In [1]:
```
from scipy.stats import binom
import numpy as np

# ensembles
ensembles = [0,1,2,3,4]

### Ensemble 0 is a test case from
# https://math.stackexchange.com/questions/1712689/to-improve-the-accuracy-a-majority-vote-is-taken-what-is-the-probability-of-an

### Ensemble 4 is a test case from
# http://courses.washington.edu/css490/2012.Winter/lecture_slides/12_ensemble_methods_1_r1.pdf
# Slide 13 indicates probability of a success, since this process is a failure, it should be 1 - 0.97 for about 0.03

# Total models in each ensemble
num_models = [17,11,11,21,21]

# Error rates
model_error_rates = [.2,.2,.49,.49,.3]

# Majority - 1 to produce a failure
minorities = [8,5,5,10,10]

for e,n,p,m in zip(ensembles,num_models,model_error_rates,minorities):
    total_p = 0

    # Need to add one to our minority iterator because python is not inclusive
    for k in range(0, m + 1):
        numerator = np.math.factorial(n)
        denominator = (np.math.factorial(n-k)) * np.math.factorial(k)

        curprob = (numerator/denominator)*(p**k)*(1-p)**(n-k)
        #print(f'Mine: {curprob}')
        #print(f'SciPy: {binom.pmf(k,n,p)}')
        total_p += curprob

    print(f'For ensemble {e}: {1-total_p}')
```

```
For ensemble 0: 0.002581462836837356
For ensemble 1: 0.011654205439999288
For ensemble 2: 0.47294772571497457
For ensemble 3: 0.46304790101273563
For ensemble 4: 0.026389940701285197
```

In [ ]:
```
```