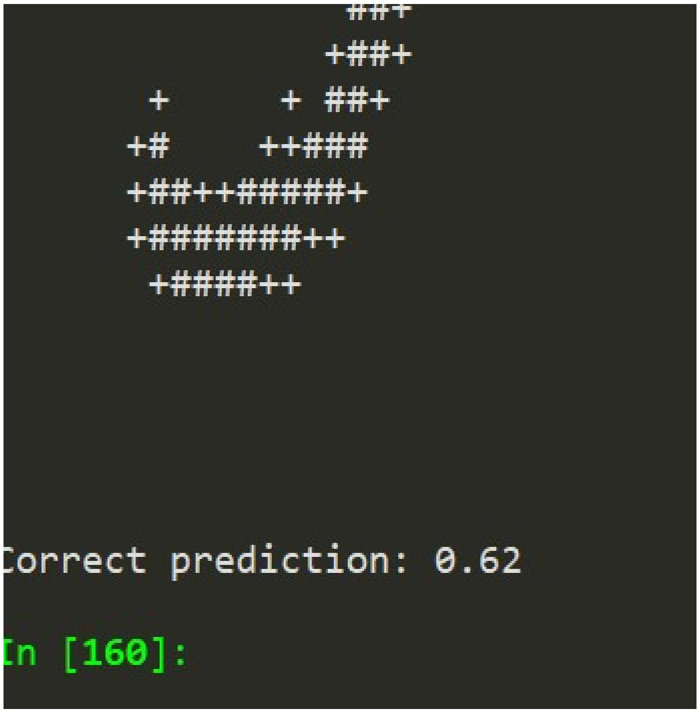
Saa221 / Jw184 MP 4 Report

a.2.: The drawback to using the regular probability product versus the logarithmic formula is the potential for arithmetic underflow which will undermine computation (in some instances, underflow still affects our program)

a.5.: Attached are some images from the actual log and filled in values for brevity:

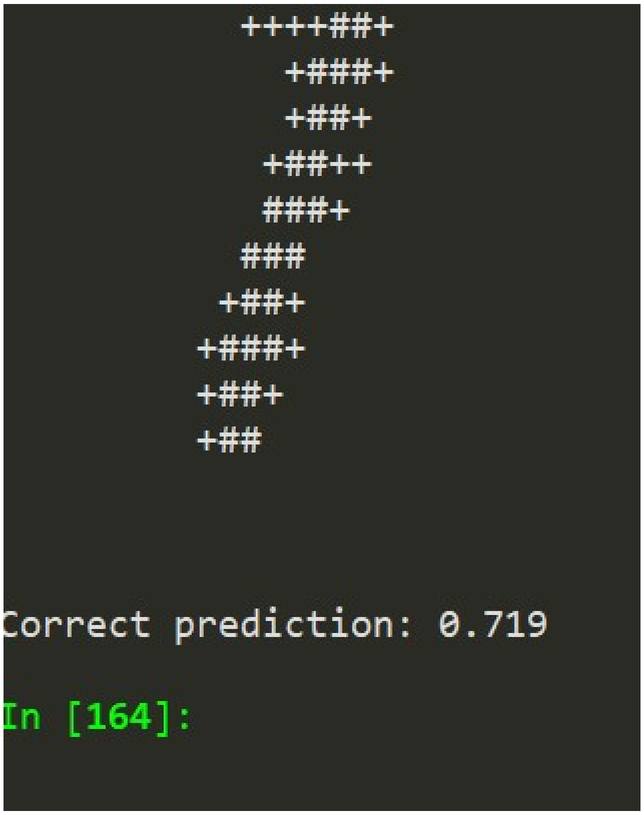


```
          ##+
         +##+
 +     + ##+
+#    ++###
+##++#####+
+#######++
 +####++




orrect prediction: 0.62

n [160]:
```

20% accuracy: 0.699

30% accuracy: 0.709

40% accuracy: 0.705



```
           ++++##+
             +####+
             +##+
            +##++
            ###+
           ###
          +##+
         +####+
         +##+
         +##



Correct prediction: 0.719

In [164]:
```

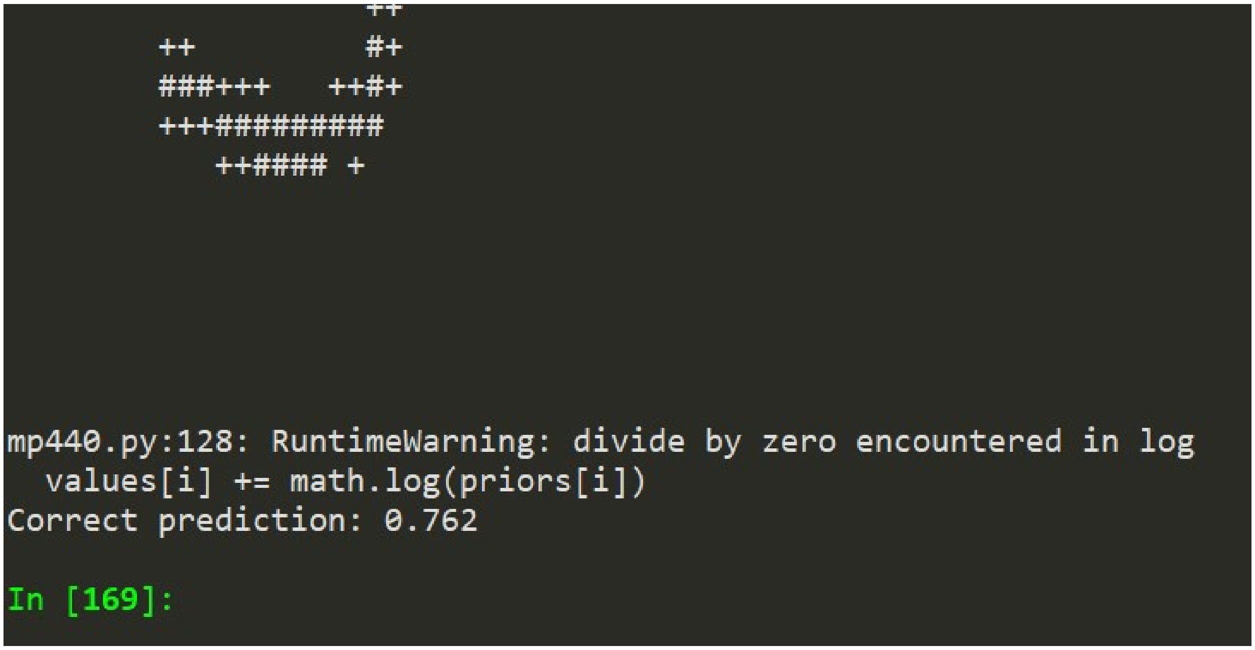
60% accuracy: 0.724

70% accuracy: 0.739

80% accuracy: 0.747

90% accuracy: 0.748

And finally,



```
                   ++
        ++         #+
        ###+++   ++#+
        +++#########
           ++#### +




mp440.py:128: RuntimeWarning: divide by zero encountered in log
  values[i] += math.log(priors[i])
Correct prediction: 0.762

In [169]:
```

a.6.:

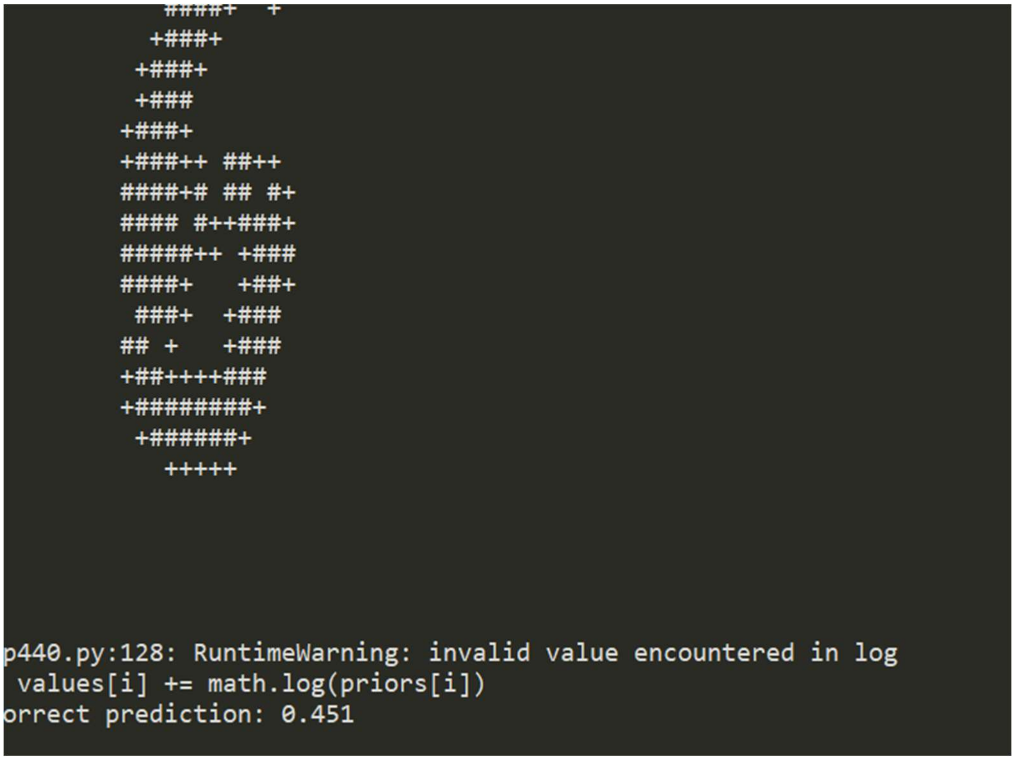
We chose the k-value of 50, implementations testing from 5 to 500 and even under 1 were tested, but for some reason the best yield was at 50.

The classifier approaches the 75% threshold on the basic implementation as required by the assignment. However, in real-life applications this type of accuracy leaves much to be desired. Decision trees on sci-kit learn can hit 80+ accuracy with no special training.

b.1.:

The advanced method used a combination of odds-ratio defined features, favoring the outer + digits, and then favoring the inner – digits, ultimately results were very poor.

The yields started at 0.32 and incrementally (almost proportionally) climbed toward the best yield being at 100% test set with:



```
           ####+  +
          +####+
         +####+
         +###
        +####+
        +###++ ##++
        ####+# ## #+
        #### #++###+
        #####++ +###
        ####+   +##+
         ###+  +###
        ## +   +###
        +##++++###
        +########+
         +######+
           +++++




p440.py:128: RuntimeWarning: invalid value encountered in log
 values[i] += math.log(priors[i])
orrect prediction: 0.451
```

b.2:

continued in final section

b.3.:

Advanced implementations using odds-ratios, or favoring the edges or middle values did not yield better results than the basic implementation, in fact they were much worse. Therefore, the final implantation uses the basic_feature_extract() method to finalize a yield of 76.2% accuracy when trained with the entire training set and 75% above 70%