Task 1

a)

$$P(B1) = P(B2) = 0.5$$

$$P(apple) = \frac{1}{2}\frac{8}{12} + \frac{1}{2}\frac{10}{12} = \frac{3}{4} = 0.75$$

$$P(B1|apple) = \frac{P(apple|B1)P(B1)}{P(apple)} = \frac{\frac{8}{12}\frac{1}{2}}{\frac{3}{4}} = \frac{4}{9} \approx 0.444$$

b)

$$P(y94, g96) = 0.2 \cdot 0.2 = 0.04$$

$$P(y96, g94) = 0.14 \cdot 0.1 = 0.014$$

$$P(y, g) = P(y94)P(g96) + P(y96)P(g94) = 0.04 + 0.014 = 0.054$$

$$P(y94, g96|y, g) = \frac{P(y, g|y94, g96)P(y94, g96)}{P(y, g)} = \frac{1 \cdot 0.04}{0.054} = \frac{20}{27} \approx 0.741$$

Task 3

The main idea is from A Simple KNN Algorithm for Text Categorization.

How do you represent the text?

Define a vector of length $|F|$, the size of the vocabulary
Each document is now represented as a vector $(w_1, \ldots, w_{|F|})^\top$ with length $|F|$ consisting of $|F|$ weights w_i with $1 \leq i \leq |F|$

The weights are according to some measure we can freely define, e.g. TF-IDF or constant values (1 and 0)

What distance function do you use?
Scalar product of two vectors
$\rightarrow$ number of matching values

What decision rule do you use?
Simple uniform class selection. The class that is the most represented by the k nearest neighbors is the predicted class

Example
Vocabulary:
father mother children family house monkey donkey cat dog shark

train doc0: father mother shark → vector: (1, 1, 0, 0, 0, 0, 0, 0, 0, 1) → class "family"
train doc1: father children family → vector: (1, 0, 1, 1, 0, 0, 0, 0, 0, 0) → class "family"
train doc2: monkey donkey dog → vector: (0, 0, 0, 0, 0, 1, 1, 0, 1, 0) → class "animal"
train doc3: children family house → vector: (0, 0, 1, 1, 1, 0, 0, 0, 0, 0) → class "family"

test doc4: children family monkey → vector: (0, 0, 1, 1, 0, 1, 0, 0, 0, 0)

distances:
→ d(d0, d4) = 0
→ d(d1, d4) = 2
→ d(d2, d4) = 1
→ d(d3, d4) = 2

k = 3
→ d1, d2, and d3 are the k = 3 closest neighbors
→ d1 and d3 belong to the class "family"
→ d3 belongs to the class "animal"

→ d4 is predicted to be in class "family"

done.

Advantages:
fast, simple, and easily adoptable to other measures than tf-idf or constant values

Disadvantages:
Accuracy may not be too good since it can "miss" and since the measure (constant value) is quite simple and doesn't take other things into consideration
Imagine doc3 is "mother family house" then d(d3, d4) = 1 which has the same score as doc2 despite it clearly belongs to the "family" class
→ in need of a good tie breaker, otherwise sometimes wrong classes will be predicted

Task 4

One big problem of knn is finding a good distance function. With increasing dimensionality the problem gets worse and worse (regardless of the computational cost). Simple distance functions are not sufficient anymore when the dimensions are big enough, because the results will most likely not be reasonable due to incporoating (probably) irrelevant data/features. The chance of a feature being irrelevant increases with increasing numbers of features.

One could try to not take all features into consideration, but only a smaller subset which would have to be chosen very carefully. Maybe some pre-processing step could determine which features could be more relevant than others and exclude those irrelevant features from further consideration. Therefore the number of considered features could be decreased considerably in order for the algorithm to work properly again or at least to make it viable again.