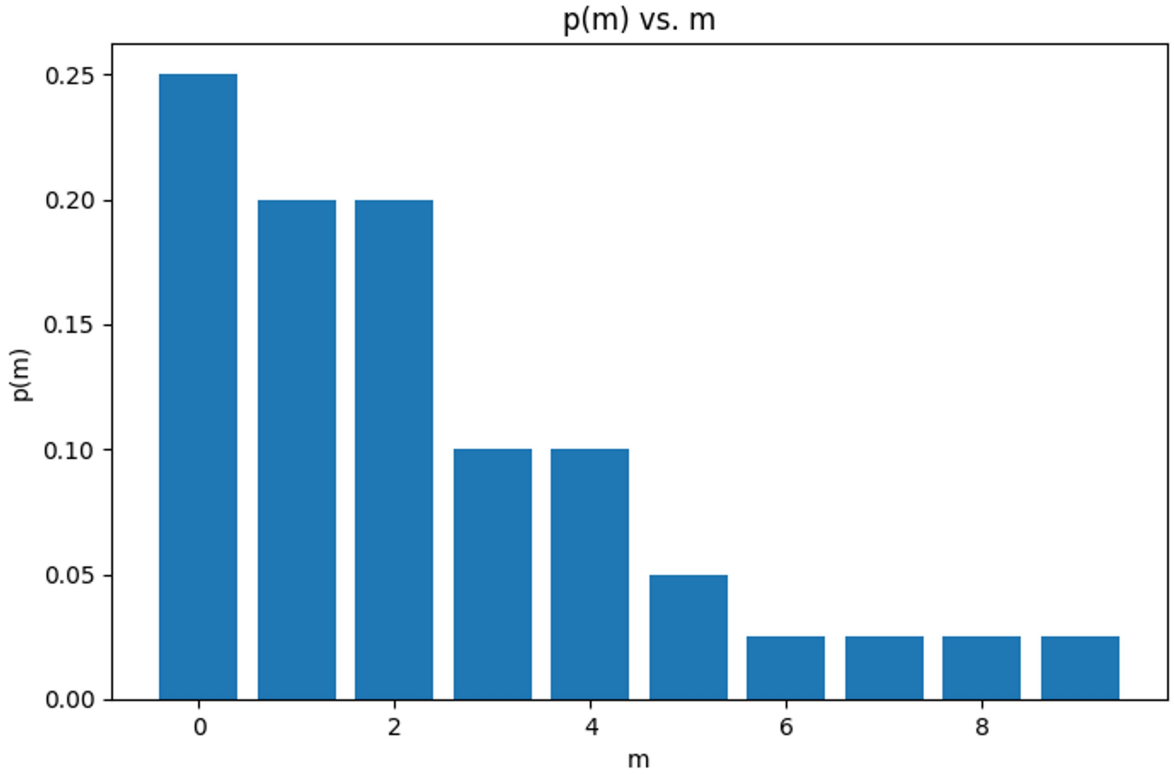
1.1

```python
def prior(m):
    dic = {
        0: 0.25,
        1: 0.2,
        2: 0.2,
        3: 0.1,
        4: 0.1,
        5: 0.05,
        6: 0.025,
        7: 0.025,
        8: 0.025,
        9: 0.025,
    }
    return dic[m]
```

2.1

```
m 0 prior(m) 0.25
m 1 prior(m) 0.2
m 2 prior(m) 0.2
m 3 prior(m) 0.1
m 4 prior(m) 0.1
m 5 prior(m) 0.05
m 6 prior(m) 0.025
m 7 prior(m) 0.025
m 8 prior(m) 0.025
m 9 prior(m) 0.025
```



3.1

```python
def likelihood_single(x,y,m):
    if m > 0:
        f_mx = x**m
    else:
        f_mx = 0
    return sp.stats.norm.pdf(y, f_mx, 0.1**2)
```

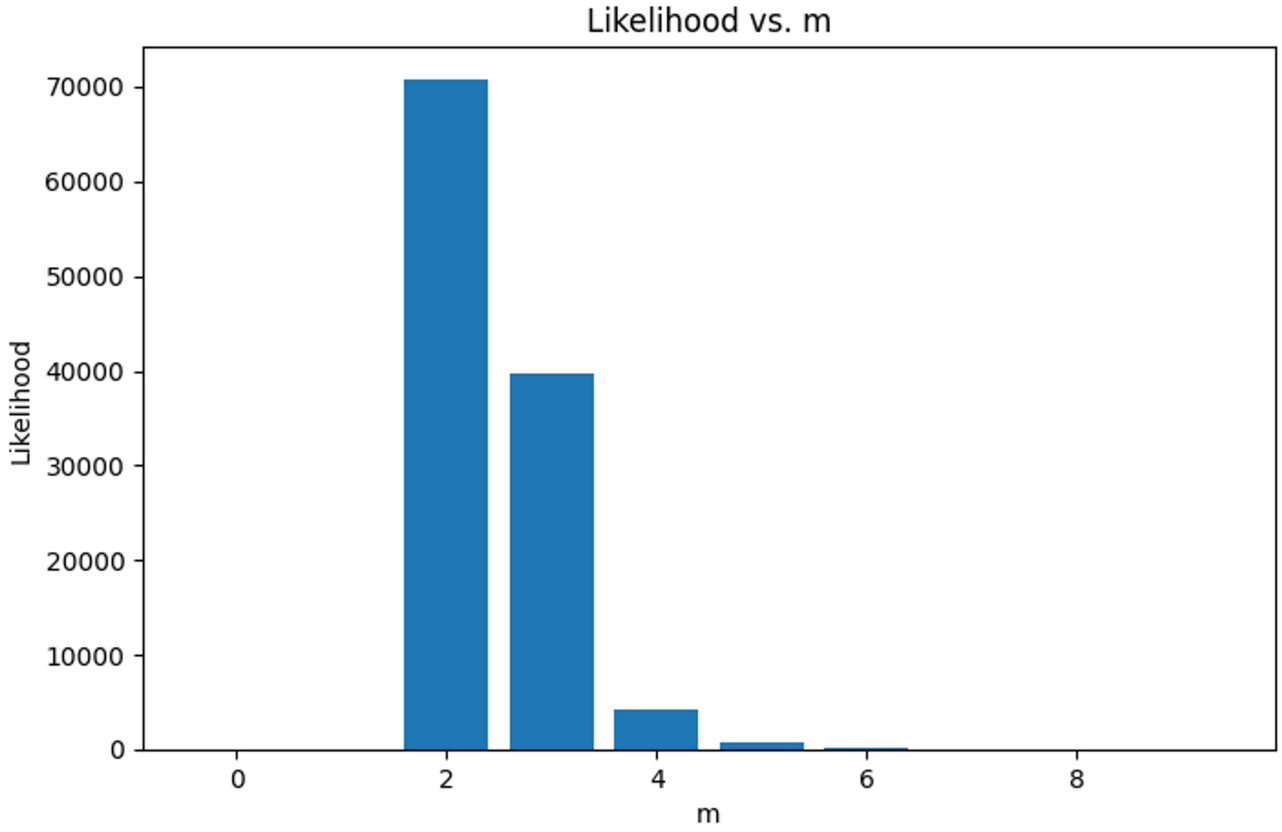
4.1

Thursday, December 1, 2022 4:40 PM

```python
def likelihood(X,Y,m):
    p = likelihood_single(X[0], Y[0], m)
    for n in range(1, len(X)):
        p = p * likelihood_single(X[n], Y[n], m)
    return p
```

5.1

Thursday, December 1, 2022 5:51 PM



6.1

Thursday, December 1, 2022 5:55 PM

$$P(M=m \mid Data) = \frac{1}{P(Data)} \times P(M=m) \times P(Data \mid M=m)$$

$$\sum_{m}^{M} P(M=m \mid Data) = 1 \qquad m = \{0, 1, 2, \ldots, m\}$$

Let $\frac{1}{P(Data)} = c$, $X_m = P(M=m) \times P(Data \mid M=m)$

So, $P(M=m \mid Data) = c \times X_m$

$$\therefore \quad 1 = \sum_{m}^{M} c * X_m \quad \text{or} \quad 1 = c * \sum_{m}^{M} X_m$$

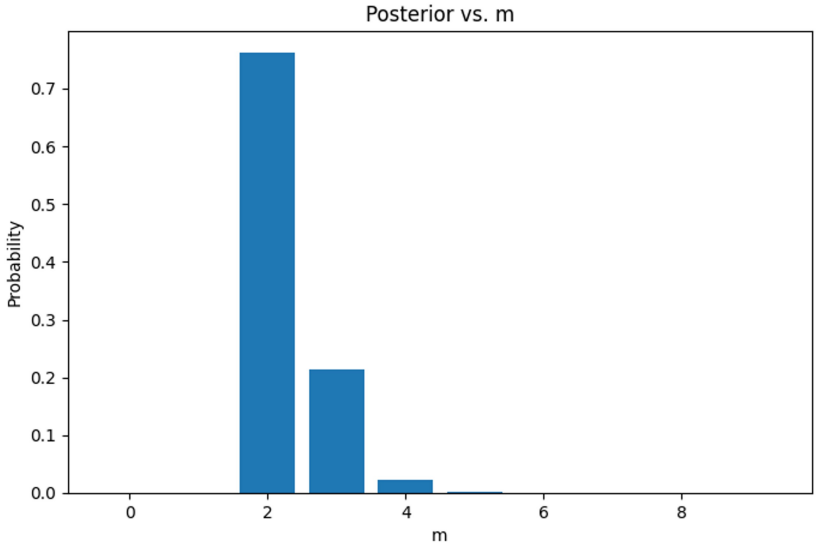
$$\frac{1}{c} = \sum_{m}^{M} X_m \quad \text{or} \quad \frac{1}{P(Data)} = \sum_{m}^{M} P(M=m) \times P(Data \mid M=m)$$

$$\therefore \quad P(M=m \mid Data) = \frac{P(M=m) \times P(Data \mid M=m)}{\sum_{x}^{M} P(Data \mid M=x) \cdot P(M=x)}$$

7.1

```
def posterior(X,Y,m):
  p_data = 0
  for i in range(10):
    p_data += likelihood(X, Y, i) * prior(i)
  return (prior(m)*likelihood(X, Y, m))/p_data
```

8.1



9.1

```
def MAP(X,Y):
  ap = [posterior(X, Y, i) for i in range(10)]
  return ap.index(max(ap))
```

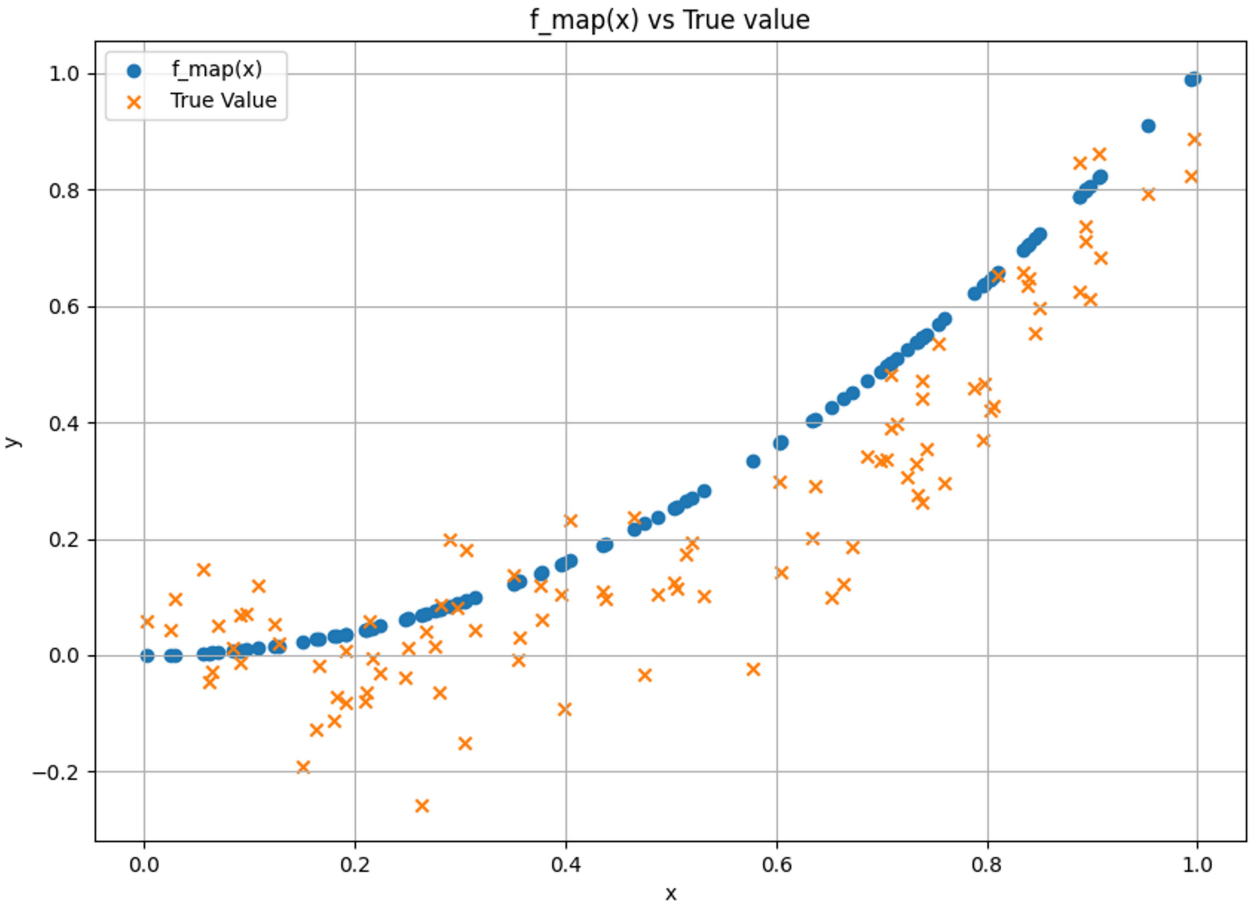
10.1

I got m = 2 and a posterior of 0.7613023972080507

11.1

```
def predict_MAP(x,X,Y):
  if MAP(X, Y) == 0:
    return 0
  return x**(MAP(X, Y))
```

12.1


f_map(x) vs True value
f_map(x)
True Value
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
y
0.0
0.2
0.4
0.6
0.8
1.0
x

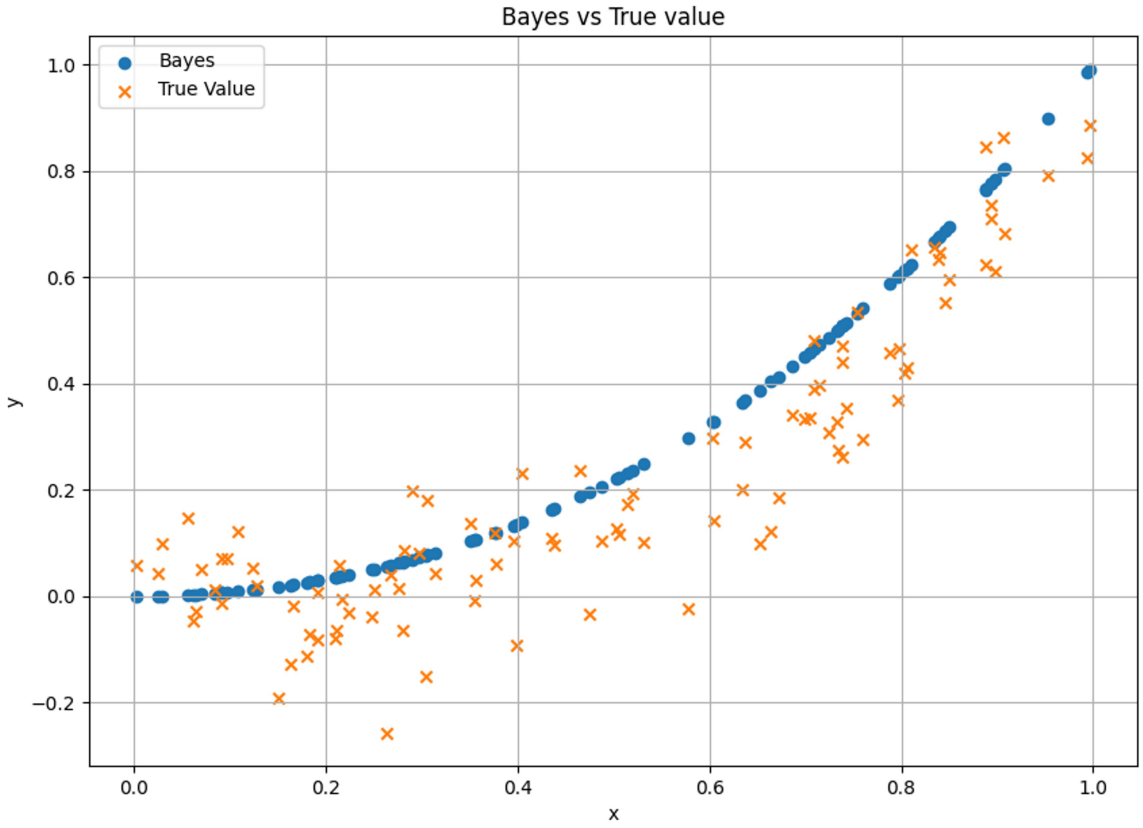
13.1

MSE = 0.02146290417007275
m = 2

14.2

```python
def predict_Bayes(x,X,Y):
    p_bayes = 0.0
    for m in range(10):
        if m == 0:
            p_bayes += posterior(X, Y, m)*0
        else:
            p_bayes += posterior(X, Y, m)*(x**m)
    return p_bayes
```

15.1


Bayes vs True value
Bayes
True Value
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
y
0.0
0.2
0.4
0.6
0.8
1.0
x

16.1

0.016372313860773324

17.1

Bayes error is lower. This makes sense because Bayes takes into account every model while MAP only chooses one model then predicts. MAP limits itself so it loses information thus making a worse prediction.