1.

To find the best guess G that minimizes the total amount of money charged after 1000 rounds, we need to minimize the expected value of the cost function: 1 + |G - R|^2.

Given:

I = 4 (always)

R is uniformly distributed between 0 and I^2 = 16 (inclusive)

The probability of each value of R is 1/17, as there are 17 possible values (0 to 16).

The expected cost is:

E[cost] = 1 + E[|G - R|^2]

E[cost] = 1 + Σ (|G - R|^2 × P(R))

To minimize the expected cost, we need to find the value of G that minimizes E[|G - R|^2].

Intuitively, the optimal value of G should be the median of the distribution of R. Since R is uniformly distributed between 0 and 16, the median is 8.

Therefore, the best guess G to make each time to minimize the total amount of money charged after 1000 rounds is 8.

2.

To estimate the conditional probabilities:

P(A|+) = 5/10 = 0.5
P(B|+) = 5/10 = 0.5
P(C|+) = 10/10 = 1
P(A|-) = 0/4 = 0
P(B|-) = 0/4 = 0
P(C|-) = 0/4 = 0

(b) Using Naive Bayes, for a test sample (A=0, B=1, C=0):

P(+|A=0, B=1, C=0) = P(A=0|+) * P(B=1|+) * P(C=0|+) * P(+) / P(A=0, B=1, C=0)
= 0.5 * 0.5 * 0 * (10/14) / P(A=0, B=1, C=0)
= 0

P(-|A=0, B=1, C=0) = P(A=0|-) * P(B=1|-) * P(C=0|-) * P(-) / P(A=0, B=1, C=0)
= 1 * 1 * 1 * (4/14) / P(A=0, B=1, C=0)
= 4/14 / P(A=0, B=1, C=0)

Normalizing, P(-|A=0, B=1, C=0) = 1, so the predicted class is -.

(c) Laplace smoothing with k=1 gives the following estimates:

P(A|+) = (5+1) / (10+3) = 6/13
P(B|+) = (5+1) / (10+3) = 6/13
P(C|+) = (10+1) / (10+3) = 11/13
P(A|-) = (0+1) / (4+3) = 1/7
P(B|-) = (0+1) / (4+3) = 1/7
P(C|-) = (0+1) / (4+3) = 1/7

(d) Using the conditional probabilities from (c):

P(+|A=0, B=1, C=0) = (7/13) * (6/13) * (2/13) * (10/14) / P(A=0, B=1, C=0)
= 0.0237 / P(A=0, B=1, C=0)

P(-|A=0, B=1, C=0) = (6/7) * (6/7) * (6/7) * (4/14) / P(A=0, B=1, C=0)
= 0.1469 / P(A=0, B=1, C=0)

Normalizing, P(-|A=0, B=1, C=0) = 0.8611, so the predicted class is - again.

(e) Laplace smoothing provides robustness against zero probabilities by adding pseudo-counts, allowing all probabilities to be non-zero. This avoids drastically changing the predicted probabilities when an unseen attribute value occurs. The Naive Bayes approach relies on the naive assumption of conditional independence between attributes given the class, which is often violated in practice. Laplace smoothing helps mitigate issues arising from this assumption to an extent.

3.
(a) Using Laplace smoothing with k=1, the conditional probabilities are:

P(A = 1|+) = (5+1) / (6+2) = 6/8 = 3/4
P(B = 1|+) = (3+1) / (6+2) = 4/8 = 1/2
P(C = 1|+) = (6+1) / (6+2) = 7/8
P(A = 1|-) = (0+1) / (4+2) = 1/6
P(B = 1|-) = (2+1) / (4+2) = 3/6 = 1/2
P(C = 1|-) = (0+1) / (4+2) = 1/6

(b) For a test sample (A=1, B=1, C=1), using Naive Bayes with the conditional probabilities from 3a:

P(+|A=1, B=1, C=1) $\propto$ P(A=1|+) * P(B=1|+) * P(C=1|+) * P(+)
= (3/4) * (1/2) * (7/8) * (6/10) = 0.1969

P(-|A=1, B=1, C=1) $\propto$ P(A=1|-) * P(B=1|-) * P(C=1|-) * P(-)
= (1/6) * (1/2) * (1/6) * (4/10) = 0.0037

Normalizing, P(+|A=1, B=1, C=1) = 0.9816, so the predicted class is +.

(c) P(A = 1) = 5/10 = 1/2
P(B = 1) = 5/10 = 1/2

A and B are independent if P(A,B) = P(A) * P(B).
But P(A=1, B=1) = 3/10 ≠ (1/2) * (1/2) = 1/4, so A and B are NOT independent.

(d) Repeating 3c with P(A=1), P(B=0) and P(A=1, B=0):

P(A=1, B=0) = 2/10 = 1/5
P(A=1) * P(B=0) = (1/2) * (1/2) = 1/4

Since P(A=1, B=0) ≠ P(A=1) * P(B=0), A and B are again shown to be dependent.

(e) P(A=1|Class=+) = 5/6, while P(A=1|Class=-) = 0/4 = 0.
P(B=1|Class=+) = 3/6 = 1/2, while P(B=1|Class=-) = 2/4 = 1/2.

4.

(a) The conditional pdf of X given being a crocodile is:
f(X|Crocodile) = 1/(2*sqrt(2π)) * exp(-0.5 * ((X-15)/2)^2)

This is a Gaussian distribution with mean 15 and standard deviation 2.

(b) The conditional pdf of X given being an alligator is:
f(X|Alligator) = 1/(2*sqrt(2π)) * exp(-0.5 * ((X-12)/2)^2)

This is a Gaussian distribution with mean 12 and standard deviation 2.

(c) The ideal decision boundary x* is the length where P(Crocodile|X=x*) = P(Alligator|X=x*).

Using Bayes' theorem and assuming equal prior probabilities:
P(X=x*|Crocodile)P(Crocodile) = P(X=x*|Alligator)P(Alligator)
f(x*|Crocodile) = f(x*|Alligator)

Equating the Gaussian pdfs and solving:

exp(-0.5 * ((x*-15)/2)^2) = exp(-0.5 * ((x*-12)/2)^2)
((x*-15)/2)^2 = ((x*-12)/2)^2
x*-15 = -(x*-12)
2x* = 27
x* = 13.5 feet

(d) When crocodiles are twice as common as alligators:
P(Crocodile) = 2/3, P(Alligator) = 1/3

The new decision boundary x* satisfies:
(2/3) * f(x*|Crocodile) = (1/3) * f(x*|Alligator)
2 * exp(-0.5 * ((x*-15)/2)^2) = exp(-0.5 * ((x*-12)/2)^2)

Solving numerically yields x* ≈ 14.1 feet, shifted towards the crocodile mean.

(e) Same as (d), with alligators now twice as common as crocodiles.
By symmetry, the decision boundary will shift towards the alligator mean:
x* ≈ 12.9 feet

(f) With different standard deviations, the decision boundary x* satisfies:
exp(-0.5 * ((x*-15)/4)^2) = exp(-0.5 * ((x*-12)/2)^2)
((x*-15)/4)^2 = ((x*-12)/2)^2
(x*-15)^2 = 4(x*-12)^2

x* ≈ 13.8

5.

(a) Given:
P(B = good, F = empty, G = empty, S = yes).

Using the probabilities from the Bayesian network:
P(B=good, F=empty, G=empty, S=yes) = P(B=good) * P(F=empty) * P(G=empty|B=good, F=empty) * P(S=yes|B=good, F=empty)
= 0.1 * 0.2 * 0.1 * 0.8
= 0.0016

(b) Given:
P(B = bad, F = empty, G = notempty, S = no).

Using the probabilities from the Bayesian network:
P(B=bad, F=empty, G=notempty, S=no) = P(B=bad) * P(F=empty) * P(G=notempty|B=bad, F=empty) * P(S=no|B=bad, F=empty)
= 0.9 * 0.2 * 0.2 * 1.0
= 0.036

(c) To calculate the probability that the car will start given the battery is bad:

P(S=yes|B=bad) = P(S=yes, B=bad) / P(B=bad)

First, calculate P(S=yes, B=bad):
P(S=yes, B=bad) = ΣF,G P(B=bad, F, G, S=yes)
= P(B=bad, F=empty, G=empty, S=yes) + P(B=bad, F=empty, G=notempty, S=yes) + P(B=bad, F=notempty, G=empty, S=yes) + P(B=bad, F=notempty, G=notempty, S=yes)
= 0.9 * 0.2 * 0.8 * 0.0 + 0.9 * 0.2 * 0.2 * 0.0 + 0.9 * 0.8 * 0.9 * 0.0 + 0.9 * 0.8 * 0.1 * 0.0
= 0

Then, P(S=yes|B=bad) = P(S=yes, B=bad) / P(B=bad) = 0 / 0.9 = 0

6. (a) The residual sum of squares (RSS) for a linear regression model $Y = \beta 1X + \varepsilon$ can be expressed as:

$RSS = \Sigma(yi - \hat{y}i)^2 = \Sigma(yi - (\beta 1 * xi))^2$

where yi are the observed values, $\hat{y}i$ are the predicted values, and xi are the known predictor values.

(b) To show that the RSS is minimized by the β1 given in equation (1), we take the derivative of RSS with respect to β1 and set it equal to zero:

$d/d\beta 1\ (RSS) = d/d\beta 1\ (\Sigma(yi - (\beta 1 * xi))^2)$
$= \Sigma(-2xi(yi - (\beta 1 * xi)))$
$= -2\Sigma(xiyi - \beta 1 * xi^2)$

Setting this equal to zero:

$-2\Sigma(xiyi - \beta 1 * xi^2) = 0$
$\Sigma(xiyi) - \beta 1 * \Sigma(xi^2) = 0$

Solving for β1:

$\beta 1 = \Sigma(xiyi) / \Sigma(xi^2)$

This matches the expression given in equation (1), proving that this choice of $\beta 1$ minimizes the RSS.

(c) If we have a new test data point xnew, we can predict its label $\hat{y}new$ using the estimated coefficient $\beta 1$ from equation (1):

$\hat{y}new = \beta 1 * xnew$

(d) The prediction in (c) is a special case of weighted KNN regression with k=n (all training points are used) and the similarity measure being:

$sim(xnew, xi) = xi / \Sigma(xj^2)$
the predicted value is a weighted average of all yi, with weights proportional to xi.