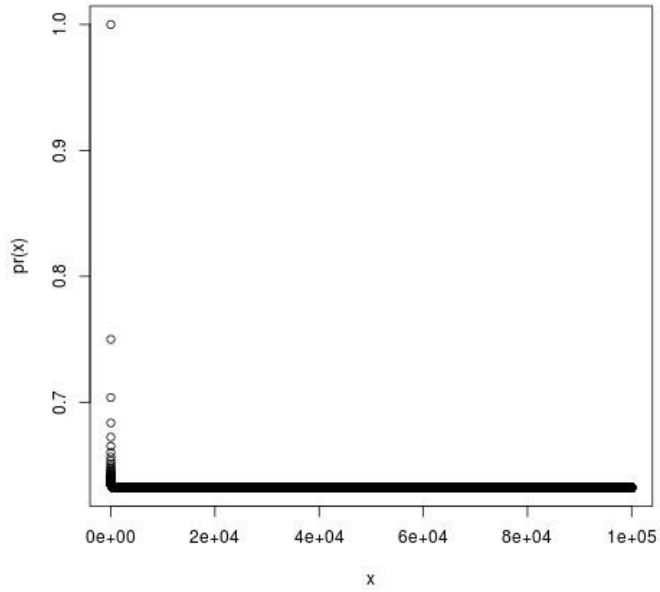
2)

g)

```
> jpeg("plot_2g.jpg")
> pr = function(n) return(1 - (1 - 1/n)^n)
> x = 1:1e+05
> plot(x, pr(x))
> dev.off()
```



the plot reaches asymptote of 0.63

h)

```
> store=rep(NA, 10000)
> for(i in 1:10000) {
+ store[i]=sum(sample (1:100,
rep=TRUE)==4) >0
+ }
> mean(store)
[1] 0.6331
```

This confirms the results found in the previous part

5)

a)

```
> library(ISLR)
> summary(Default)
 default    student       balance           income
 No :9667   No :7056   Min.   :   0.0   Min.   :  772
 Yes: 333   Yes:2944   1st Qu.: 481.7   1st Qu.:21340
                       Median : 823.6   Median :34553
                       Mean   : 835.4   Mean   :33517
                       3rd Qu.:1166.3   3rd Qu.:43808
                       Max.   :2654.3   Max.   :73554
> attach(Default)
> set.seed(1)
> glm.fit = glm(default ~ income + balance, data = Default, family = binomial)
> glm.fit

Call:  glm(formula = default ~ income + balance, family = binomial,
    data = Default)

Coefficients:
(Intercept)       income      balance
 -1.154e+01    2.081e-05    5.647e-03

Degrees of Freedom: 9999 Total (i.e. Null);  9997 Residual
Null Deviance:              2921
Residual Deviance: 1579       AIC: 1585
```

b)
```
# i.
> train = sample(dim(Default)[1], dim(Default)[1]/2)
# ii.
> glm.fit = glm(default ~ income + balance, data = Default, family = binomial, subset = train)
# iii.
> glm.pred = rep("No", dim(Default)[1]/2)
> glm.probs = predict(glm.fit, Default[-train, ], type = "response")
> glm.pred[glm.probs > 0.5] = "Yes"
# iv.
> mean(glm.pred != Default[-train, ]$default)
[1] 0.0286
```

Test error rate from validation set approach: 0.0286

c)
```
#1st
[1] 0.0236

#2nd
[1] 0.0252

#3rd
[1] 0.0268
```

On average, test error rate: 0.02

d)
```
> train = sample(dim(Default)[1], dim(Default)[1]/2)
> glm.fit = glm(default ~ income + balance + student, data = Default, family = binomial,
+     subset = train)
> glm.pred = rep("No", dim(Default)[1]/2)
> glm.probs = predict(glm.fit, Default[-train, ], type = "response")
> glm.pred[glm.probs > 0.5] = "Yes"
> mean(glm.pred != Default[-train, ]$default)
[1] 0.0264
```

Test error rate: 0.0264 with dummy variable ‘student’, it doesn’t appear to reduce the test error rate with adding dummy variable ‘student’

7)
a)
```
> library(ISLR)
> summary(Weekly)
      Year          Lag1               Lag2               Lag3
 Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
 1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
```

```
 Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
 Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472
 3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090
 Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
     Lag4              Lag5              Volume            Today
 Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747   Min.   :-18.1950
 1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202   1st Qu.: -1.1540
 Median :  0.2380   Median :  0.2340   Median :1.00268   Median :  0.2410
 Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462   Mean   :  0.1499
 3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373   3rd Qu.:  1.4050
 Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821   Max.   : 12.0260
 Direction
 Down:484
 Up  :605

> set.seed(1)
> attach(Weekly)
> glm.fit = glm(Direction ~ Lag1 + Lag2, data = Weekly, family = binomial)
> summary(glm.fit)

Call:
glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.623  -1.261   1.001   1.083   1.506

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.22122    0.06147   3.599 0.000319 ***
Lag1        -0.03872    0.02622  -1.477 0.139672
Lag2         0.06025    0.02655   2.270 0.023232 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1496.2  on 1088  degrees of freedom
Residual deviance: 1488.2  on 1086  degrees of freedom
AIC: 1494.2

Number of Fisher Scoring iterations: 4

b)
> glm.fit = glm(Direction ~ Lag1 + Lag2, data = Weekly[-1, ], family = binomial)
> summary(glm.fit)

Call:
glm(formula = Direction ~ Lag1 + Lag2, family = binomial, data = Weekly[-1, ])
```

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6258  -1.2617   0.9999   1.0819   1.5071

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.22324    0.06150   3.630 0.000283 ***
Lag1        -0.03843    0.02622  -1.466 0.142683
Lag2         0.06085    0.02656   2.291 0.021971 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1494.6  on 1087  degrees of freedom
Residual deviance: 1486.5  on 1085  degrees of freedom
AIC: 1492.5

Number of Fisher Scoring iterations: 4
```

c)

```
> predict.glm(glm.fit, Weekly[1, ], type = "response")
        1
0.5713923
> Weekly[1, ]
  Year  Lag1  Lag2   Lag3   Lag4   Lag5   Volume Today Direction
1 1990 0.816 1.572 -3.936 -0.229 -3.484 0.154976 -0.27      Down
```

So the prediction is not correct, since prediction is UP, true direction was DOWN.

d)

```
> count = rep(0, dim(Weekly)[1])
> for (i in 1:(dim(Weekly)[1])) {
+     glm.fit = glm(Direction ~ Lag1 + Lag2, data = Weekly[-i, ], family = binomial)
+     is_up = predict.glm(glm.fit, Weekly[i, ], type = "response") > 0.5
+     is_true_up = Weekly[i, ]$Direction == "Up"
+     if (is_up != is_true_up)
+         count[i] = 1
+ }
> sum(count)
[1] 490
```

Total errors: 490

e)

```
> mean(count)
[1] 0.4499541
```

LOOCV estimates a test error rate of 0.45

9)
a)

```
> library(MASS)
> summary(Boston)
      crim                zn              indus            chas        
 Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000  
 1st Qu.: 0.08204   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000  
 Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000  
 Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917  
 3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000  
 Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000  
      nox               rm             age              dis         
 Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130  
 1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100  
 Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207  
 Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795  
 3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188  
 Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127  
      rad              tax           ptratio          black       
 Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32  
 1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38  
 Median : 5.000   Median :330.0   Median :19.05   Median :391.44  
 Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67  
 3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23  
 Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90  
     lstat            medv      
 Min.   : 1.73   Min.   : 5.00  
 1st Qu.: 6.95   1st Qu.:17.02  
 Median :11.36   Median :21.20  
 Mean   :12.65   Mean   :22.53  
 3rd Qu.:16.95   3rd Qu.:25.00  
 Max.   :37.97   Max.   :50.00  

> mean(medv)
[1] 22.53281
```

b)

```
> sd(medv)/sqrt(length(medv))
[1] 0.4088611
```

c)

```
> boot(medv, function(data, index) return(mean(data[index])), 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = function(data, index) return(mean(data[index])),
    R = 1000)
```

Bootstrap Statistics :
    original   bias   std. error
t1* 22.53281 0.0027583   0.4120131

Standard error: 0.4120131 similar to part b) 0.4088611

d)
> t.test(medv)

        One Sample t-test

data: medv
t = 55.111, df = 505, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 21.72953 23.33608
sample estimates:
mean of x
 22.53281

> c(bstrap$t0 - 2 * 0.4119, bstrap$t0 + 2 * 0.4119)
[1] 21.70901 23.35661

Bootstrap estimates (21.70901, 23.35661) compared to (21.72953, 23.33608) t-test

e)
> median(medv)
[1] 21.2

f)
> boot.fn = function(data, index) return(median(data[index]))
> boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP

Call:
boot(data = medv, statistic = boot.fn, R = 1000)

Bootstrap Statistics :
    original bias   std. error
t1*   21.2 -0.0025   0.374358

Bootstrap estimates median of 21.2 with standard error: 0.374358

g)
```
> quantile(medv, c(0.1))
  10%
12.75
```

h)
```
> boot.fn = function(data, index) return(quantile(data[index], c(0.1)))
> boot(medv, boot.fn, 1000)

ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = medv, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
    original  bias    std. error
t1*    12.75  0.0261   0.4912231
```

Bootstrap estimates 10^{th}-quantile of 12.75 with standard error 0.4912231