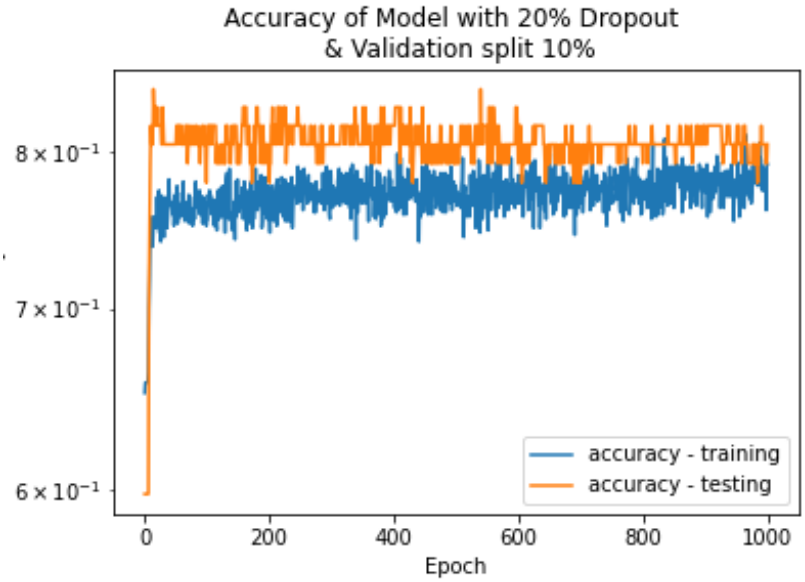
```
768/768 [==============================] - 0s 22us/step

accuracy: 81.12%
====================
Saved model to disk
====================
Confusion Matrix
================
True negatives:  434
False negatives:  79
False positives:  66
True positives:  189
```

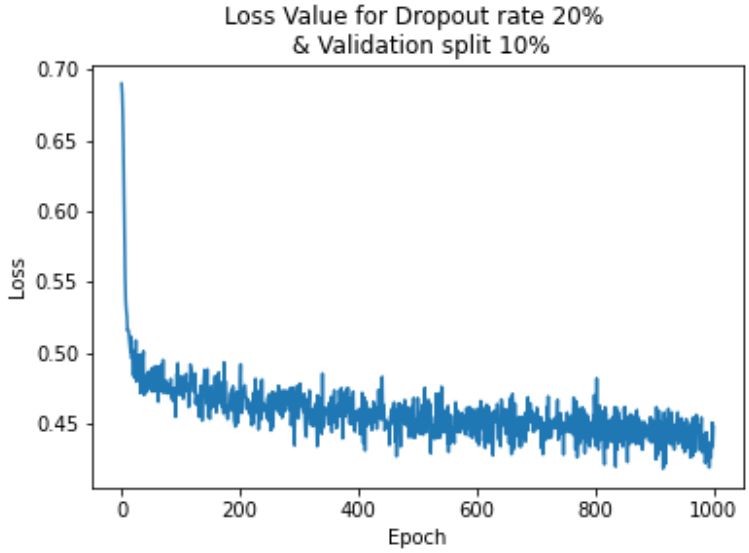




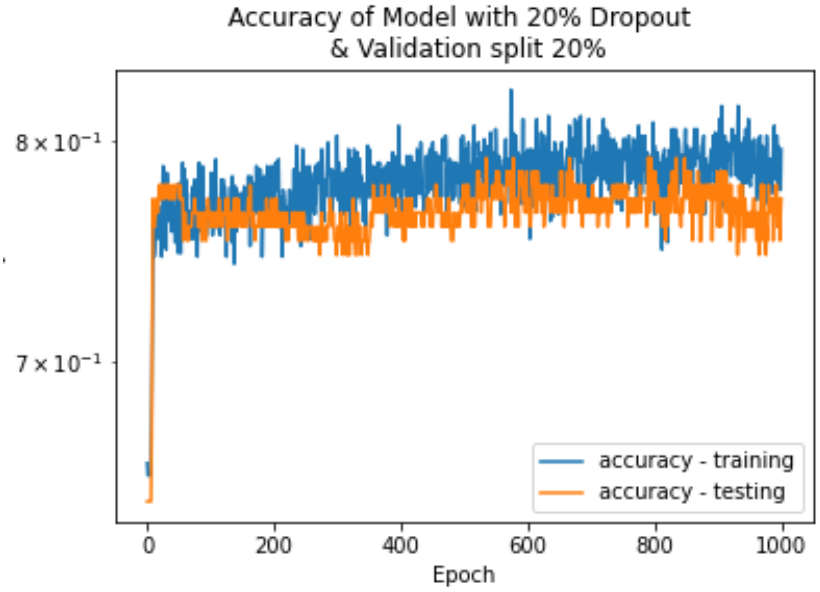
The next 2 demonstrations of how these different approaches of reducing overfitting can effect a categorical neural network.

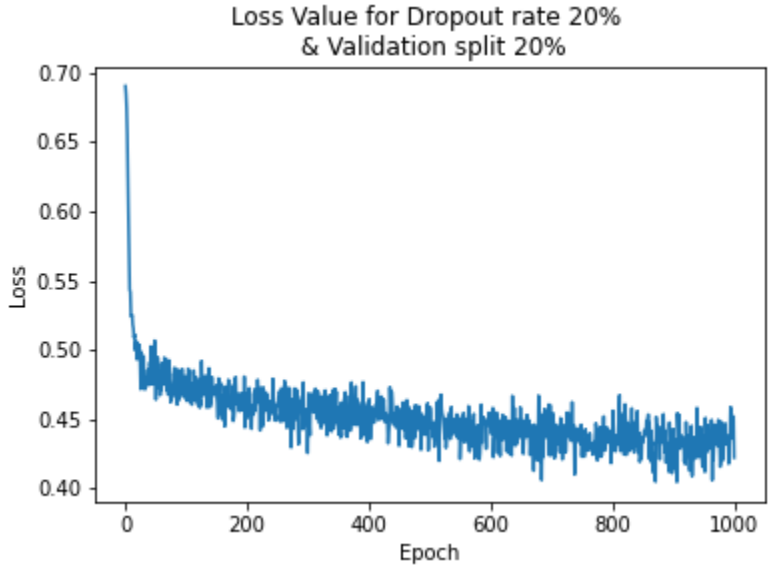
```
768/768 [==============================] - 0s 21us/step

accuracy: 80.08%
==================
Saved model to disk
==================
Confusion Matrix
================
True negatives:  440
False negatives:  93
False positives:  60
True positives:  175
```



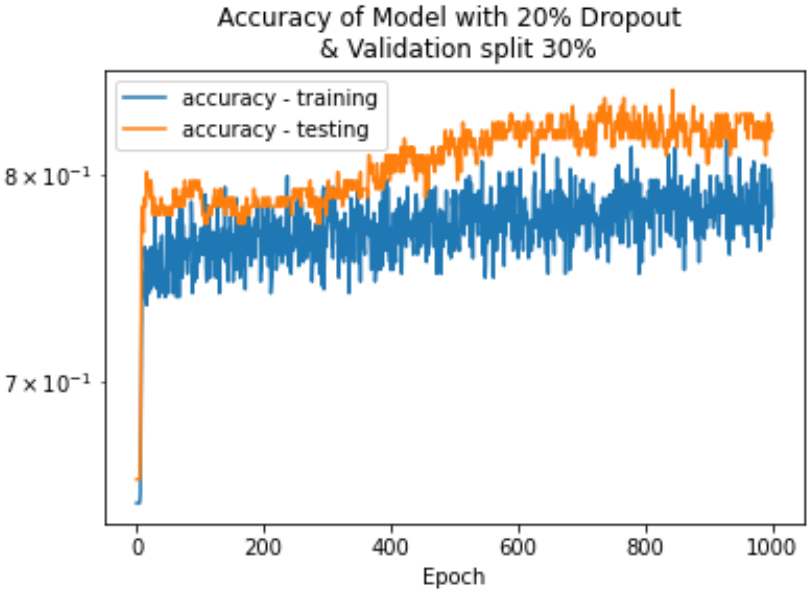


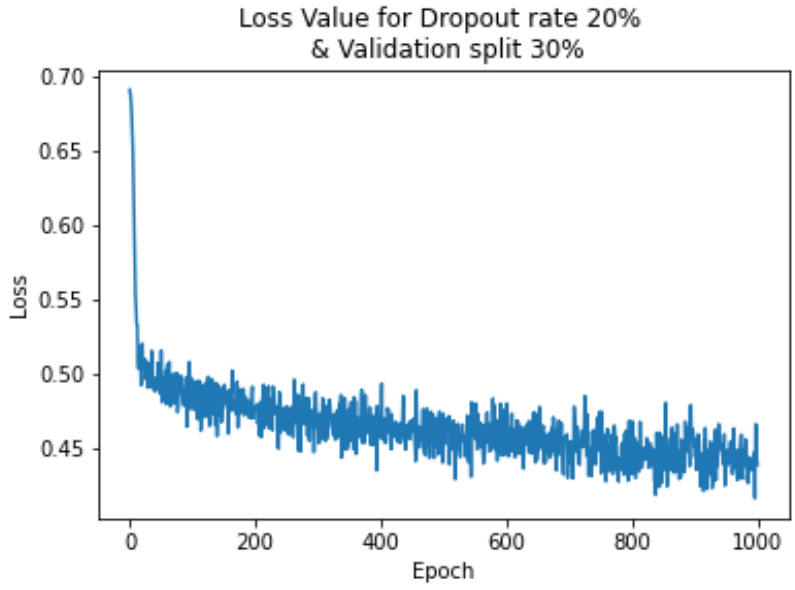
```
768/768 [==============================] - 0s 23us/step

accuracy: 82.16%
==================
Saved model to disk
==================
Confusion Matrix
================
True negatives:  457
False negatives:  94
False positives:  43
True positives:  174
```





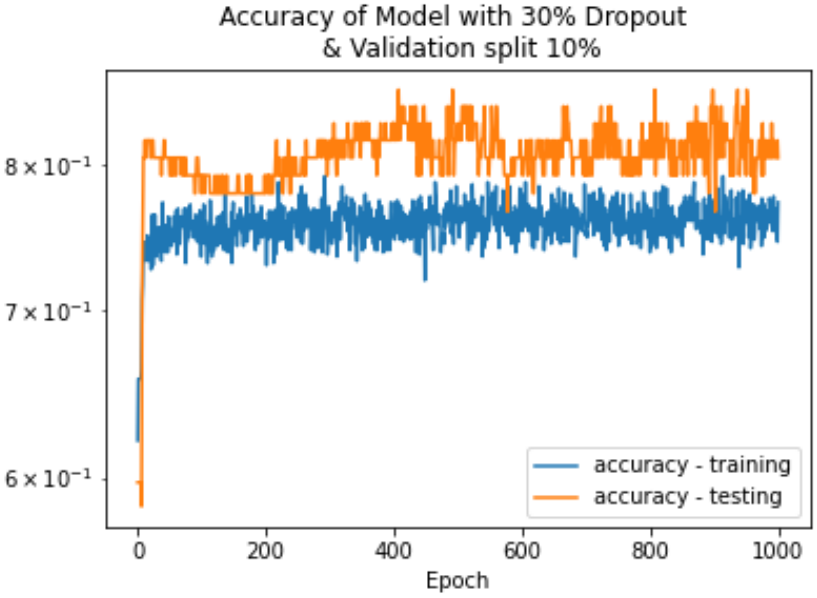
```
768/768 [==============================] - 0s 23us/step

accuracy: 81.64%
====================
Saved model to disk
====================
Confusion Matrix
================
True negatives:  449
False negatives:  90
False positives:  51
True positives:  178
```

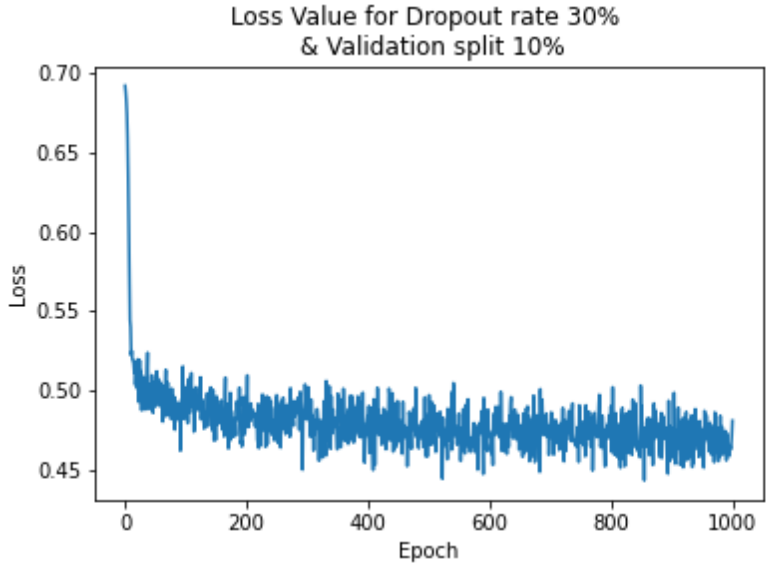




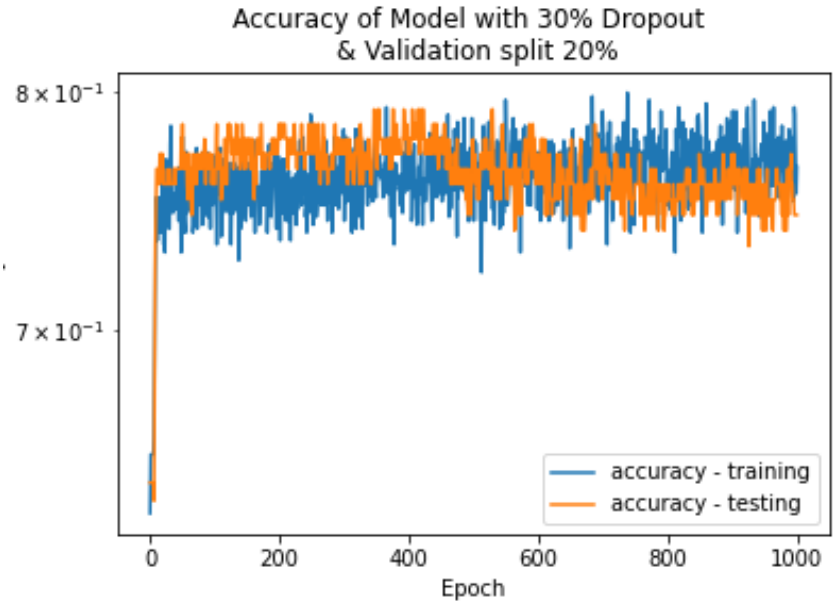
The final example will demonstrate the same ANN with Dropout rate at 30% and validation split as 10%, 20%, 30% respectfully.

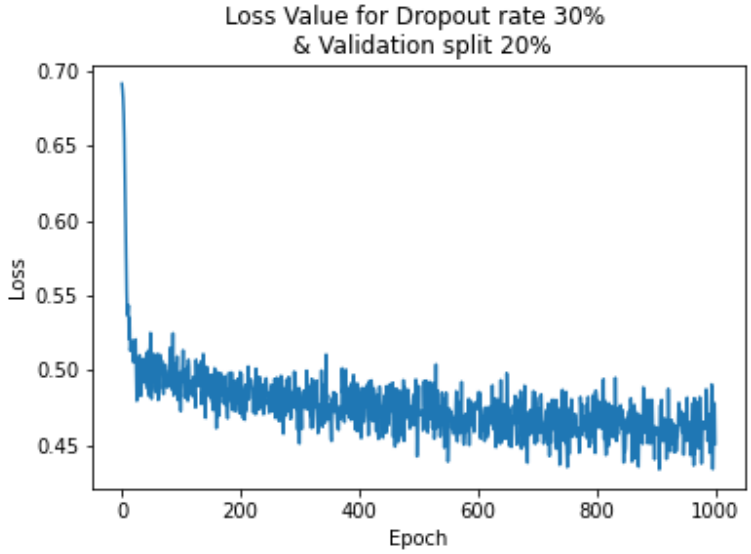
```
768/768 [==============================] - 0s 23us/step

accuracy: 80.73%
====================
Saved model to disk
====================
Confusion Matrix
================
True negatives:  455
False negatives:  103
False positives:  45
True positives:  165
```



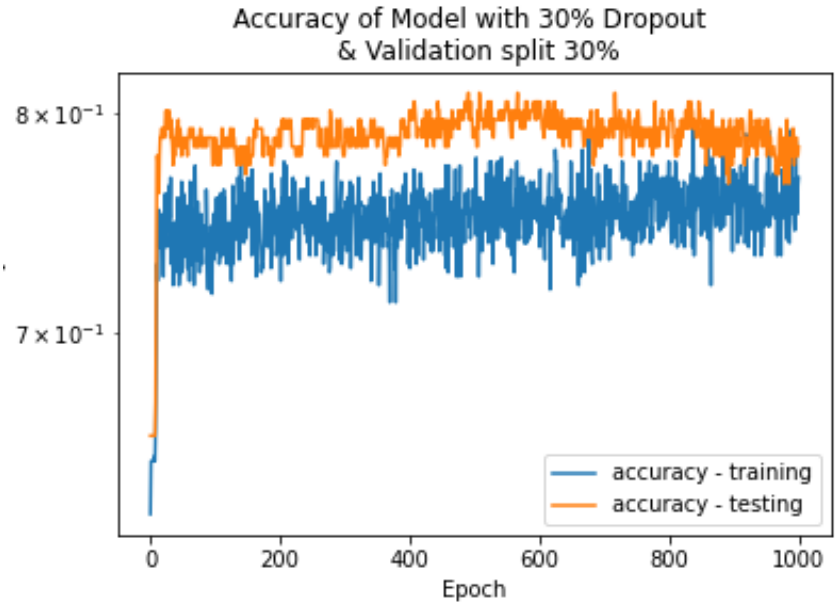


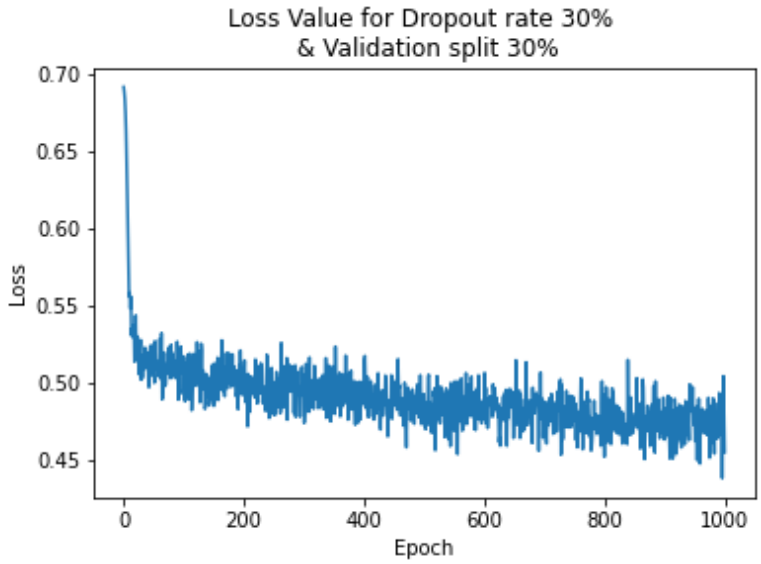
```
768/768 [==============================] - 0s 22us/step

accuracy: 80.86%
==================
Saved model to disk
==================
Confusion Matrix
================
True negatives:  456
False negatives:  103
False positives:  44
True positives:  165
```





```
768/768 [==============================] - 0s 26us/step

accuracy: 78.91%
====================
Saved model to disk
====================
Confusion Matrix
================
True negatives:  439
False negatives:  101
False positives:  61
True positives:  167
```





The network with the highest accuracy is the network with 10% drop out rate and 10% validation split. That took me as a surprise because I expected a need to drop more weights, but the networks with 30% dropout rate didn't do so well. There are many other factors that could be effecting the networks accuracy. Such as not enough data, the 'best' weights weren't used since there was no checkpoint or early stoppage included in the code, or features need to be normalized properly.