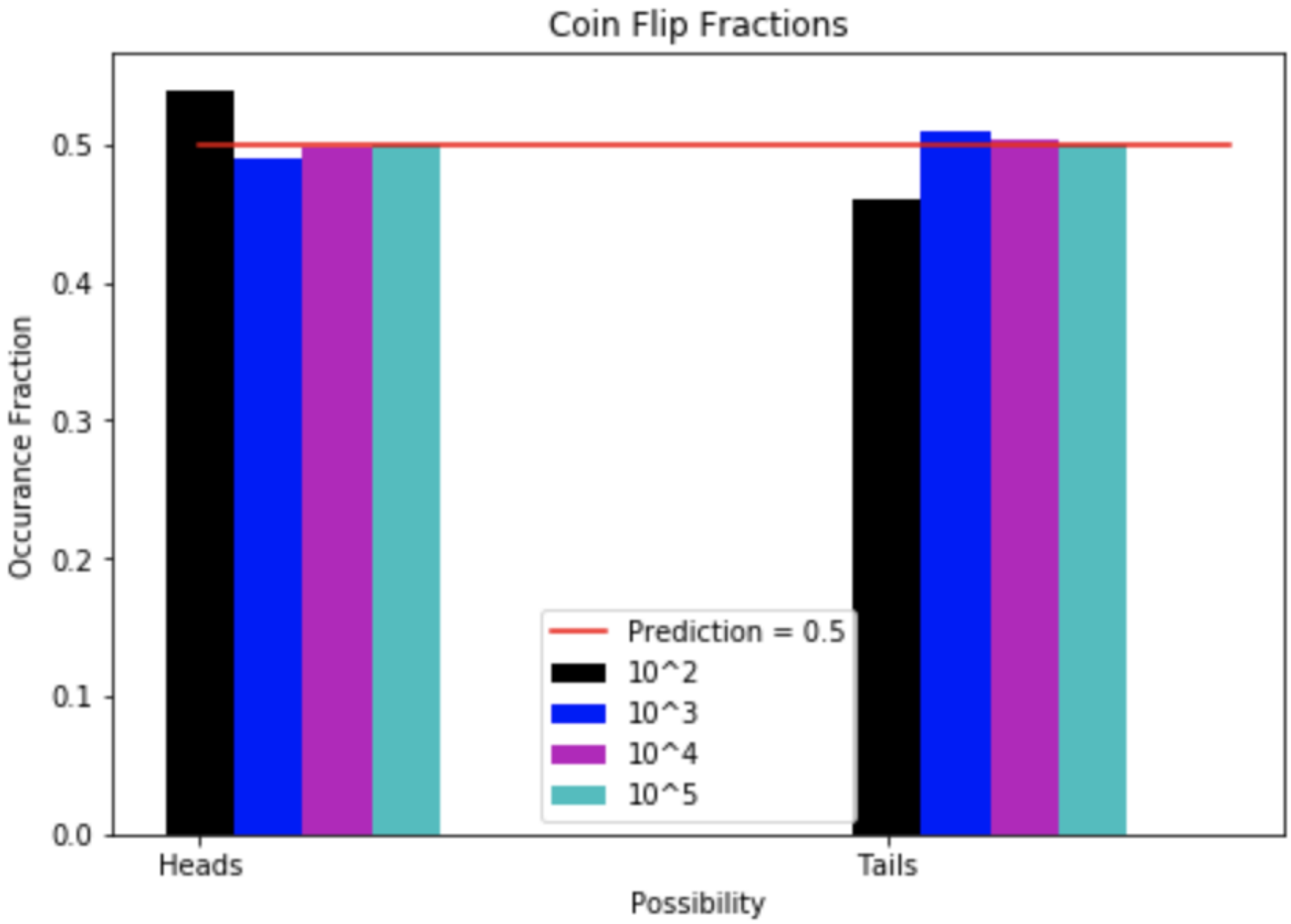
Introduction

In this experiement I test the idea of the Law of Large Numbers. This is the idea that as the amount of data increases, the occurance fractions for the possibilities will converge to the prediction of those possibilities.

Data

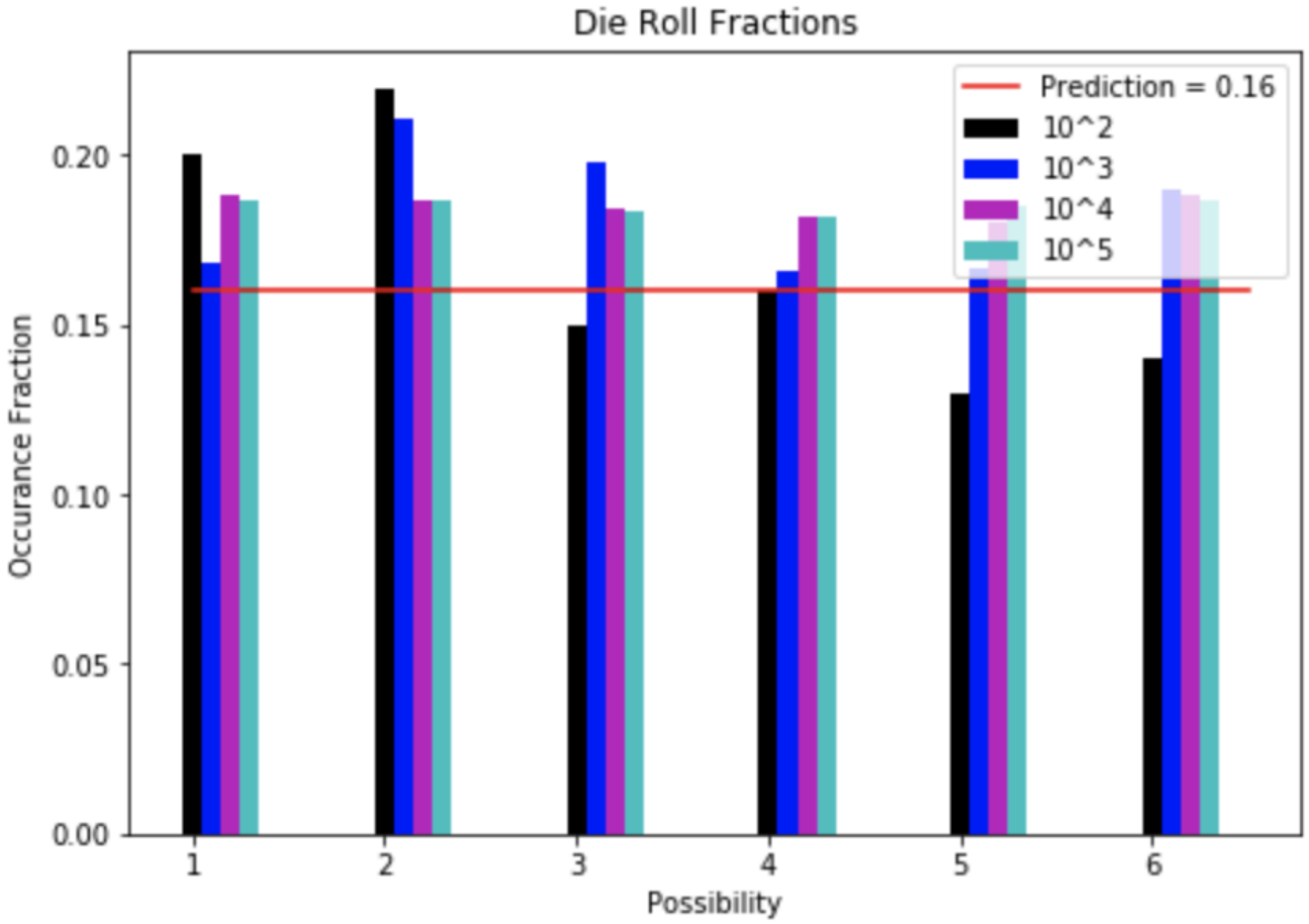
Tossing a coin

First I toss a coin. I simulate this by creating a python code to output heads or tails at random. The prediction value is 1/2 as there are 2 possible outcomes.



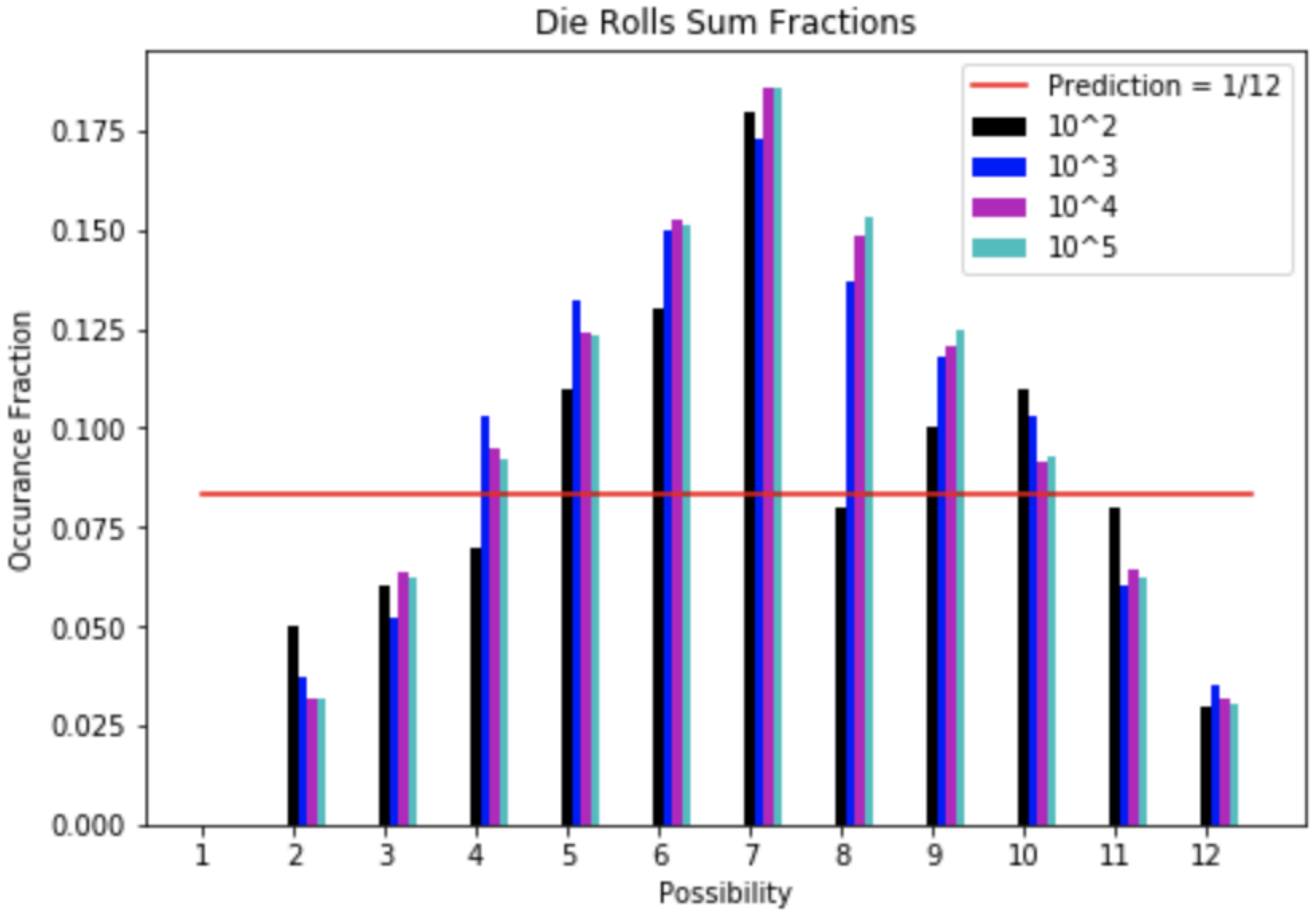
Rolling a die

Next I roll a die using the same method of simulation. The prediction value is 1/6 as there are 6 possible outcomes.



Rolling two dice

Finally, I roll two dice and add together their sums for each roll and simulate it the same as the last two. The prediction value is 1/12 as there are 12 possible outcomes.



Conclusion

In this experiment I have found that the idea of the Law of Large Numbers can be proved as the occurance fractions converge more toward the predicted outcome as more data is added.

Appendix

The simulation results are produced with the following code.

Flipping a coin

In [625]:
```
# Necessary coding packages

# Numerical python
import numpy as np

# Plotting
import matplotlib.pyplot as plt

# Random outcome functions
import random
```

In [626]:
```
# Function to determine outcome of coin flip
# n is number of flips
def coin_flip(n):

    output = []

    # Loop to repeat coin flip
    for amount in range(n):
        flip = random.randint(0, 1)

        if(flip == 0):
            output.append(1)

    return(sum(output)/n)
```

In [627]:
```
x = coin_flip(100)
tails_x = [x, 1-x]
y = coin_flip(10*10*10)
tails_y = [y, 1-y]
z = coin_flip(10*10*10*10)
tails_z = [z, 1-z]
a = coin_flip(10*10*10*10*10)
tails_a = [a, 1-a]
```

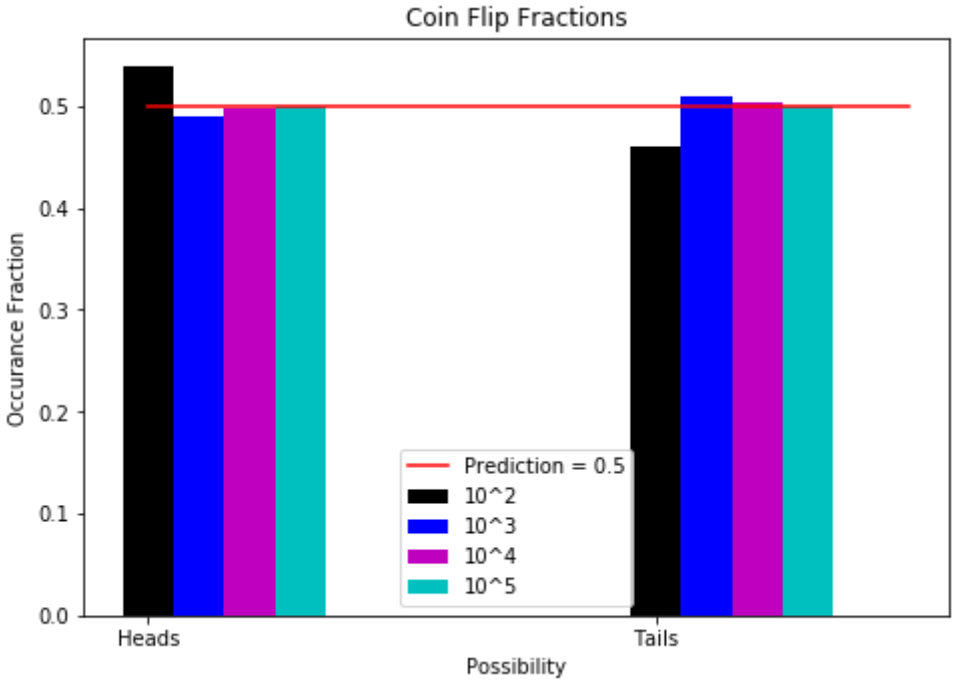
In [628]:
```
X = np.arange(2)
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.bar(X + 0.0, tails_x, color = 'k', width = 0.1, label = "10^2")
ax.bar(X + 0.1, tails_y, color = 'b', width = 0.1, label = "10^3")
ax.bar(X + 0.2, tails_z, color = 'm', width = 0.1, label = "10^4")
ax.bar(X + 0.3, tails_a, color = 'c', width = 0.1, label = "10^5")

x_coordinates = [0, 1.5]
y_coordinates = [.5, .5]

plt.plot(x_coordinates, y_coordinates, color = 'r', label = "Prediction
 = 0.5")

labels = ['Heads', 'Tails']
x = np.arange(len(labels))
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.legend()
ax.set_xlabel('Possibility')
ax.set_ylabel('Occurance Fraction')
ax.set_title('Coin Flip Fractions')
```

Out[628]: Text(0.5, 1.0, 'Coin Flip Fractions')



Rolling a die

In [629]:
```
# Roll a die 100 times
```

In [649]:
```
rolled = []

def die():
    outcome = random.randrange(1,7)
    return outcome


def die_roll(n):

    times_rolled = 0;

    for i in range(1,n + 1):
        number = die()
        rolled.append(number)
        times_rolled+=1
    count = [rolled.count(1),rolled.count(2),rolled.count(3),rolled.coun
t(4),rolled.count(5),rolled.count(6)]

    for i in range(0,6):
        fraction = []
        fraction = ((count[i] / n))
        print(fraction)
```

In [650]:
```
die_roll(100)
print(" ")
die_roll(10*10*10)
print(" ")
die_roll(10*10*10*10)
print(" ")
die_roll(10*10*10*10*10)
```

```
0.21
0.21
0.13
0.1
0.18
0.17

0.19
0.192
0.172
0.174
0.174
0.198

0.183
0.1926
0.1797
0.1845
0.1844
0.1858

0.18372
0.18703
0.18455
0.18401
0.18441
0.18728
```

In [651]:
```
x1 = [0.2, 0.22, 0.15, 0.16, 0.13, 0.14]

y1 = [0.168, 0.211, 0.198, 0.166, 0.167, 0.19]

z1 = [0.1885, 0.1867, 0.1844, 0.1817, 0.1806, 0.1881]

a1 = [0.18634, 0.18699, 0.1837, 0.18204, 0.18508, 0.18685]
```

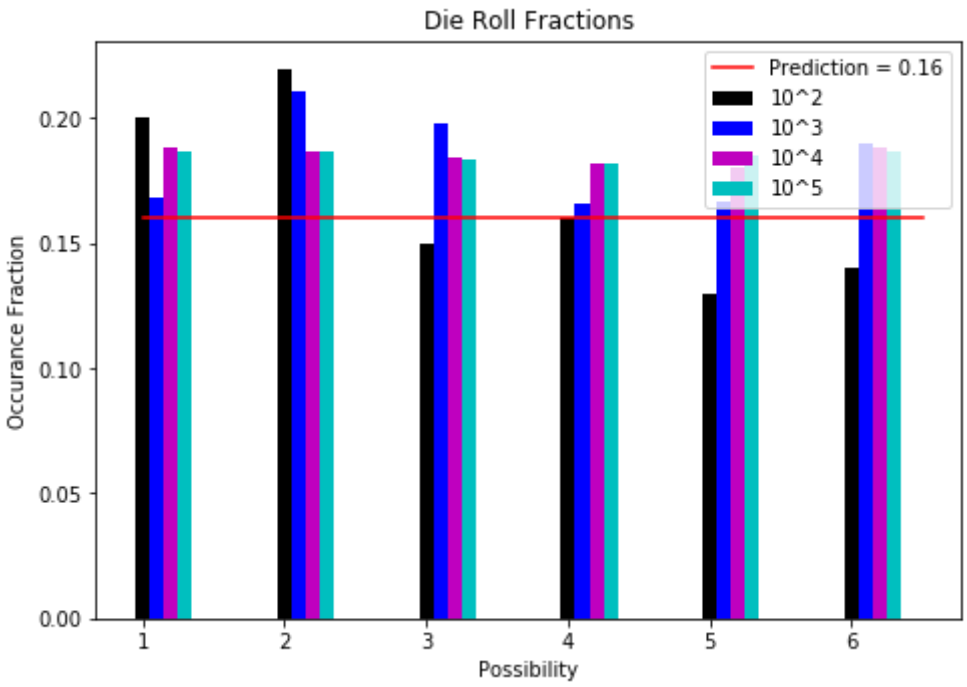
In [652]:
```
X = np.arange(6)
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.bar(X + 0.0, x1, color = 'k', width = 0.1, label = "10^2")
ax.bar(X + 0.1, y1, color = 'b', width = 0.1, label = "10^3")
ax.bar(X + 0.2, z1, color = 'm', width = 0.1, label = "10^4")
ax.bar(X + 0.3, a1, color = 'c', width = 0.1, label = "10^5")

x_coordinates = [0, 5.5]
y_coordinates = [.16, .16]

plt.plot(x_coordinates, y_coordinates, color = 'r', label = "Prediction
 = 0.16")

labels = ['1', '2', '3', '4', '5', '6']
x = np.arange(len(labels))
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.legend()
ax.set_xlabel('Possibility')
ax.set_ylabel('Occurance Fraction')
ax.set_title('Die Roll Fractions')
```

Out[652]: Text(0.5, 1.0, 'Die Roll Fractions')



Rolling two dice and adding the sums

In [653]:
```
# Roll two dice 100 times and add the sum
```

In [654]:
```
# Function to determine outcome of die rolls added together
# n is number of rolls


rolled = []

def die():
    outcome = random.randrange(1,7) + random.randrange(1,7)
    return outcome
def die_roll(n):

    times_rolled = 0;

    for i in range(1,n + 1):
        number = die()
        rolled.append(number)
        times_rolled+=1
    count = [rolled.count(1),rolled.count(2),rolled.count(3),rolled.coun
t(4),rolled.count(5),rolled.count(6),rolled.count(7),rolled.count(8),rol
led.count(9),rolled.count(10),rolled.count(11),rolled.count(12)]

    for i in range(0,12):
        fraction = []
        fraction = ((count[i] / n))

        print(fraction)
```

In [655]:
```
die_roll(100)
print(" ")
die_roll(10*10*10)
print(" ")
die_roll(10*10*10*10)
print(" ")
die_roll(10*10*10*10*10)
```

```
0.0
0.03
0.06
0.04
0.14
0.13
0.18
0.17
0.09
0.06
0.07
0.03

0.0
0.024
0.056
0.097
0.119
0.137
0.196
0.16
0.113
0.09
0.068
0.04

0.0
0.0281
0.0607
0.0892
0.1271
0.1537
0.1945
0.1504
0.124
0.0883
0.0611
0.0329

0.0
0.03016
0.06276
0.09116
0.12494
0.15459
0.1862
0.15204
0.12374
0.09268
0.06141
0.03132
```

In [656]:
```
x2 = [0.0, 0.05, 0.06, 0.07, 0.11, 0.13, 0.18, 0.08, 0.1, 0.11, 0.08, 0.
03]

y2 = [0.0, 0.037, 0.052, 0.103, 0.132, 0.15, 0.173, 0.137, 0.118, 0.103,
0.06, 0.035]

z2 = [0.0, 0.0321, 0.0634, 0.0949, 0.1243, 0.1526, 0.1858, 0.1485, 0.120
8, 0.0918, 0.0641, 0.0317]

a2 = [0.0, 0.03198, 0.06206, 0.09237, 0.12358, 0.15145, 0.18618, 0.15328
, 0.1246, 0.09281, 0.06245, 0.03024]
```

In [657]:
```
X = np.arange(12)
fig = plt.figure()
ax = fig.add_axes([0,0,1,1])
ax.bar(X + 0.0, x2, color = 'k', width = 0.1, label = "10^2")
ax.bar(X + 0.1, y2, color = 'b', width = 0.1, label = "10^3")
ax.bar(X + 0.2, z2, color = 'm', width = 0.1, label = "10^4")
ax.bar(X + 0.3, a2, color = 'c', width = 0.1, label = "10^5")

x_coordinates = [0, 11.5]
y_coordinates = [1/12, 1/12]

plt.plot(x_coordinates, y_coordinates, color = 'r', label = "Prediction
 = 1/12")

labels = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12']
x = np.arange(len(labels))
ax.set_xticks(x)
ax.set_xticklabels(labels)
ax.legend()
ax.set_xlabel('Possibility')
ax.set_ylabel('Occurance Fraction')
ax.set_title('Die Rolls Sum Fractions')
```

Out[657]:
```
Text(0.5, 1.0, 'Die Rolls Sum Fractions')
```