Supplement for Lecture 27: Logistic Regression

Load Data

```
#Load and Preview Data
data(Putts1)
head(Putts1)
```

```
##   Length Made
## 1      3    1
## 2      3    1
## 3      3    1
## 4      3    1
## 5      3    1
## 6      3    1
```

```
#Inspect Data
nrow(Putts1)
```

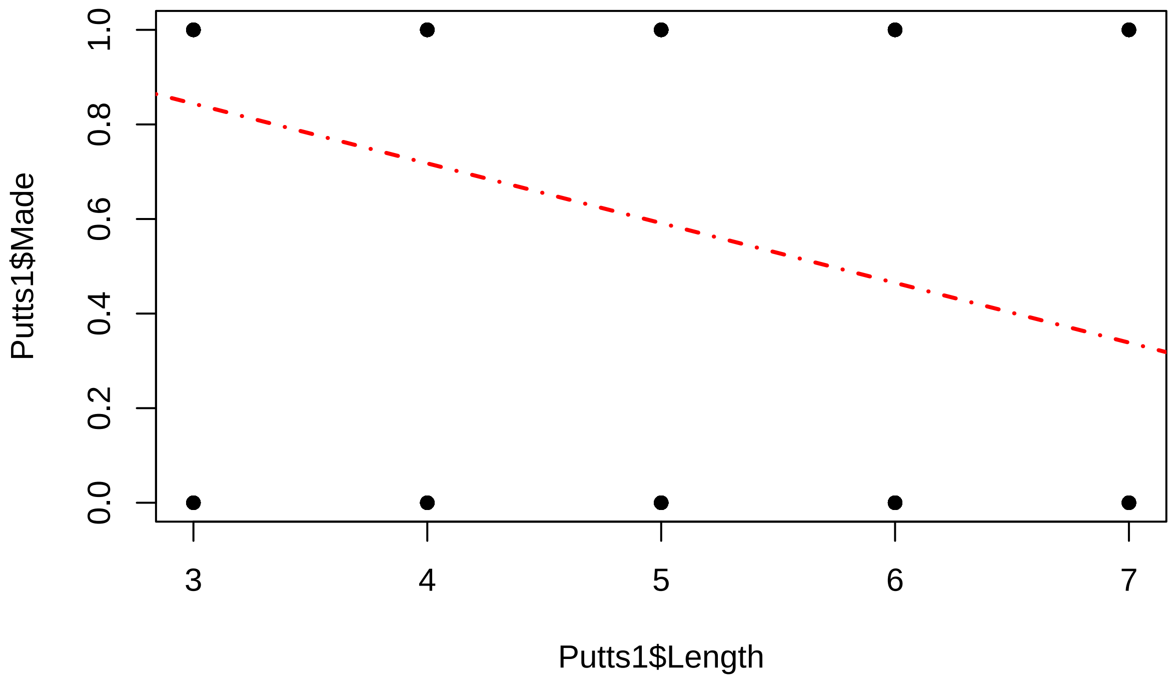
```
## [1] 587
```

```
table(Putts1$Made,Putts1$Length)
```

```
##
##      3  4  5  6  7
##   0 17 31 47 64 90
##   1 84 88 61 61 44
```

Plot of Raw Data

```
plot(Putts1$Length,Putts1$Made,pch=16)
abline(lm(Made~Length,data=Putts1),lty=4,lwd=2,col="red")
```



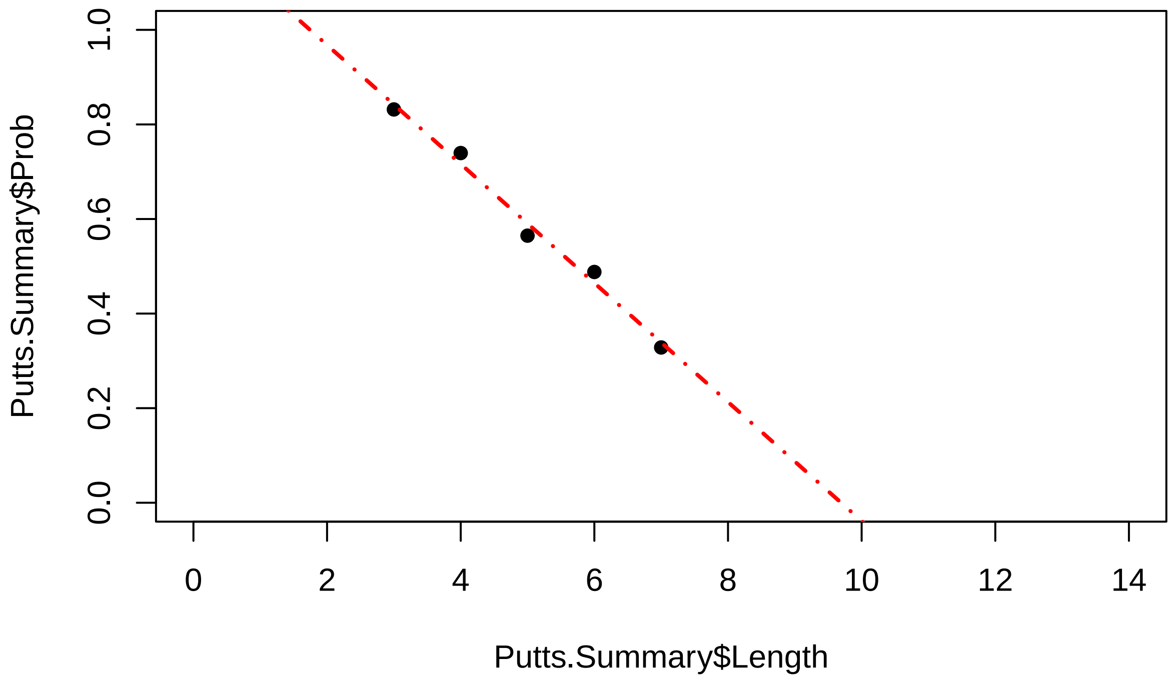
Plot of Summarized Data

```
tapply(Putts1$Made,Putts1$Length,FUN=mean)
```

```
##         3         4         5         6         7
## 0.8316832 0.7394958 0.5648148 0.4880000 0.3283582
```

```
Putts.Summary=data.frame(Length=3:7,Prob=tapply(Putts1$Made,Putts1$Length,FUN=mean))

plot(Putts.Summary$Length,Putts.Summary$Prob,pch=16,ylim=c(0,1),xlim=c(0,14))
abline(lm(Prob~Length,data=Putts.Summary),lty=4,lwd=2,col="red")
```



Logistic Regression Model

```
putt.mod = glm(Made~Length,family=binomial,data=Putts1)
summary(putt.mod)
```

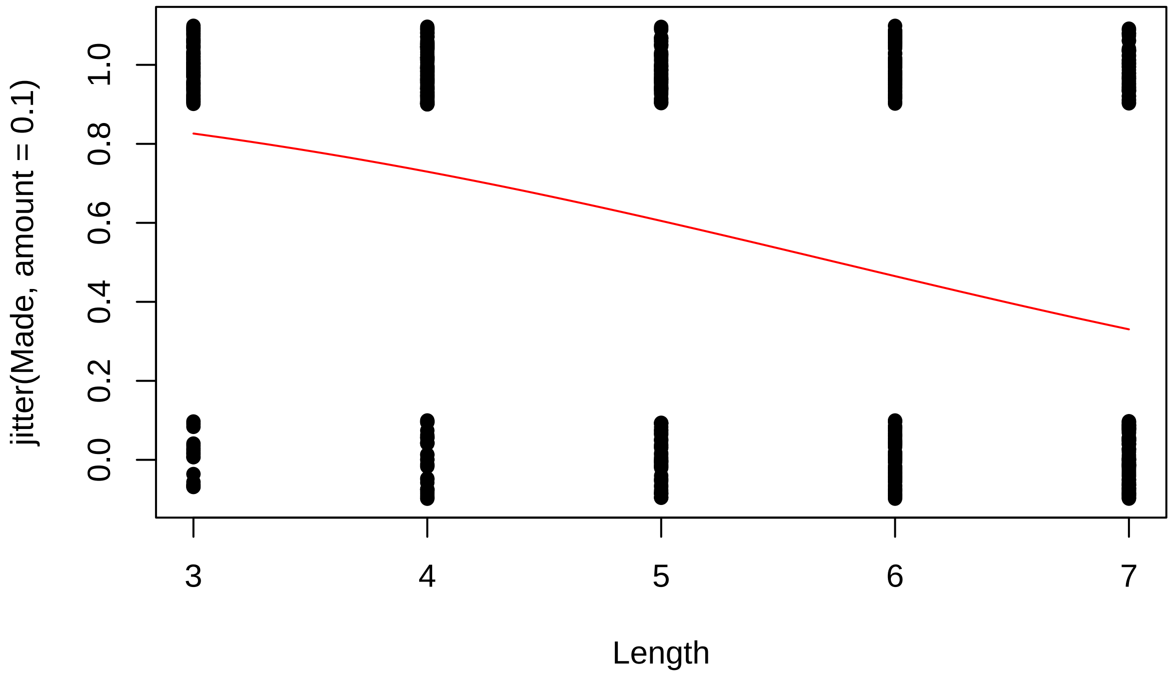
```
##
## Call:
## glm(formula = Made ~ Length, family = binomial, data = Putts1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8705  -1.1186   0.6181   1.0026   1.4882
##
## Coefficients:
##             Estimate Std. Error z value            Pr(>|z|)
## (Intercept)  3.25684    0.36893   8.828 <0.0000000000000002 ***
## Length      -0.56614    0.06747  -8.391 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 800.21  on 586  degrees of freedom
## Residual deviance: 719.89  on 585  degrees of freedom
## AIC: 723.89
```

```
##
## Number of Fisher Scoring iterations: 4
```

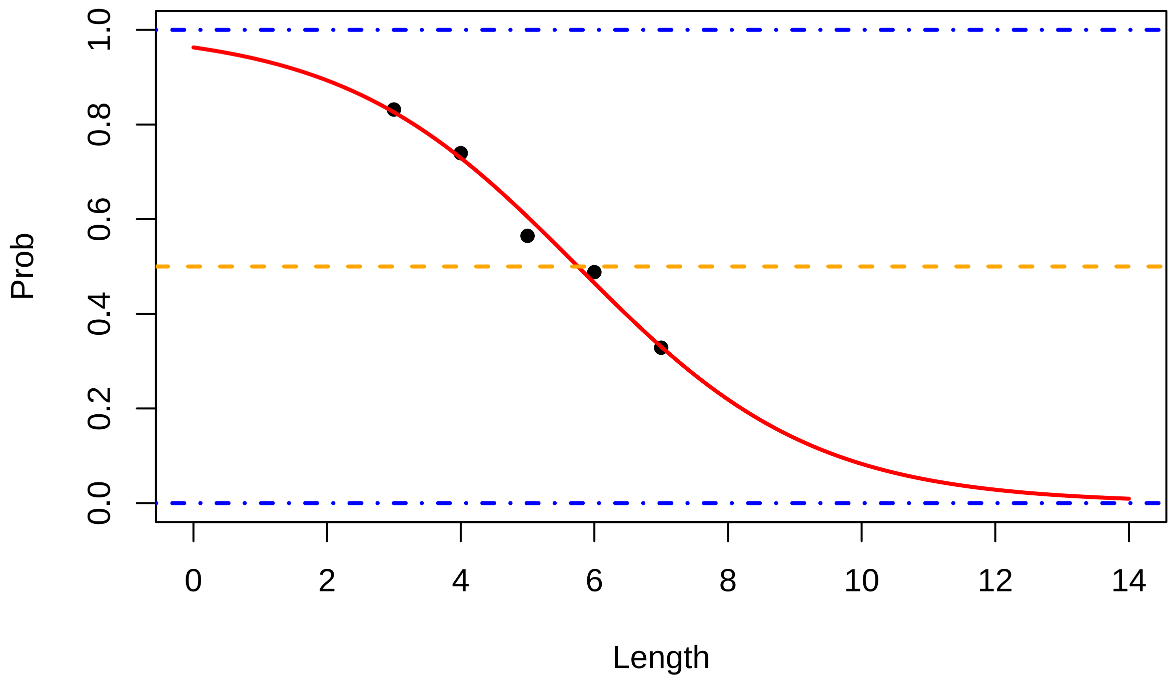
Visualization of Logistic Regression Model

```
b0 = as.numeric(coef(putt.mod)[1])
b1 = as.numeric(coef(putt.mod)[2])

plot(jitter(Made,amount=0.1)~Length,data=Putts1,pch=16)
curve(exp(b0+b1*x)/(1+exp(b0+b1*x)),col="red",add=TRUE)
```



```
plot(Prob~Length,data=Putts.Summary,pch=16,ylim=c(0,1),xlim=c(0,14))
curve(exp(b0+b1*x)/(1+exp(b0+b1*x)),col="red",lwd=2,add=TRUE)
abline(h=c(0,1),lwd=2,col="blue",lty=4)
abline(h=0.5,lwd=2, col="orange",lty=2)
```



Comparing Sample Proportions to Estimated Probabilities

```
prop=as.numeric(tapply(Putts1$Made,Putts1$Length,FUN=mean))
prob=as.numeric(predict(putt.mod,type="response",newdata=data.frame(Length=3:7)))

OUT = data.frame(Length=3:7,Proportion = prop, Probability=prob)
OUT
```

```
##   Length Proportion Probability
## 1      3  0.8316832   0.8261256
## 2      4  0.7394958   0.7295364
## 3      5  0.5648148   0.6049492
## 4      6  0.4880000   0.4650541
## 5      7  0.3283582   0.3304493
```

Odds

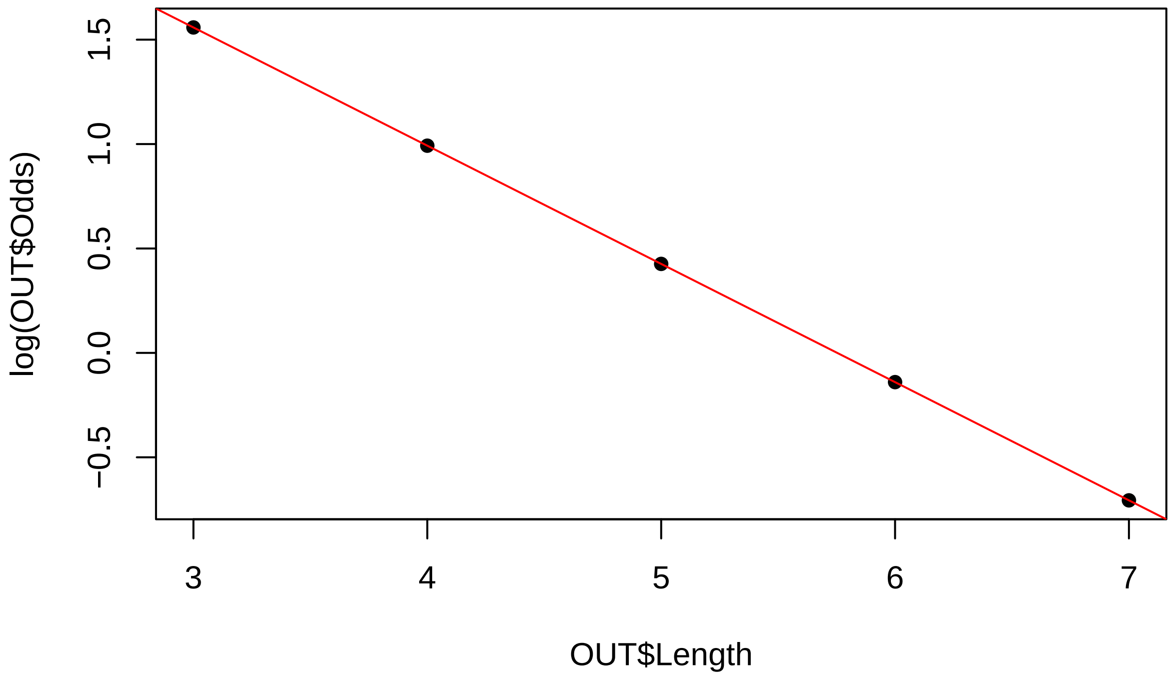
```
#Calculate using Formula
OUT$Odds = OUT$Probability/(1-OUT$Probability)
OUT
```

```
##   Length Proportion Probability     Odds
## 1      3  0.8316832   0.8261256 4.751277
## 2      4  0.7394958   0.7295364 2.697355
## 3      5  0.5648148   0.6049492 1.531320
## 4      6  0.4880000   0.4650541 0.869348
## 5      7  0.3283582   0.3304493 0.493539
```

```
#Calculate using Predict Function
exp(predict(putt.mod,newdata=data.frame(Length=3:7)))
```

```
##        1        2        3        4        5 
## 4.751277 2.697355 1.531320 0.869348 0.493539
```

```
#Plot log(odds) vs Length
plot(x=OUT$Length,y=log(OUT$Odds),pch=16)
abline(a=b0,b=b1,col="red")
```



Odds Ratios

```
#Compare 3ft Putts to 7ft Putts
exp(b0+b1*3)/exp(b0+b1*7)
```

```
## [1] 9.626953
```

```
#Compare 7ft Putts to 3ft Putts (Reciprocal)
exp(b0+b1*7)/exp(b0+b1*3)
```

```
## [1] 0.103875
```

Interpretation: The odds of making a 3ft putt is 9.63 times the odds of making a 7ft putt. This is equivalent to saying the odds of making a 7ft putt is 0.10 times the odds of making a 3ft putt. Typically, statisticians prefer interpreting odds >1 which requires putting the group with the higher chance of success in the numerator.

Relationship to Slope of Line

```
#Compare 4ft Putts to 3ft Putts
exp(b0+b1*4)/exp(b0+b1*3)
```

```
## [1] 0.5677116
```

```
#Compare 7ft Putts to 6ft Putts
exp(b0+b1*7)/exp(b0+b1*6)
```

```
## [1] 0.5677116
```

```
#Calculate Slope From Odds Ratio
log(0.5677116)
```

```
## [1] -0.5661417
```

```
b1
```

```
## [1] -0.5661417
```

```
#Notice the difference here
exp(b0+b1*7)
```

```
## [1] 0.493539
```

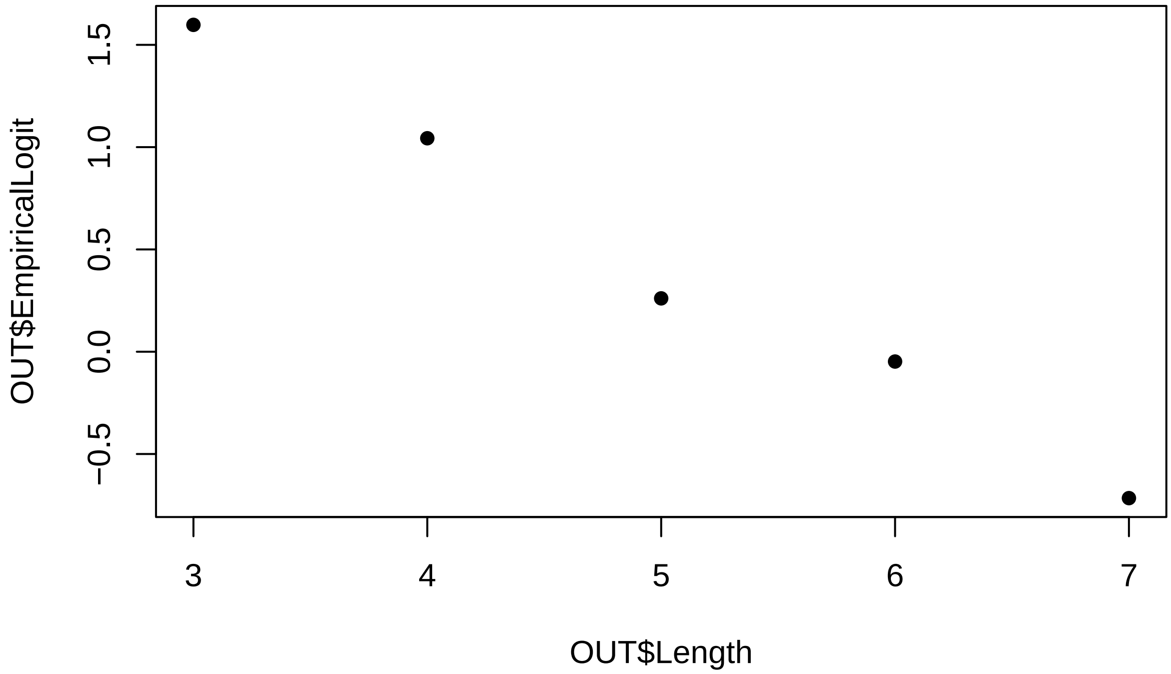
```
exp(b0+b1*6)*exp(b1)
```

```
## [1] 0.493539
```

Notice: For every one unit increase in X, the odds of success increases by a factor of e^b1

Empirical Logit Plot

```
OUT$EmpiricalLogit = log(OUT$Proportion/(1-OUT$Proportion))
plot(x=OUT$Length,y=OUT$EmpiricalLogit,pch=16)
```



```
lm(EmpiricalLogit~Length,data=OUT)
```

```
##
## Call:
## lm(formula = EmpiricalLogit ~ Length, data = OUT)
##
## Coefficients:
## (Intercept)       Length
##      3.2865      -0.5718
```

```
glm(Made~Length,data=Putts1,family="binomial")
```

```
##
## Call:  glm(formula = Made ~ Length, family = "binomial", data = Putts1)
##
## Coefficients:
## (Intercept)       Length
##      3.2568      -0.5661
##
## Degrees of Freedom: 586 Total (i.e. Null);  585 Residual
## Null Deviance:       800.2
## Residual Deviance: 719.9     AIC: 723.9
```

Hypothesis Test and CI for Slope

```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.2.3
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
#Notice z value and p-value
summary(putt.mod)
```

```
##
## Call:
## glm(formula = Made ~ Length, family = binomial, data = Putts1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8705  -1.1186   0.6181   1.0026   1.4882
##
## Coefficients:
##             Estimate Std. Error z value             Pr(>|z|)
## (Intercept)  3.25684    0.36893   8.828 <0.0000000000000002 ***
## Length      -0.56614    0.06747  -8.391 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 800.21  on 586  degrees of freedom
## Residual deviance: 719.89  on 585  degrees of freedom
## AIC: 723.89
##
## Number of Fisher Scoring iterations: 4
```

```
-0.56614/0.06747
```

```
## [1] -8.390989
```

```
2*(1-pnorm(abs(-8.391)))
```

```
## [1] 0
```

```
#Acquire CI for Slope
SE.b1=summary(putt.mod)$coefficients[2,2]
b1-1.96*SE.b1
```

```
## [1] -0.6983844
```

```
b1+1.96*SE.b1
```

```
## [1] -0.433899
```

```
confint(putt.mod) #Incorrect (Different Formula for Standard Error of Slope)
```

```
## Waiting for profiling to be done...
##                   2.5 %     97.5 %
## (Intercept)  2.5492465  3.9972923
## Length      -0.7010561 -0.4362681
```

```
confint.default(putt.mod) #Correct
```

```
##                  2.5 %     97.5 %
## (Intercept)  2.533746  3.9799310
## Length      -0.698382 -0.4339014
```

```
#Create CI for Odd Ratio
exp(confint.default(putt.mod))
```

```
##                   2.5 %     97.5 %
## (Intercept) 12.6006177 53.5133410
## Length       0.4973894  0.6479761
```

Likelihood Ratio Test

```
G.stat = summary(putt.mod)$null.deviance-summary(putt.mod)$deviance #Difference in Deviance
G.stat
```

```
## [1] 80.31729
```

```
df.G.stat = summary(putt.mod)$df.null - summary(putt.mod)$df.residual #Difference in Degrees of Freedom
df.G.stat
```

```
## [1] 1
```

```
pvalue = 1-pchisq(G.stat,df=df.G.stat) #area to right in chi-square distribution
pvalue
```

```
## [1] 0
```

Probabilities of Likelihood

```
#Full Deck
(1/52)*(1/51)
```

```
## [1] 0.0003770739
```

```
#Euchre Deck
(1/24)*(1/23)
```

```
## [1] 0.001811594
```

```
#Red Deck
(1/26)*(1/25)
```

```
## [1] 0.001538462
```