Jiaji Guo Machine learning homework 2

From the training data, in the 21 test data, if we use naive bayesian, we only get 7 out of them, with python index 4,8,9,12,14,16,18, this rate is only 1/3 =0.333, another interesting thing is that, if I create a vector, ['x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'x', 'positive'], the chance of positive is only 80.7%, which is contrast to our intuition.
The reason here is from this equation

$$
\begin{aligned}
p(C_k \mid x_1, \ldots, x_n) &\propto p(C_k, x_1, \ldots, x_n) \\
&\propto p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\
&\propto p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,.
\end{aligned}
$$

here we multiplied p(x1|Ck) p(x2|Ck) means that we assume that x1 and x2 affect the Ck(positive/negative) independently with each other, however, in the tic-tac-toe, this is not a rational/approximate assumption, so naive bayesian here will not be a good model.

P(“fill in =” | positive)

	x	o	b
1	0.4617	0.3073	0.2308
2	0.3479	0.3723	0.2796
3	0.4617	0.3073	0.2308
4	0.3560	0.3723	0.2715
5	0.5951	0.2243	0.1804
6	0.3609	0.3642	0.2747
7	0.4764	0.2991	0.2243
8	0.3658	0.3609	0.2731
9	0.4796	0.2975	0.2227

P(“fill in =” | negative)

	x	o	b
1	0.3738	0.4392	0.1869
2	0.4548	0.3115	0.2336
3	0.3769	0.4267	0.1962
4	0.4610	0.3021	0.2367
5	0.2710	0.5825	0.1464
6	0.4579	0.2990	0.2429
7	0.3582	0.4454	0.1962
8	0.4579	0.2990	0.2429
9	0.3613	0.4423	0.1962

Probability to be positive in test data

0.4099
0.4071
0.2748
0.4092
0.7029
0.4154
0.2817
0.4175
0.6849
0.5454
0.6282
0.8172
0.3359
0.7501
0.2584
0.5096
0.2448
0.7646
0.2695
0.7704
0.7427