Worksheet 2

Working with probability distributions in R for MATH414

Once again load the fastR2 package. If it is not installed, run the following commands.

```
install.packages("devtools")
```

```
install.packages("fastR2",repos = "http://cran.us.r-project.org")
```

The following command loads the fastR2 package.

```
require(fastR2)
```

Datasets

Inbuilt Datasets There are datasets built into R which are loaded as follows

```
data(iris)
```

Import Datasets If you wish to load you own data, it could be txt file or a csv file. A possible source for already cleaned up data regarding India can be found at https://www.dataforindia.com/. This website contains articles on data related to population, health, economy, living conditions, work, and measurement. Each article has graphs and by clicking a certain link on this graphic, you can download the data as a csv file which you can then load into R. However, the data is usually time series data or data relating to distribution of some variable across indian states. Hence, it might not be typically useful for the kind of exercises we might be interested in. Another useful website is https://www.kaggle.com/datasets. You can download more pedagogy based datasets but you have to sign up for an account with them. Please download the dataset at https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009. You can directly download the csv file if you scroll down a bit.

```
winedata <- read.csv("~/Downloads/winequality.csv", header=TRUE)
```

```
glimpse(winedata)
```

```
## Rows: 1,599
## Columns: 12
## $ fixed.acidity        <dbl> 7.4, 7.8, 7.8, 11.2, 7.4, 7.4, 7.9, 7.3, 7.8, 7.5~
## $ volatile.acidity     <dbl> 0.700, 0.880, 0.760, 0.280, 0.700, 0.660, 0.600, ~
## $ citric.acid          <dbl> 0.00, 0.00, 0.04, 0.56, 0.00, 0.00, 0.06, 0.00, 0~
## $ residual.sugar       <dbl> 1.9, 2.6, 2.3, 1.9, 1.9, 1.8, 1.6, 1.2, 2.0, 6.1,~
## $ chlorides            <dbl> 0.076, 0.098, 0.092, 0.075, 0.076, 0.075, 0.069, ~
## $ free.sulfur.dioxide  <dbl> 11, 25, 15, 17, 11, 13, 15, 15, 9, 17, 15, 17, 16~
## $ total.sulfur.dioxide <dbl> 34, 67, 54, 60, 34, 40, 59, 21, 18, 102, 65, 102,~
## $ density              <dbl> 0.9978, 0.9968, 0.9970, 0.9980, 0.9978, 0.9978, 0~
## $ pH                   <dbl> 3.51, 3.20, 3.26, 3.16, 3.51, 3.51, 3.30, 3.39, 3~
## $ sulphates            <dbl> 0.56, 0.68, 0.65, 0.58, 0.56, 0.56, 0.46, 0.47, 0~
## $ alcohol              <dbl> 9.4, 9.8, 9.8, 9.8, 9.4, 9.4, 9.4, 10.0, 9.5, 10.~
## $ quality              <int> 5, 5, 5, 6, 5, 5, 5, 7, 7, 5, 5, 5, 5, 5, 5, 5, 7~
```

Exercise 1: Import this dataset into R.

Enter Datasets by hand You can create columns of a dataset using

```
myData1 <- c(15, 18, 12, 21, 23, 50, 15)
myData2 <- c("red", "red", "orange", "green", "blue", "blue", "red")
```

You can create a dataframe by

```
myDataFrame <- data.frame(color = myData2, number = myData1)
myDataFrame
```

```
##    color number
## 1    red     15
## 2    red     18
## 3 orange     12
## 4  green     21
## 5   blue     23
## 6   blue     50
## 7    red     15
```

> **Exercise 2**: Similarly create a dataset where one column is numbers from 1 to 100 and the other column is the squares of those numbers.

Probability Distributions in R

Corresponding to all well-known probability distributions, R has four commands to produce an action, for example, rbinom(m,n,p) produces m samples from Bin(n,p); dbinom(x,n,p) returns value of Bin(n,p) at x; pbinom(x,n,p) returns the cumulative distribution function at x; and qbinom(alpha,n,p) returns x such that alpha = pbinom(x,n,p). One can repeat this with pois, exp, norm, beta, gamma, chisq, t distributions.

```
dbinom(5,50,0.9)
```

```
## [1] 1.251107e-39
```

```
pbinom(5,50,0.9)
```

```
## [1] 1.26636e-39
```

```
qbinom(0.975,50,0.9)
```

```
## [1] 49
```

The last one will return 50 samples from Bin(100,0.9)

```
rbinom(50,100,0.9)
```

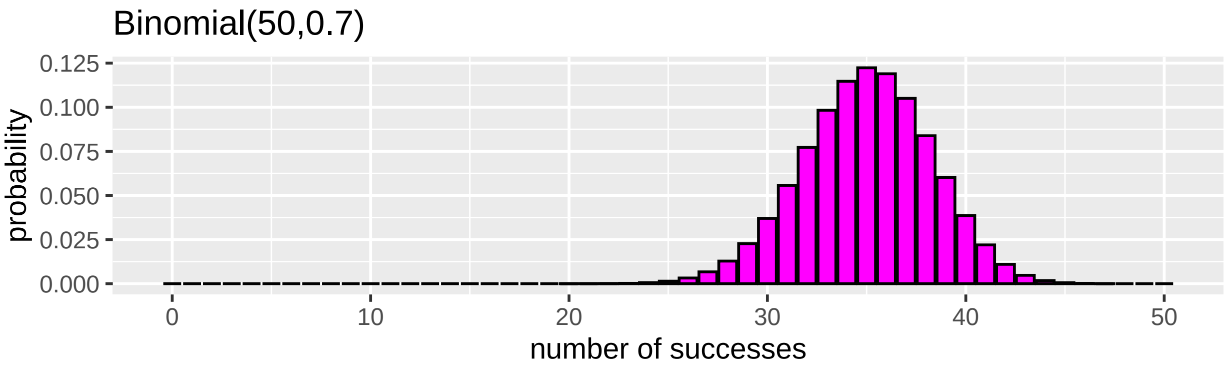
```
##  [1] 92 90 90 92 91 93 90 86 86 90 95 90 90 86 86 92 87 85 88 92 91 95 89 91 86
## [26] 88 90 91 90 93 86 91 84 91 87 86 90 91 90 89 87 93 90 89 90 91 90 87 89 90
```

> **Exercise 3**: Try the previous three commands for Poisson distribution.

Plotting pmfs, pdfs

We can use gf_col to plot pmfs

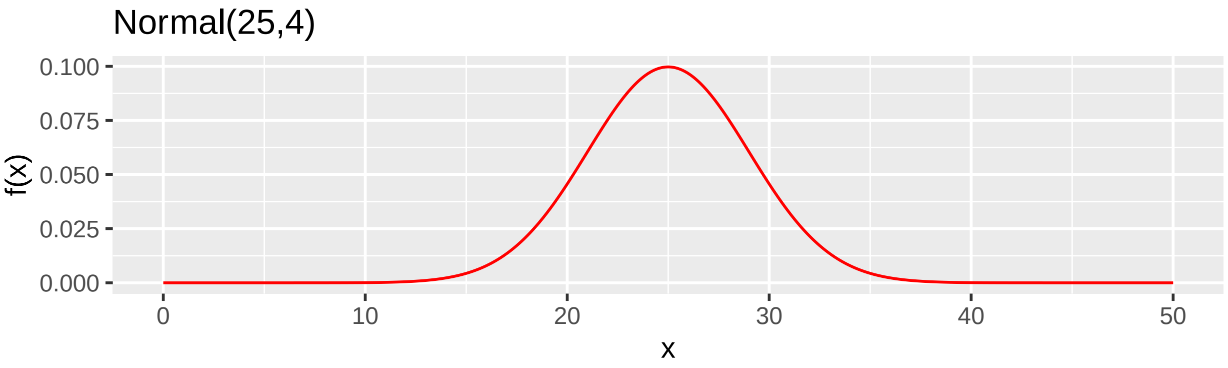
```
gf_col(dbinom(0:50,50,0.7) ~ 0:50, fill = "magenta", colour="black") %>%
  gf_labs(title = "Binomial(50,0.7)", x="number of successes", y="probability")
```



Exercise 4: Plot the pmf of Poisson distribution.

We can use gf_line to plot pdfs

```
x <- seq(0,50,by=0.01)
y <- dnorm(x,25,4)
gf_line(y~x, col="red") %>%
  gf_labs(title="Normal(25,4)",x="x",y="f(x)")
```



Exercise 4: Plot the pdf of the chi square distribution on 19 degrees of freedom.

Simulate Datasets

One can sample from a dataset or column with or without replacement

```
samples <- rbinom(500,100,0.8)
sample(samples,50)
```

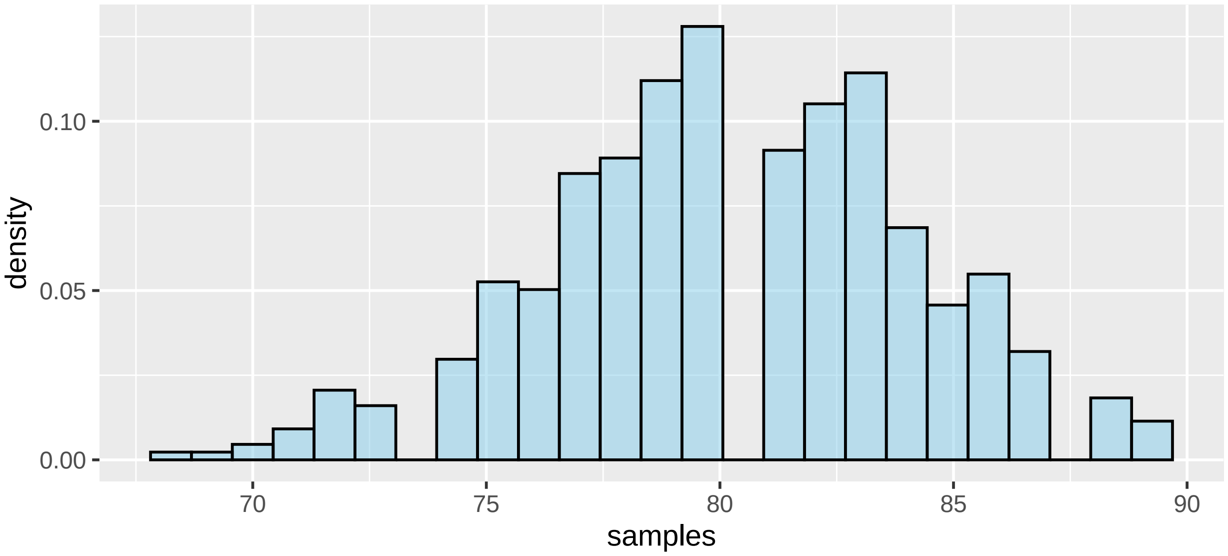
```
##  [1] 80 88 79 83 89 86 80 79 83 75 80 85 77 81 86 79 84 80 77 78 84 83 80 76 83
## [26] 85 75 87 84 82 84 80 85 77 85 85 79 88 85 83 75 84 78 86 78 76 84 80 75 81
```

```
resample(samples,50)
```

```
##  [1] 78 80 83 81 76 82 85 83 83 83 78 86 86 86 87 81 82 80 86 79 80 76 84 76 84
## [26] 85 69 80 71 87 75 80 79 83 82 80 78 76 80 81 74 69 85 80 89 84 85 80 85 82
```

Next, we can plot this using gf_histogram

```
gf_dhistogram( ~ samples, color = "black", fill = "skyblue")
```



Exercise 5: Find 1000 sample points from the chi square distribution and plot its histogram.

Fitting Datasets to distributions

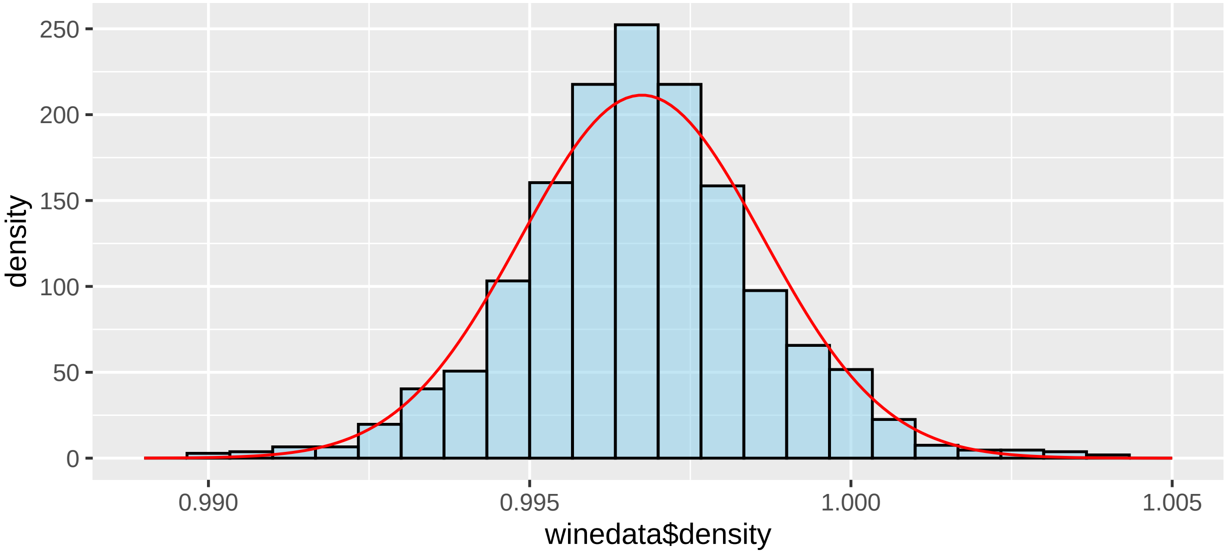
In the previous dataset on wine quality, let us try to fit a normal distribution to the variable "density". We will first find the sample mean and the sample variance and see how it looks.

Find estimators

```
m <- mean(winedata$density)
s <- sd(winedata$density)
```

Overlaying plots We will plot this data as a histogram and overlay the normal distribution on top of it.

```
x <- seq(0.989,1.005,by=0.0001)
y <- dnorm(x,m,s)
gf_dhistogram( ~ winedata$density, color = "black", fill = "skyblue") %>%
  gf_line(y~x, col="red")
```



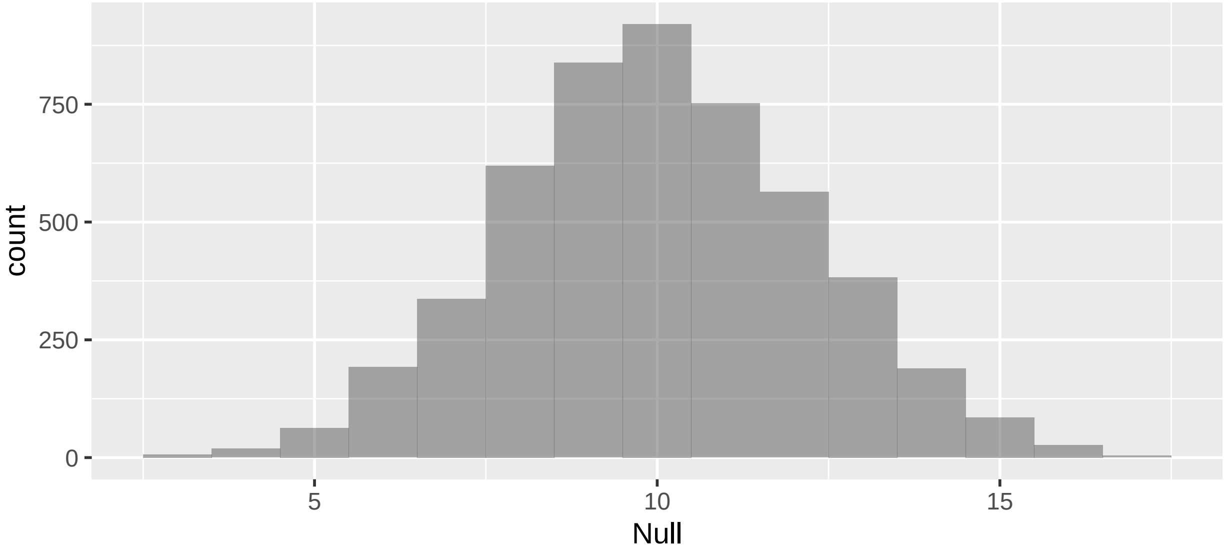
Exercise 6: Repeat the above procedure for the column called residual.sugar.

A simulation of the Binomial test

Testing the biasedness of a coin

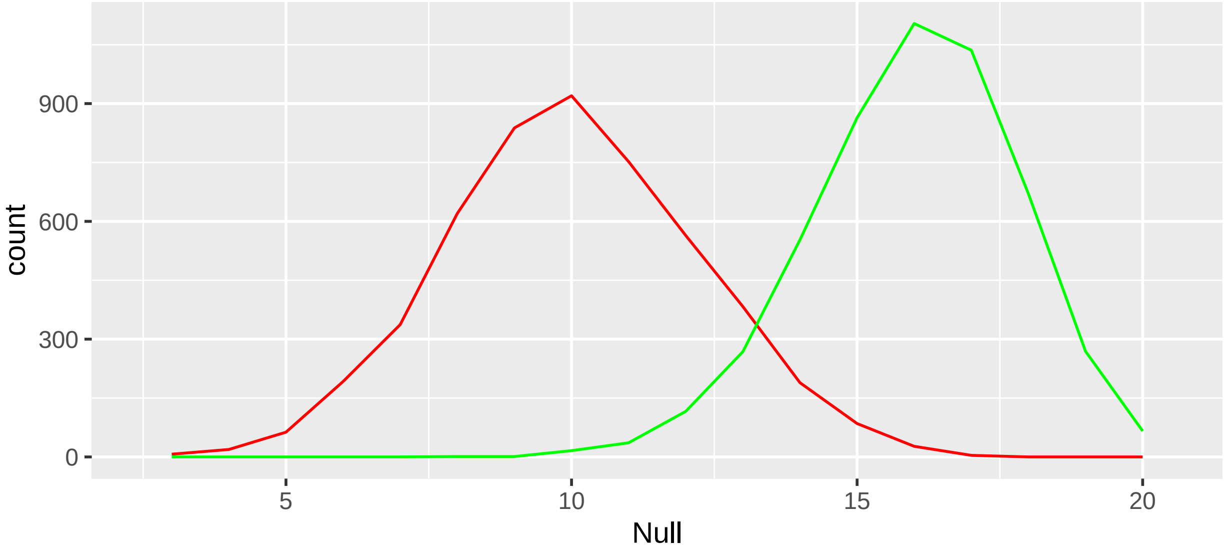
Step 1: Simulate a coin toss 20 times, count the number of heads, run the experiment 1000 times and plot of a histogram as well as store the data somewhere.

```r
Null <- rbinom(5000,20,0.5)
gf_histogram(~Null, binwidth = 1)
```



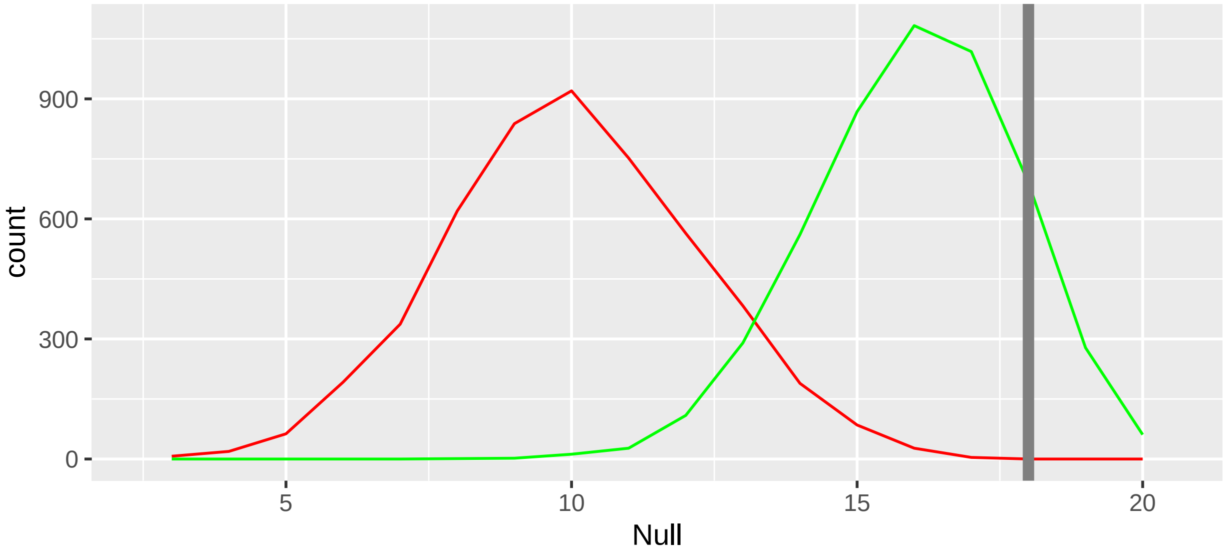
Step 2: Suppose that the alternative hypothesis is also a point hypothesis, it is that the coin will turn up heads with a probability of 0.8. Thus the alternative hypothesis looks like

```r
Alt <- rbinom(5000,20,0.8)
gf_freqpoly(~Null, binwidth = 1, color = "red") %>%
  gf_freqpoly(~Alt, binwidth = 1, color="green")
```



Step 3: Suppose in an experiment we get 18 heads out of 20. Do we have evidence to reject the null hypothesis?

```
Alt <- rbinom(5000,20,0.8)
gf_freqpoly(~Null, binwidth = 1, color = "red") %>%
  gf_freqpoly(~Alt, binwidth = 1, color="green") %>%
      gf_vline( xintercept = 18 , color = "gray50", size = 2)
```



Step 4: What is the p-value? In this calculation, we find the proportion of samples more than or equal to 18, i.e, we find the proportion of occurences that are as unlikely as 18 were the null hypothesis true.

```
p <-sum(Null>=18)/5000
```

Step 5: What is the type II error? We assume the alternative hypothesis to be true and find out the region in the altenative distribution which intersects with the accept region of the null hypothesis.

```
typeIIerror <- sum(Alt<18)/5000
power = 1-typeIIerror
```

Exercise 7: Repeat the above simulation of a test for samples from normal distribution.

References

1. Randall Pruim, Foundations and Applications of Statistics, An Introduction Using R, Second Edition, American Mathematical Society, 2018