MSMS – 301 Time Series Analysis

Practical 1

Consider the Nile Data Set in R and perform the following tasks:

1) Plot the data in R decompose the plot in different components

2) Obtain the autocorrelation and partial autocorrelation up to lag 5 and draw it.

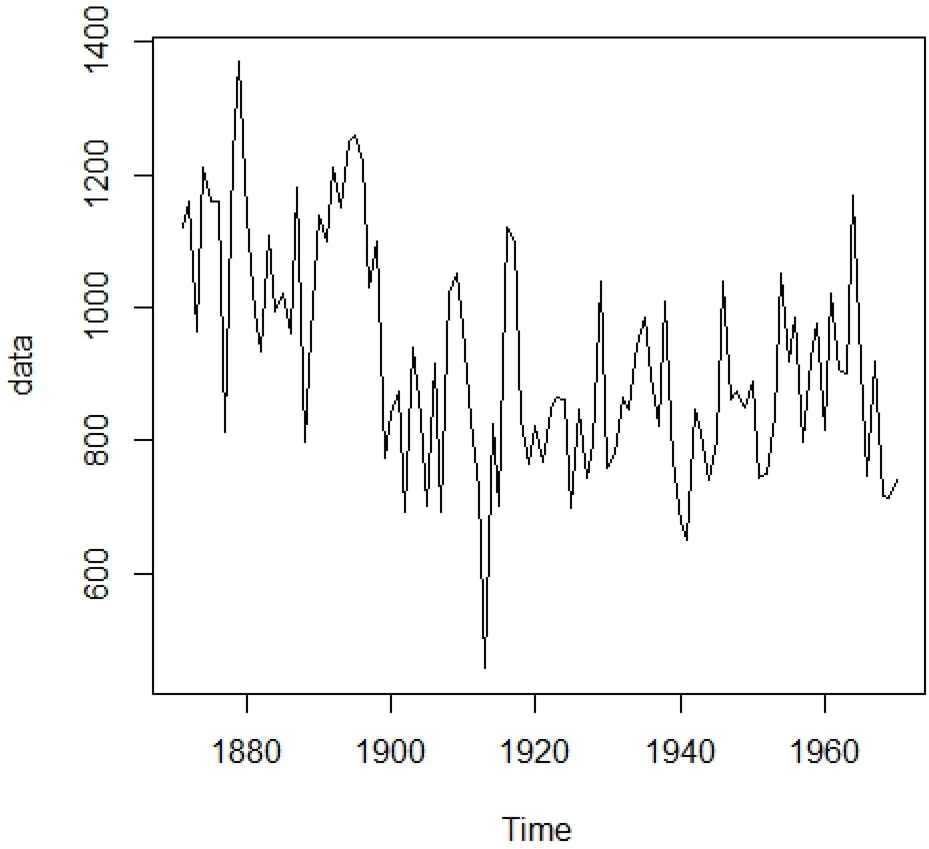
CODE:

```
library(datasets)

data=Nile

data

plot(data)

require(graphics)

data1<-ts(data,start=c(1871,1), end=c(1970,12), frequency=12)

data1

decomp<-decompose(data1) # to decompose the data into different components

plot(decomp)

Autocorr<-acf(data1,lag.max = 5,plot=TRUE)

Autocorr

Partial_Autocorr<-acf(data1,lag.max = 5,type=c("partial"),plot=TRUE)

Partial_Autocorr
```

OUTPUT:

```
> data

Time Series:

Start = 1871

End = 1970

Frequency = 1

 [1] 1120 1160  963 1210 1160 1160  813 1230 1370 1140  995  935 1110  994 1020

[16]  960 1180  799  958 1140 1100 1210 1150 1250 1260 1220 1030 1100  774  840

[31]  874  694  940  833  701  916  692 1020 1050  969  831  726  456  824  702

[46] 1120 1100  832  764  821  768  845  864  862  698  845  744  796 1040  759

[61]  781  865  845  944  984  897  822 1010  771  676  649  846  812  742  801

[76] 1040  860  874  848  890  744  749  838 1050  918  986  797  923  975  815

[91] 1020  906  901 1170  912  746  919  718  714  740
```

```
> plot(data)
```




```
> require(graphics)
> data1<-ts(data,start=c(1871,1), end=c(1970,12), frequency=12)
> data1
Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
1871 1120 1160  963 1210 1160 1160  813 1230 1370 1140  995  935
1872 1110  994 1020  960 1180  799  958 1140 1100 1210 1150 1250
1873 1260 1220 1030 1100  774  840  874  694  940  833  701  916
1874  692 1020 1050  969  831  726  456  824  702 1120 1100  832
1875  764  821  768  845  864  862  698  845  744  796 1040  759
1876  781  865  845  944  984  897  822 1010  771  676  649  846
1877  812  742  801 1040  860  874  848  890  744  749  838 1050
1878  918  986  797  923  975  815 1020  906  901 1170  912  746
1879  919  718  714  740 1120 1160  963 1210 1160 1160  813 1230
1880 1370 1140  995  935 1110  994 1020  960 1180  799  958 1140
1881 1100 1210 1150 1250 1260 1220 1030 1100  774  840  874  694
1882  940  833  701  916  692 1020 1050  969  831  726  456  824
```

1883 702 1120 1100 832 764 821 768 845 864 862 698 845

1884 744 796 1040 759 781 865 845 944 984 897 822 1010

1885 771 676 649 846 812 742 801 1040 860 874 848 890

1886 744 749 838 1050 918 986 797 923 975 815 1020 906

1887 901 1170 912 746 919 718 714 740 1120 1160 963 1210

1888 1160 1160 813 1230 1370 1140 995 935 1110 994 1020 960

1889 1180 799 958 1140 1100 1210 1150 1250 1260 1220 1030 1100

1890 774 840 874 694 940 833 701 916 692 1020 1050 969

1891 831 726 456 824 702 1120 1100 832 764 821 768 845

1892 864 862 698 845 744 796 1040 759 781 865 845 944

1893 984 897 822 1010 771 676 649 846 812 742 801 1040

1894 860 874 848 890 744 749 838 1050 918 986 797 923

1895 975 815 1020 906 901 1170 912 746 919 718 714 740

1896 1120 1160 963 1210 1160 1160 813 1230 1370 1140 995 935

1897 1110 994 1020 960 1180 799 958 1140 1100 1210 1150 1250

1898 1260 1220 1030 1100 774 840 874 694 940 833 701 916

1899 692 1020 1050 969 831 726 456 824 702 1120 1100 832

1900 764 821 768 845 864 862 698 845 744 796 1040 759

1901 781 865 845 944 984 897 822 1010 771 676 649 846

1902 812 742 801 1040 860 874 848 890 744 749 838 1050

1903 918 986 797 923 975 815 1020 906 901 1170 912 746

1904 919 718 714 740 1120 1160 963 1210 1160 1160 813 1230

1905 1370 1140 995 935 1110 994 1020 960 1180 799 958 1140

1906 1100 1210 1150 1250 1260 1220 1030 1100 774 840 874 694

1907 940 833 701 916 692 1020 1050 969 831 726 456 824

1908 702 1120 1100 832 764 821 768 845 864 862 698 845

1909 744 796 1040 759 781 865 845 944 984 897 822 1010

1910 771 676 649 846 812 742 801 1040 860 874 848 890

1911 744 749 838 1050 918 986 797 923 975 815 1020 906

1912 901 1170 912 746 919 718 714 740 1120 1160 963 1210

1913 1160 1160 813 1230 1370 1140 995 935 1110 994 1020 960

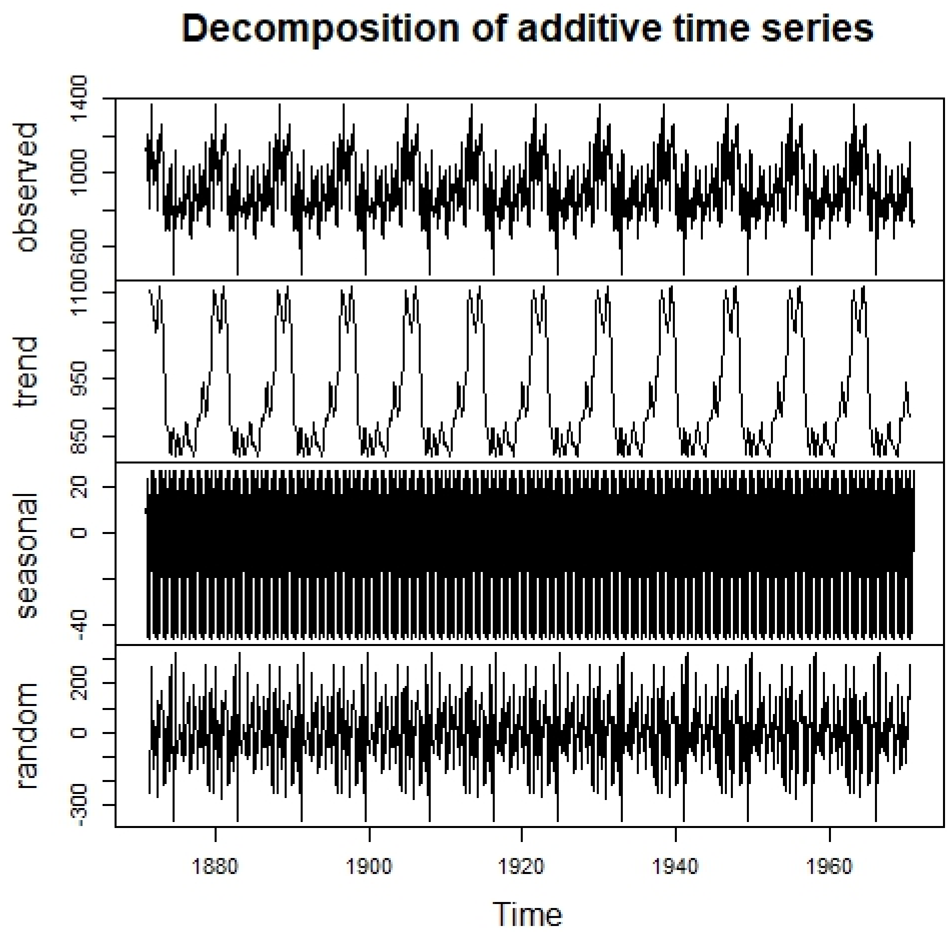
1914 1180 799 958 1140 1100 1210 1150 1250 1260 1220 1030 1100

1915 774 840 874 694 940 833 701 916 692 1020 1050 969

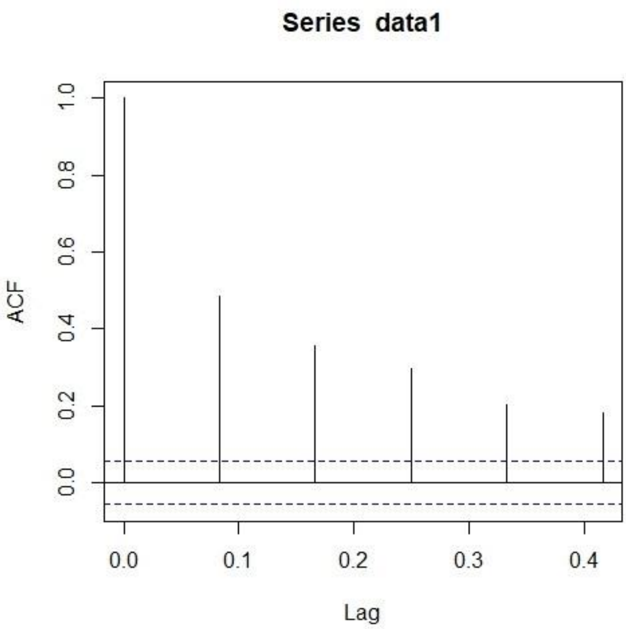
1916 831 726 456 824 702 1120 1100 832 764 821 768 845

1917	864	862	698	845	744	796	1040	759	781	865	845	944
1918	984	897	822	1010	771	676	649	846	812	742	801	1040
1919	860	874	848	890	744	749	838	1050	918	986	797	923
1920	975	815	1020	906	901	1170	912	746	919	718	714	740
1921	1120	1160	963	1210	1160	1160	813	1230	1370	1140	995	935
1922	1110	994	1020	960	1180	799	958	1140	1100	1210	1150	1250
1923	1260	1220	1030	1100	774	840	874	694	940	833	701	916
1924	692	1020	1050	969	831	726	456	824	702	1120	1100	832
1925	764	821	768	845	864	862	698	845	744	796	1040	759
1926	781	865	845	944	984	897	822	1010	771	676	649	846
1927	812	742	801	1040	860	874	848	890	744	749	838	1050
1928	918	986	797	923	975	815	1020	906	901	1170	912	746
1929	919	718	714	740	1120	1160	963	1210	1160	1160	813	1230
1930	1370	1140	995	935	1110	994	1020	960	1180	799	958	1140
1931	1100	1210	1150	1250	1260	1220	1030	1100	774	840	874	694
1932	940	833	701	916	692	1020	1050	969	831	726	456	824
1933	702	1120	1100	832	764	821	768	845	864	862	698	845
1934	744	796	1040	759	781	865	845	944	984	897	822	1010
1935	771	676	649	846	812	742	801	1040	860	874	848	890
1936	744	749	838	1050	918	986	797	923	975	815	1020	906
1937	901	1170	912	746	919	718	714	740	1120	1160	963	1210
1938	1160	1160	813	1230	1370	1140	995	935	1110	994	1020	960
1939	1180	799	958	1140	1100	1210	1150	1250	1260	1220	1030	1100
1940	774	840	874	694	940	833	701	916	692	1020	1050	969
1941	831	726	456	824	702	1120	1100	832	764	821	768	845
1942	864	862	698	845	744	796	1040	759	781	865	845	944
1943	984	897	822	1010	771	676	649	846	812	742	801	1040
1944	860	874	848	890	744	749	838	1050	918	986	797	923
1945	975	815	1020	906	901	1170	912	746	919	718	714	740
1946	1120	1160	963	1210	1160	1160	813	1230	1370	1140	995	935
1947	1110	994	1020	960	1180	799	958	1140	1100	1210	1150	1250
1948	1260	1220	1030	1100	774	840	874	694	940	833	701	916
1949	692	1020	1050	969	831	726	456	824	702	1120	1100	832
1950	764	821	768	845	864	862	698	845	744	796	1040	759

```
1951  781  865  845  944  984  897  822 1010  771  676  649  846
1952  812  742  801 1040  860  874  848  890  744  749  838 1050
1953  918  986  797  923  975  815 1020  906  901 1170  912  746
1954  919  718  714  740 1120 1160  963 1210 1160 1160  813 1230
1955 1370 1140  995  935 1110  994 1020  960 1180  799  958 1140
1956 1100 1210 1150 1250 1260 1220 1030 1100  774  840  874  694
1957  940  833  701  916  692 1020 1050  969  831  726  456  824
1958  702 1120 1100  832  764  821  768  845  864  862  698  845
1959  744  796 1040  759  781  865  845  944  984  897  822 1010
1960  771  676  649  846  812  742  801 1040  860  874  848  890
1961  744  749  838 1050  918  986  797  923  975  815 1020  906
1962  901 1170  912  746  919  718  714  740 1120 1160  963 1210
1963 1160 1160  813 1230 1370 1140  995  935 1110  994 1020  960
1964 1180  799  958 1140 1100 1210 1150 1250 1260 1220 1030 1100
1965  774  840  874  694  940  833  701  916  692 1020 1050  969
1966  831  726  456  824  702 1120 1100  832  764  821  768  845
1967  864  862  698  845  744  796 1040  759  781  865  845  944
1968  984  897  822 1010  771  676  649  846  812  742  801 1040
1969  860  874  848  890  744  749  838 1050  918  986  797  923
1970  975  815 1020  906  901 1170  912  746  919  718  714  740
> decomp<-decompose(data1) # to decompose the data into different components
> plot(decomp)
```



>Autocorr<-acf(data1,lag.max = 5,plot=TRUE)



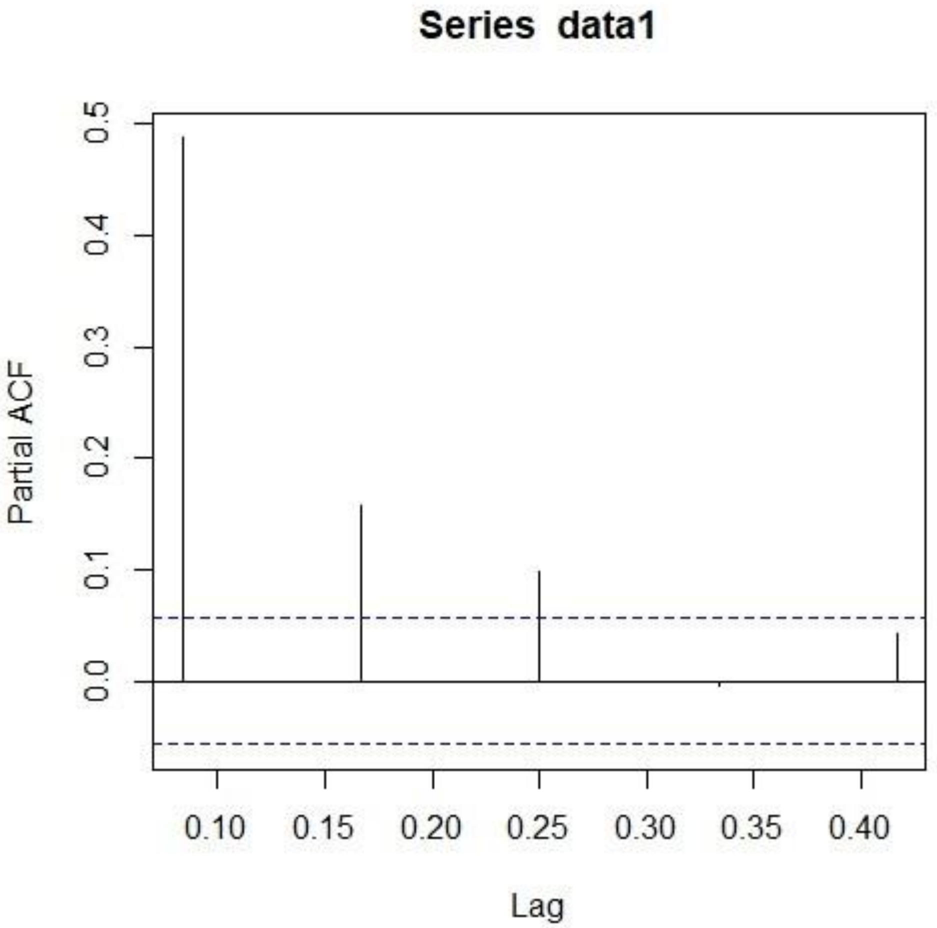
>Autocorr

Autocorrelations of series ‘data1’, by lag

0.0000 0.0833 0.1667 0.2500 0.3333 0.4167

1.000 0.487 0.357 0.296 0.204 0.181

```
>Partial_Autocorr<-acf(data1,lag.max = 5,type=c("partial"),plot=TRUE)
```




```
>Partial_Autocorr
```

Partial autocorrelations of series ‘data1’, by lag

0.0833 0.1667 0.2500 0.3333 0.4167

0.487 0.158 0.098 -0.004 0.043

Practical 2

For the given time series problem, write a R program to obtain the estimates of mean, variance, autocorrelation, partial autocorrelation upto lag 3

yt= 13, 8, 15, 4, 4, 12, 11, 7, 14, 12

CODE:

```
# Load necessary libraries
library(stats)

# Create a sample time series data (replace this with your own time series data)
# Example time series data
ts_data <- ts(c(13, 8, 15, 4, 4, 12, 11, 7, 14, 12))

# Function to calculate autocorrelation up to lag 3
autocorrelation <- function(data, lag = 3) {
  acf_result <- acf(data, lag.max = lag, plot = FALSE)$acf
  return(acf_result)
}

# Function to calculate partial autocorrelation up to lag 3
partial_autocorrelation <- function(data, lag = 3) {
pacf_result <- pacf(data, lag.max = lag, plot = FALSE)$acf
return(pacf_result)
}

# Calculate mean and variance of the time series data
mean_ts <- mean(ts_data)
variance_ts <- var(ts_data)

# Calculate autocorrelation up to lag 3
autocorr_ts <- autocorrelation(ts_data)

# Calculate partial autocorrelation up to lag 3
partial_autocorr_ts <- partial_autocorrelation(ts_data)

# Print the results
```

```
cat("Mean of the time series data:", mean_ts, "\n")
cat("Variance of the time series data:", variance_ts, "\n")
cat("Autocorrelation up to lag 3:", autocorr_ts, "\n")
cat("Partial autocorrelation up to lag 3:", partial_autocorr_ts, "\n")
```

OUTPUT:

```
# Print the results
Mean of the time series data: 10
Variance of the time series data: 16
Autocorrelation up to lag 3: 1 -0.1875 -0.2013889 0.1805556
Partial autocorrelation up to lag 3: -0.1875 -0.2451642 0.09656417
```

MSMS – 302 Statistical Machine Learning

Practical 1

YearsExperience	Salary
1.1	39343
1.3	46205
1.5	37731
2	43525
2.2	39891
2.9	56642
3	60150
3.2	54445
3.2	64445
3.7	57189
3.9	63218
4	55794
4	56957
4.1	57081
4.5	61111
4.9	67938
5.1	66029
5.3	83088
5.9	81363
6	93940
6.8	91738
7.1	98273
7.9	101302
8.2	113812
8.7	109431
9	105582
9.5	116969
9.6	112635
10.3	122391
10.5	121872

CODE:

```
# In[1]:

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# In[2]:

dataset=pd.read_csv("C:/Users /Downloads/Salary_Data.csv");dataset

# In[3]:

X=dataset.iloc[:,:-1].values

Y=dataset.iloc[:,-1].values
```

```
# In[4]:

X

# In[5]:

Y

# In[6]:

from sklearn.model_selection import train_test_split

X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=0)

# In[7]:

from sklearn.linear_model import LinearRegression

regressor=LinearRegression()

regressor.fit(X_train,Y_train)

# In[8]:

Y_pred=regressor.predict(X_test);Y_pred

# In[9]:

from sklearn.metrics import r2_score

# In[10]:

r2_score(Y_test,Y_pred)

# In[11]:

plt.scatter(X_test,Y_test,color="red")

plt.plot(X_train,regressor.predict(X_train),color="blue")

plt.title("Salary vs experience ")

plt.xlabel("Experience")

plt.ylabel("Salary")

plt.show()

# In[12]:

from sklearn.preprocessing import PolynomialFeatures

poly = PolynomialFeatures(degree=2)

X_trainpoly = poly.fit_transform(X_train)

X_testpoly=poly.fit_transform(X_test)

poly.fit(X_trainpoly, Y_train)

lin2 = LinearRegression()

lin2.fit(X_trainpoly, Y_train)

# In[13]:

Y_polypred=lin2.predict(X_testpoly);Y_polypred
```

```
# In[14]:

from sklearn.metrics import mean_squared_error

mean_squared_error(Y_polypred,Y_test)

# In[15]:

plt.scatter(X_test,Y_test,color="red")

plt.plot(X_trainpoly,lin2.predict(X_trainpoly),color="blue")

plt.title("Salary vs experience ")

plt.xlabel("Experience")

plt.ylabel("Salary")

plt.show()
```

OUTPUT:

```
# Out[4]:

array([[ 1.1],
       [ 1.3],
       [ 1.5],
       [ 2. ],
       [ 2.2],
       [ 2.9],
       [ 3. ],
       [ 3.2],
       [ 3.2],
       [ 3.7],
       [ 3.9],
       [ 4. ],
       [ 4. ],
       [ 4.1],
       [ 4.5],
       [ 4.9],
       [ 5.1],
       [ 5.3],
       [ 5.9],
       [ 6. ],
       [ 6.8],
       [ 7.1],
       [ 7.9],
       [ 8.2],
       [ 8.7],
       [ 9. ],
       [ 9.5],
       [ 9.6],
       [10.3],
       [10.5]])
# Out[5]:

array([ 39343.,  46205.,  37731.,  43525.,  39891.,  56642.,  60150.,

54445.,  64445.,  57189.,  63218.,  55794.,  56957.,  57081.,
61111.,  67938.,  66029.,  83088.,  81363.,  93940.,  91738.,
        98273., 101302., 113812., 109431., 105582., 116969., 112635.,
       122391., 121872.])
```

Out[8]:
array([40748.96184072, 122699.62295594, 64961.65717022, 63099.14214487,

115249.56285456, 107799.50275317])

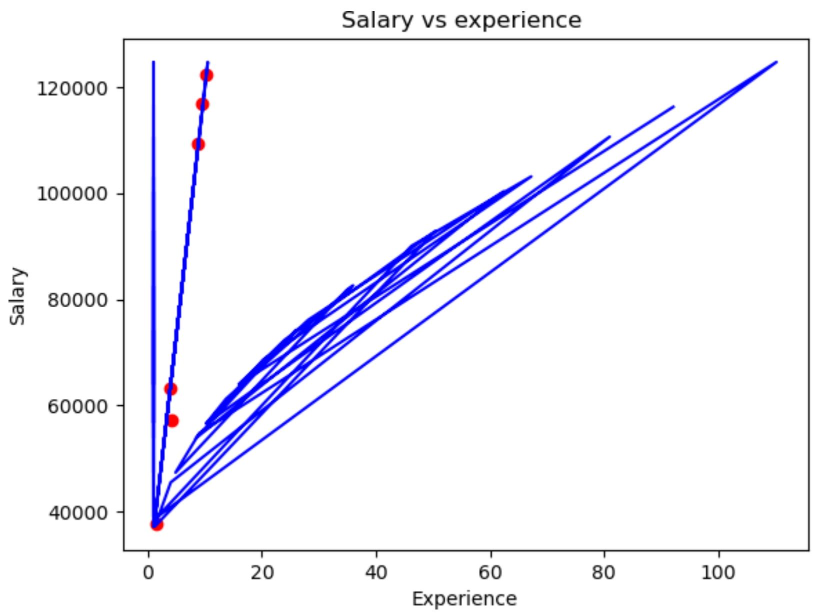
Out[10]:
0.988169515729126

Out[11]:

Out[13]:

array([40843.9467707 , 122840.44243636, 64917.43648533, 63061.08126735,
115325.5713329 , 107822.82907439])
Out[14]:

12768977.98132371
Out[15]:



Practical 2

Use atleast three different methods to address the missing values to make the data suitable for machine learning models.

import pandas as pd

data = {'Country': ['France', 'Spain', 'Germany', 'Spain', 'Germany', 'France', 'Spain', 'France', 'Germany', 'France'],

'Age': [44, 27, 30, 38, 40, 35, None, 48, 50, 37],

'Salary': [72000, 48000, 54000, 61000, None, 58000, 52000, 79000, 83000, 67000],

'Purchased': ['No', 'Yes', 'No', 'No', 'Yes', 'Yes', 'No', 'Yes', 'No', 'Yes']}

df = pd.DataFrame(data)

df2=df.copy()

df3=df.copy()

Impute missing values with median

df['Age'].fillna(df['Age'].median(), inplace=True)

df['Salary'].fillna(df['Salary'].median(), inplace=True)

print(df)

```
Country   Age   Salary Purchased
0   France  44.0  72000.0        No
1    Spain  27.0  48000.0       Yes
2  Germany  30.0  54000.0        No
3    Spain  38.0  61000.0        No
4  Germany  40.0  61000.0       Yes
5   France  35.0  58000.0       Yes
6    Spain  38.0  52000.0        No
7   France  48.0  79000.0       Yes
8  Germany  50.0  83000.0        No
9   France  37.0  67000.0       Yes
```

df2['Age'].fillna(df['Age'].min(), inplace=True)

df2['Salary'].fillna(df['Salary'].min(), inplace=True)

print(df2)

```
Country   Age   Salary Purchased
0   France  44.0  72000.0        No
1    Spain  27.0  48000.0       Yes
2  Germany  30.0  54000.0        No
3    Spain  38.0  61000.0        No
4  Germany  40.0  48000.0       Yes
5   France  35.0  58000.0       Yes
6    Spain  27.0  52000.0        No
7   France  48.0  79000.0       Yes
8  Germany  50.0  83000.0        No
```

```
9   France  37.0  67000.0       Yes
```

df3['Age'].fillna(df['Age'].max(), inplace=True)

df3['Salary'].fillna(df['Salary'].max(), inplace=True)

print(df3)

```
Country   Age   Salary Purchased
0   France  44.0  72000.0        No
1    Spain  27.0  48000.0       Yes
2  Germany  30.0  54000.0        No
3    Spain  38.0  61000.0        No
4  Germany  40.0  83000.0       Yes
5   France  35.0  58000.0       Yes
6    Spain  50.0  52000.0        No
7   France  48.0  79000.0       Yes
8  Germany  50.0  83000.0        No
9   France  37.0  67000.0       Yes
```

Practical 3:

Hosmer & Lerneshow (1989) give a dataset ("birthwt" available in R MASS library) on189 births at a US hospital, with the main interest being in low birth weight. The main variable of interest is low birth weight, a binary response variable low. You can use variable "low" as binary response variable and remining variables as regressor variable. Divide the whole dataset into training and test dataset as solve perform following task

a) Learn logistic classification model from training dataset and predict response using test dataset predictors.

b) Obtain specificity, sensitivity, positive predictive value, negative predictive value of the model using test data set.

CODE :

```
library(MASS)
df1<-data("birthwt")
write.csv(birthwt,"birthwt.csv")  ##R portion

import numpy as pd
import pandas as pd
import matplotlib.pyplot as plt

data=pd.read_csv("birthwt.csv")

X=data.drop(columns=['low'])
y=data['low']

from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.20,random_state=25)

from sklearn.preprocessing import StandardScaler
from sklearn.metrics import
make_scorer,accuracy_score,confusion_matrix,classification_report,recall_score,f1_s
core,precision_score
from sklearn.linear_model import LogisticRegression

scaler=StandardScaler()
Scaled_X_train = scaler.fit_transform(X_train)
Scaled_X_test = scaler.fit_transform(X_test)

scaler=StandardScaler()
Scaled_X_train = scaler.fit_transform(X_train)
Scaled_X_test = scaler.fit_transform(X_test)

model=LogisticRegression()
model.fit(Scaled_X_train,y_train)

y_pred=model.predict(Scaled_X_test);y_pred

tn, fn, fp, tp = confusion_matrix(y_test, y_pred).ravel()
specificity = tn / (tn+fp)
NPV = tn/ (tn + fn)
sensitivity=tp/(tp+fn)
PPV=tp/(tp+fp)
```

```
print(specificity,NPV,sensitivity,PPV)
```

OUTPUT :

```
0.9333333333333333 1.0 1.0 0.8
```

Practical 4:

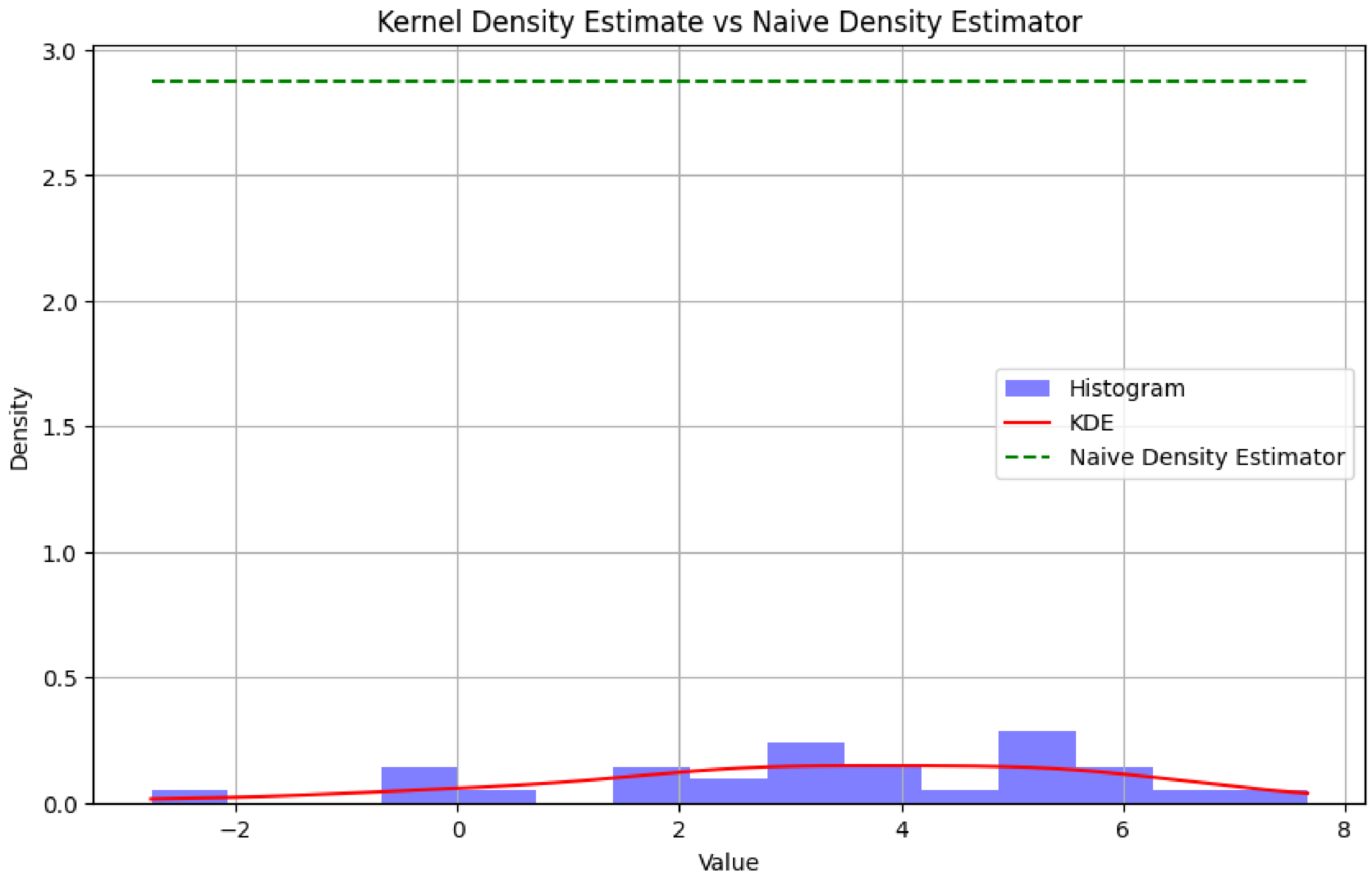
For following dataset, obtain kernel density estimate and Naive density estimator. Also plot both the estimator. 5.65746599 5.38283914 2.79892121 2.85423660 2.95252721 5.42626667 7.66239113 -0.18001073 0.65083500 2.40276530 -0.09929884 6.32619215 5.03650752 2.07470777 1.78019174 6.12891558 4.05352439 2.02686971 3.50834853 -2.76449768 4.98428763 3.01292677 2.82448038 3.98110437 5.09371862 5.97961648 4.56968496 -0.48814532 5.08736697 2.4175760

CODE :

```
import numpy as np

import matplotlib.pyplot as plt

from scipy.stats import gaussian_kde


# Given dataset

data = np.array([5.65746599, 5.38283914, 2.79892121, 2.85423660, 2.95252721, 5.42626667,

        7.66239113, -0.18001073, 0.65083500, 2.40276530, -0.09929884, 6.32619215,

        5.03650752, 2.07470777, 1.78019174, 6.12891558, 4.05352439, 2.02686971,

        3.50834853, -2.76449768, 4.98428763, 3.01292677, 2.82448038, 3.98110437,

        5.09371862, 5.97961648, 4.56968496, -0.48814532, 5.08736697, 2.41757609])


# Kernel Density Estimate (KDE)

kde = gaussian_kde(data)

kde_x = np.linspace(min(data), max(data), 1000)

kde_y = kde(kde_x)


# Naive Density Estimator

naive_density = len(data) / (max(data) - min(data))

naive_y = np.full_like(kde_x, naive_density)


# Plotting
```

```python
plt.figure(figsize=(10, 6))

plt.hist(data, bins=15, density=True, alpha=0.5, color='blue', label='Histogram')

plt.plot(kde_x, kde_y, color='red', label='KDE')

plt.plot(kde_x, naive_y, linestyle='--', color='green', label='Naive Density Estimator')

plt.title('Kernel Density Estimate vs Naive Density Estimator')

plt.xlabel('Value')

plt.ylabel('Density')

plt.legend()

plt.grid(True)

plt.show()
```

OUTPUT :



MSMS – 303 Multivariate Analysis

Practical 1

Find MLE of Σ , μ *and* ρ for the data given in table and also find the result given below.

Head Length, First Son (x_1)	Head Breadth, First Son (x_2)	Head Length, Second Son (x_3)	Head Breadth, Second Son (x_4)
191	155	179	145
195	149	201	152
181	148	185	149
183	153	188	149
176	144	171	142
208	157	192	152
189	150	190	149
197	159	189	152
188	152	197	159
192	150	187	151
179	158	186	148
183	147	174	147
174	150	185	152
190	159	195	157
188	151	187	158
163	137	161	130
195	155	183	158
186	153	173	148
181	145	182	146
175	140	165	137
192	154	185	152
174	143	178	147
176	139	176	143
197	167	200	158
190	163	187	150

A) **Find the estimates of parameters of conditional distribution of (x_3, x_4) given (x_1, x_2) i.e. find $S_{21}{S_{11}}^{-1}$ *and* $S_{22.1} = S_{22} - S_{21}{S_{11}}^{-1}S_{12}$**

B) **Find the partial correlation $r_{34.12}$**

C) **Use Fisher's Z to find a confidence interval for $\rho_{34.12}$ with confidence 0.95**

D) **Find the sample multiple correlation coefficients between x_3 and (x_1, x_2) and between x_4 and (x_1, x_2)**

E) **Test the hypothesis that x_3 is independent of (x_1, x_2) and x_4 is independent of (x_1, x_2)**

CODE:

```
X1<-
c(191,195,181,183,176,208,189,197,188,192,179,183,174,190,188,163,195,186,181,175,192,174,176,197,190);X1

X2<-
c(155,149,148,153,144,157,150,159,152,150,158,147,150,159,151,137,155,153,145,140,154,143,139,167,163);X2

X3<-
c(179,201,185,188,171,192,190,189,197,187,186,174,185,195,187,161,183,173,182,165,185,178,176,200,187);X3
```

```
X4<-
c(145,152,149,149,142,152,149,152,159,151,148,147,152,157,158,130,158,148,146,137,152,147,143,158,150);X4

X<-matrix(c(X1,X2,X3,X4),ncol = 4);X

X_mean<-colMeans(X);X_mean

sigma<-
matrix(c(var(X1),cov(X1,X2),cov(X1,X3),cov(X1,X4),cov(X2,X1),var(X2),cov(X2,X3),cov(X2,X4),cov(X3,X1),cov(X3,X2),va
r(X3),cov(X3,X4),cov(X4,X1),cov(X4,X2),cov(X4,X3),var(X4)),nrow = 4,ncol = 4,byrow = T);sigma

rho<-
matrix(c(cor(X1,X1),cor(X1,X2),cor(X1,X3),cor(X1,X4),cor(X2,X1),cor(X2,X2),cor(X2,X3),cor(X2,X4),cor(X3,X1),cor(X3,X
2),cor(X3,X3),cor(X3,X4),cor(X4,X1),cor(X4,X2),cor(X4,X3),cor(X4,X4)),nrow = 4,ncol = 4,byrow = T);rho

sigma11<-matrix(c(var(X3),cov(X3,X4),cov(X4,X3),var(X4)),nrow = 2,byrow = T);sigma11

sigma22<-matrix(c(var(X1),cov(X1,X2),cov(X1,X2),var(X2)),nrow = 2,byrow = T);sigma22

sigma12<-matrix(c(cov(X1,X3),cov(X1,X4),cov(X2,X3),cov(X2,X4)),nrow = 2,byrow = T);sigma12

sigma21<-t(sigma12);sigma21

m2<-matrix(c(mean(X1),mean(X2)),nrow = 2);m2

m1<-matrix(c(mean(X3),mean(X4)),nrow = 2);m1

miu<-m1-sigma12%*%(solve(sigma22))%*%m2;miu

sigma<-sigma11-((sigma12%*%solve(sigma22))%*%sigma21);sigma

s11<-cov(X3,X3);s11

s12<-cov(X3,X4);s12

s13<-matrix(c(cov(X3,X1),cov(X3,X2)),nrow = 2);s13

s33<-matrix(c(var(X1),cov(X1,X2),cov(X2,X1),var(X2)),nrow = 2);s33

s23<-matrix(c(cov(X4,X1),cov(X4,X2)),nrow = 2);s23

s22<-cov(X4,X4);s22

numerator<-s12-(t(s13)%*%solve(s33)%*%s23);numerator

denominator<-((s11-(t(s13)%*%(solve(s33))%*%s13))%*%(s22-(t(s23)%*%(solve(s33))%*%s23)))^0.5

partial<-numerator/denominator;partial

n=length(X1)

p=4

a12<-matrix(c(cov(X3,X1),cov(X3,X2)),nrow=2)

a22<-matrix(c(var(X1),cov(X1,X2),cov(X2,X1),var(X2)),nrow = 2,byrow = T)

a11<-var(X3)

r1<-((t(a12)%*%solve(a22))%*%a12)/a11;r1

c11<-var(X4)

c12<-matrix(c(cov(X4,X1),cov(X4,X2)),nrow = 2)

c22<-matrix(c(var(X1),cov(X1,X2),cov(X2,X1),var(X2)),nrow = 2,byrow = T)
```

```
r2<-((t(c12)%*%solve(c22))%*%c12)/c11;r2

x1<-((r1^2)*(n-p))/((1-r1^2)*(p-1));x1

x2<-((r2^2)*(n-p))/((1-r2^2)*(p-1));x2

qf(p=0.05,df1=(p-1),df2=(n-p),lower.tail=F)
```

OUTPUT:

```
>X_mean<-colMeans(X);X_mean

[1] 185.72 151.12 183.84 149.24

> sigma<-
matrix(c(var(X1),cov(X1,X2),cov(X1,X3),cov(X1,X4),cov(X2,X1),var(X2),cov(X2,X3),cov(X2,X4),cov(X3,X1),cov(X3,X2),var(X3),cov(X3,X4),cov(X4,X1),cov(X4,X2),cov(X4,X3),var(X4)),nrow = 4,ncol = 4,byrow = T);sigma

         [,1]     [,2]      [,3]     [,4]

[1,] 95.29333 52.86833  69.66167 46.11167

[2,] 52.86833 54.36000  51.31167 35.05333

[3,] 69.66167 51.31167 100.80667 56.54000

[4,] 46.11167 35.05333  56.54000 45.02333

> rho<-
matrix(c(cor(X1,X1),cor(X1,X2),cor(X1,X3),cor(X1,X4),cor(X2,X1),cor(X2,X2),cor(X2,X3),cor(X2,X4),cor(X3,X1),cor(X3,X2),cor(X3,X3),cor(X3,X4),cor(X4,X1),cor(X4,X2),cor(X4,X3),cor(X4,X4)),nrow = 4,ncol = 4,byrow = T);rho

          [,1]      [,2]      [,3]      [,4]

[1,] 1.0000000 0.7345555 0.7107518 0.7039807

[2,] 0.7345555 1.0000000 0.6931573 0.7085504

[3,] 0.7107518 0.6931573 1.0000000 0.8392519

[4,] 0.7039807 0.7085504 0.8392519 1.0000000

> sigma11<-matrix(c(var(X3),cov(X3,X4),cov(X4,X3),var(X4)),nrow = 2,byrow = T);sigma11

         [,1]     [,2]

[1,] 100.8067 56.54000

[2,]  56.5400 45.02333

> sigma22<-matrix(c(var(X1),cov(X1,X2),cov(X1,X2),var(X2)),nrow = 2,byrow = T);sigma22

         [,1]     [,2]

[1,] 95.29333 52.86833

[2,] 52.86833 54.36000

> sigma12<-matrix(c(cov(X1,X3),cov(X1,X4),cov(X2,X3),cov(X2,X4)),nrow = 2,byrow = T);sigma12

         [,1]     [,2]

[1,] 69.66167 46.11167
```

```
[2,] 51.31167 35.05333
> sigma21<-t(sigma12);sigma21
         [,1]     [,2]
[1,] 69.66167 51.31167
[2,] 46.11167 35.05333
> m2<-matrix(c(mean(X1),mean(X2)),nrow = 2);m2
       [,1]
[1,] 185.72
[2,] 151.12
> m1<-matrix(c(mean(X3),mean(X4)),nrow = 2);m1
       [,1]
[1,] 183.84
[2,] 149.24
>miu<-m1-sigma12%*%(solve(sigma22))%*%m2;miu
         [,1]
[1,] 33.73574
[2,] 36.58502
> sigma<-sigma11-((sigma12%*%solve(sigma22))%*%sigma21);sigma
         [,1]     [,2]
[1,] 47.65667 17.06605
[2,] 17.06605 15.66111
> s11<-cov(X3,X3);s11
[1] 100.8067
> s12<-cov(X3,X4);s12
[1] 56.54
> s13<-matrix(c(cov(X3,X1),cov(X3,X2)),nrow = 2);s13
         [,1]
[1,] 69.66167
[2,] 51.31167
> s33<-matrix(c(var(X1),cov(X1,X2),cov(X2,X1),var(X2)),nrow = 2);s33
         [,1]     [,2]
[1,] 95.29333 52.86833
[2,] 52.86833 54.36000
> s23<-matrix(c(cov(X4,X1),cov(X4,X2)),nrow = 2);s23
```

```
     [,1]
[1,] 46.11167
[2,] 35.05333
> s22<-cov(X4,X4);s22
[1] 45.02333
> numerator<-s12-(t(s13)%*%solve(s33)%*%s23);numerator
     [,1]
[1,] 18.03943
> denominator<-((s11-(t(s13)%*%(solve(s33))%*%s13))%*%(s22-(t(s23)%*%(solve(s33))%*%s23)))^0.5
> partial<-numerator/denominator;partial
     [,1]
[1,] 0.625582
> n=length(X1)
> p=4
> a12<-matrix(c(cov(X3,X1),cov(X3,X2)),nrow=2)
> a22<-matrix(c(var(X1),cov(X1,X2),cov(X2,X1),var(X2)),nrow = 2,byrow = T)
> a11<-var(X3)
> r1<-((t(a12)%*%solve(a22))%*%a12)/a11;r1
      [,1]
[1,] 0.5687288
> c11<-var(X4)
> c12<-matrix(c(cov(X4,X1),cov(X4,X2)),nrow = 2)
> c22<-matrix(c(var(X1),cov(X1,X2),cov(X2,X1),var(X2)),nrow = 2,byrow = T)
> r2<-((t(c12)%*%solve(c22))%*%c12)/c11;r2
     [,1]
[1,] 0.575185
> x1<-((r1^2)*(n-p))/((1-r1^2)*(p-1));x1
    [,1]
[1,] 3.34665
> x2<-((r2^2)*(n-p))/((1-r2^2)*(p-1));x2
     [,1]
[1,] 3.460842
>qf(p=0.05,df1=(p-1),df2=(n-p),lower.tail=F)
[1] 3.072467
```

MSMS – 304 Biostatistics

1. **Imagine that the incidence of gun violence is compared in two cities, one with relaxed gun laws (A), the other with strict gun laws (B). In the city with relaxed gun laws, there were 50 shootings in a population of 100,000 and in the other city, 10 shootings in a population of 100,000.**

   (a) What is the relative risk of gun violence in the city with relaxed gun laws (A)?

   (b) What is the relative risk of gun violence in the city with strict gun laws (B)?

   (c) What questions need to be asked before concluding that there is an association between shootings and gun laws?

   Solution:

   (a) The relative risk of gun violence in the city with relaxed gun laws (A) is:

$$\frac{indidence\ in\ A}{indidence\ in\ B} = \frac{\frac{50}{100{,}000}}{\frac{10}{100{,}000}} = \frac{50}{10} = 5$$

   (b) The relative risk of gun violence in the city with strict gun laws (B) is:

$$\frac{incidence\ in\ B}{incidence\ in\ A} = \frac{10/1000000}{50/1000000} = \frac{10}{50} = 0.50$$

   (c) The seemingly obvious conclusion is that the relaxed gun laws in city. A cause more gun violence, quintupling the risk. However, before jumping to conclusions, it may be helpful to consider the following questions:

   - Is the age distribution and socioeconomic status of each Population similar? Younger people involved in gangs or individuals of low socio economic status, may more likely to resort to gun violence. City A may be more prone to such situations.
   - Were the risk exposure patterns several decades ago, when the laws were first induced, similar to those in the present? "Are the judicial systems and records of gun violenes, different in each city?

2. **A study looking at breast cancer in women compared cases with non-cases, and found that 75/100 cases did not use calcium supplements compared with 25/100 of the non-cases.**

   (a) Develop a table to display the data.

   (b) Calculate the odds of exposure in cases and non-cases.

   (c) Calculate the odds ratio using the cross-product ratio

   (d) How does the difference between the two prevalence of breast cancer (75% vs 25%) compare to the odds ratio?

   Solution:

   a) a table to display the data is given below:

Risk factor/exposure	Disease Group	
	Case	Control
No calcium supplement	75(a)	25(6)
Calcium Supplement	25(c)	75(d)

b) by the odds of exposure in case group:

$$\frac{a}{c} = \frac{75}{25} = 3$$

by the odds of exposure in control group:

$$\frac{b}{d} = \frac{25}{75} = \frac{1}{3}$$

c) The odds ratio using cross product:

$$\frac{a}{b} \times \frac{d}{c} = \frac{75 \times 75}{25 \times 25} = 9$$

d) After calculating the odds ratio, we observe a 3-Fold differences in the prevalence rate (75% vs 25%) change to a 9 - Fold differences in the odds ratio. Clearly, the two methods produce opposing results.

3. Let us consider the relationship between smoking and lung cancer. Suppose exposure to cigarette smoke increases the incidence of lung cancer by 20% (i.e. the relative risk is 1.2). Lung cancer has a base line incidence of 3% per year (in the non- exposed group). Suppose as well that baseline incidence in obese individuals is 1/3 less (i.e. 1%/yr.), and the relative risk associated with the exposure is also 1.2. You follow up 1000 non-obese and 1000 obese subjects with the exposure, and an equivalent number without the exposure, The study lasts 25 years. Work with 25-year cumulative incidence and a denominator of 1000.

(a) Create a table to show the data for obese and non-obese subjects.

(b) Calculate the odds ratio of disease in the exposed group in relation to those who are not exposed.

(c) Compare the odds ratio with the relative risk of 1.2.

Solution:

(a) Data on exposure in those who are and are not obese; annual disease incidence at baseline = 3% and RR = 1.2 (25-year follow up)

	Not Obese		Obese	
	Diseased	Not Diseased	Diseased	Not Diseased
Exposed	900	100	300	700
Not Exposed	750	250	250	750

(b) Relative Risk and Odds Ratio for the non-obese:

$$Relative\ Risk = \frac{Exposed\ Rate}{Not\ Exposed\ Rate} = \frac{900/1000}{750/1000} = 1.20$$

$$Odds\ Ratio = \frac{900 \times 250}{100 \times 750} = 3$$

Relative Risk and Odds Ratio for the obese:

$$Relative\ Risk = \frac{Exposed\ Rate}{Not\ Exposed\ Rate} = \frac{300/1000}{250/1000} = 1.20$$

$$Odds\ Ratio = \frac{300 \times 700}{250 \times 750} = 1.29$$

(c) Overall, we can see that decreasing the baseline incidence will decrease the odd ratio (3.00 in those who are non-obese versus 1.23 in these who are obese). Obviously, these results run counter to expected results, putting the onus on the researcher to justify them. Similarly, you should find that increasing the incidence will increase the odds ratio.

From the data in the previous chart, we can also calculate the relative risk for a lack of disease in non-obese individuals:

$$Relative\ Risk: \frac{(100/1000)}{(250/1000)} = 0.40$$

Finally using the data in the previous chart, we can calculate the Odds ratio for a lack of disease. In non-obese Individuals by use of the cross-product ratio:

$$Odds\ Ratio = \frac{100 \times 750}{250 \times\ 900} = \ 0.33$$

Consider that the odds ratio for a lack of disease in non-obese Individuals (0.333) is equivalent to the reciprocal of the odds ratio for the presence of disease is non-obese individuals (3.00. as Calculated in the previous example). The advantageous properly holds for all odds ratios.

Note both relative risk and the odds ratio are only sensical in well-executed studies which are able to be related to the population from which you wish to draw associations.

4. **Use the following table to calculate the attributable risk associated with taking a supplement containing folate during pregnancy:**

	Annual Death Rates per 100 000	
	Neural Tube Defects	**Premature Births**
No Folate	**631**	**727**
Folate	**24**	**563**

Solution:
Excess risk for no folate supplementation on Neural Tube Defects (NTD):

$$631 - 24 = 607$$

Excess risk for no folate supplementation on Premature Births:

$$787 - 563 = 164$$

As we wish to express attributable risk as a percentage, perform the following:
Attributable risk for no folate supplementation on Neural Tube Defects:

$$\frac{607}{631} \times 100\% = 96.2\%$$

Attributable risk for no folate supplementation on Premature births:

$$\frac{164}{727} \times 100\% = 22.6\%$$

So, we claim of pregnant women not consuming folate, 96.2% of neural tube defect cases can be attributed to a lack of folate supplementation. Therefore, if the cause were to be removed, the disease could be reduced by up to 96.2% and 607 lives could be saved. Similarly, the attributable risk for premature births is 22.6%.

MSMS – 306 Lifetime data Analysis

Practical 1

The recorded death times of 15 patients were 7.35, 8.69, 8.80, 9.63, 9.63, 9.89, 9.98, 10.24, 10,36, 10.37, 10.48, 11.33, 11.39, 12.02 and 13.12 days, 10 patients whose are alive were removed from the test at 20 days. Suppose recorded time follows Weibull distribution, then

- **a. Find maximum likelihood estimates of parameter.**
- **b. Using estimates of part 1 draw survival and hazard rate curve.**
- **c. Comment on behaviour of hazard rate.**

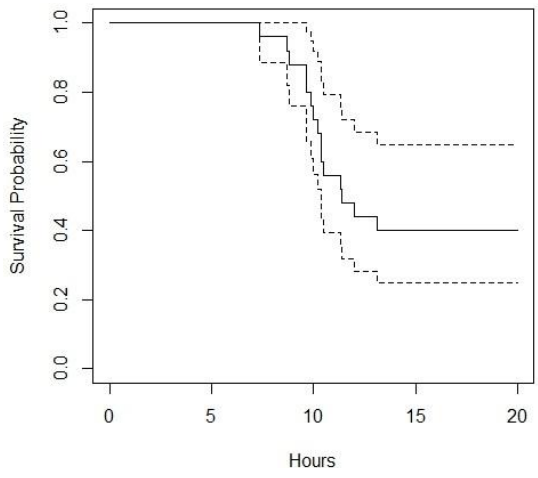
CODE:

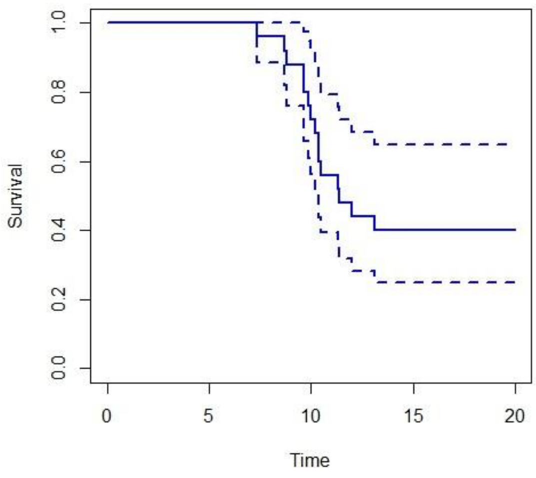
```
require(survival)

#Loading required package: survival

failures=c(7.35,8.69,8.80,9.63,9.63,9.89,9.98,10.24,10.36,10.37,10.48,11.33,11.39,12.02,13.12)

y=Surv(c(failures,rep(20,10)),c(rep(1,length(failures)),rep(0,10)))

yw=survreg(y~1,dist="weibull")

summary(yw)

etaHAT=exp(coefficients(yw)[1])

betaHAT=1/yw$scale

signif(c(eta=etaHAT,beta=betaHAT),6)

ys=survfit(y~1,type="kaplan-meier")

summary(ys)

plot(ys,xlab="Hours",ylab="Survival  Probability")

plot(ys, lwd=2, xlab="Time", ylab="Survival", col="blue")

lines(survfit(y~1, type='kaplan-meier'),fun="event", lwd = 2, col = "red", type = "s")

legend("topright", legend=c("Survival", "Hazard"), col=c("blue", "red"), lty=1)
```

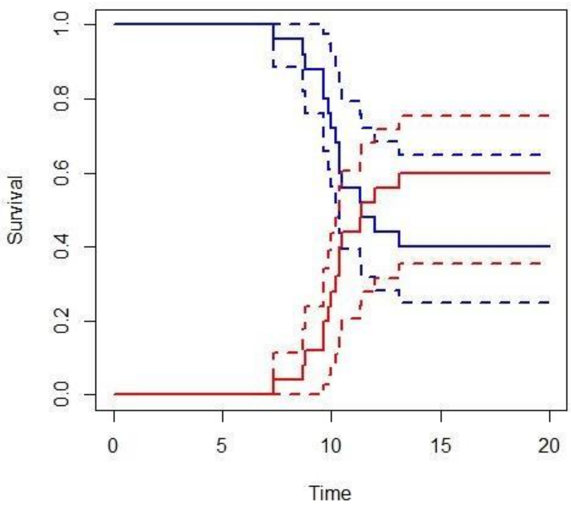
OUTPUT:

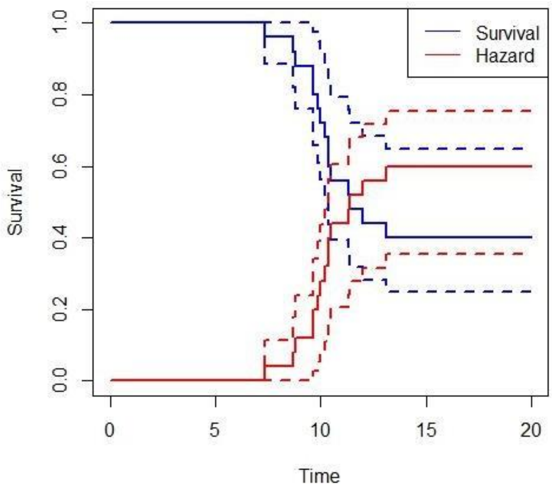
```
> summary(yw)


Call:

survreg(formula = y ~ 1, dist = "weibull")

            Value Std. Error     z       p

(Intercept)  2.960      0.133 22.33 <2e-16

Log(scale)  -0.698      0.221 -3.15 0.0016
```

```
Scale= 0.498

Weibull distribution
Loglik(model)= -58.7 Loglik(intercept only)= -58.7
Number of Newton-Raphson Iterations: 4
n= 25
>signif(c(eta=etaHAT,beta=betaHAT),6)
eta.(Intercept)           beta
       19.29780        2.00979
> summary(ys)
Call: survfit(formula = y ~ 1, type = "kaplan-meier")

  time n.riskn.event survival std.err lower 95% CI upper 95% CI
  7.35     25       1     0.96  0.0392        0.886        1.000
  8.69     24       1     0.92  0.0543        0.820        1.000
  8.80     23       1     0.88  0.0650        0.761        1.000
  9.63     22       2     0.80  0.0800        0.658        0.973
  9.89     20       1     0.76  0.0854        0.610        0.947
  9.98     19       1     0.72  0.0898        0.564        0.919
 10.24     18       1     0.68  0.0933        0.520        0.890
 10.36     17       1     0.64  0.0960        0.477        0.859
 10.37     16       1     0.60  0.0980        0.436        0.826
 10.48     15       1     0.56  0.0993        0.396        0.793
 11.33     14       1     0.52  0.0999        0.357        0.758
 11.39     13       1     0.48  0.0999        0.319        0.722
 12.02     12       1     0.44  0.0993        0.283        0.685
 13.12     11       1     0.40  0.0980        0.247        0.64
```









Practical 2

Generate 100 observations from Weibull distribution with shape parameter 3 and scale parameter 10. Hence obtain the ML estimation of its parameters. Also draw the two dimensional likelihood plot of Weibull model for the given dataset. Finally obtain the ML estimation of mean failure time and compare it with sample mean.

CODE:

```
install.packages('ggdist')

install.packages("fitdistrplus")

# Generate 100 observations from a Weibull distribution with shape parameter 3 and scale parameter 10

set.seed(123)

x <- rweibull(100, shape = 3, scale = 10)
```

```
# Maximum likelihood estimation of parameters
library(fitdistrplus)
weibull_fit<- fitdist(x, "weibull")
weibull_ml_est<- coef(weibull_fit)
weibull_ml_est
# Two-dimensional likelihood plot
library(MASS)
library(ggplot2)
library(ggdist)
ggdist(x,fit = weibull_fit, type = "density") + stat_density_2d(aes(fill = ..density..), alpha = 0.5, contour = F) + scale_fill_gradient(low = "white", high = "blue") + ggtitle("Two-dimensional Likelihood Plot")
# ML estimate of Mean failure time
mean_failure_time_ml<- weibull_ml_est["scale"] / gamma(1 + 1 / weibull_ml_est["shape"])
mean_failure_time_ml
# Sample mean
mean_failure_time_sample<- mean(x)
mean_failure_time_sample
# Compare ML estimate and sample mean
mean_failure_time_ml - mean_failure_time_sample
```

OUTPUT:

```
>weibull_ml_est
  shape  scale
3.060734 9.998053
>mean_failure_time_ml
  scale
11.18624
>mean_failure_time_sample
[1] 8.938964
>mean_failure_time_ml - mean_failure_time_sample
  scale
2.247278
legend("topright", legend=c("Survival", "Hazard"), col=c("blue", "red"), lty=1)
```

Practical 3

Fifty leukaemia patients were subjected to a test and the test is terminated when 35 patients were failed. Their lifetimes (in weeks) are given below:

22.3 26.8 30.3 31.9 32.1 33.3 33.7 33.9 34.7 36.1 36.4 36.5 36.6
37.1 37.6 38.2 38.5 38.7 38.7 38.9 38.9 39.1 41.1 41.1 41.4 42.4
43.6 43.8 44.0 45.3 45.8 50.4 51.3 51.4 51.5

Assume lifetimes follow lognormal distribution and estimate the two parameters of the distribution. Also estimate mean time to failure and median time to failure. Draw survival and hazard curve.

CODE:

```
install.packages("EnvStats")

library(EnvStats)

x=c(22.3,26.8,30.3,31.9,32.1,33.3,33.7,33.9,34.7,36.1,36.4,36.5,36.6,37.1,37.6,38.2,38.5,38.7,38.7,38.9,38.9,39.1,41.1,41.1,41.4,42.4,43.6,43.8,44.0,45.3,45.8,50.4,51.3,51.4,51.5)

elnormAlt(x, method = "mle") $distribution

elnormAlt(x, method = "mle")$sample.size

elnormAlt(x, method = "mle")$parameters

elnormAlt(x, method = "mle")$n.param.est

elnormAlt(x, method = "mle")$method

elnormAlt(x, method = "mle")$data.name

elnormAlt(x, method = "mle")$bad.obs

attr(elnormAlt(x, method = "mle"),"class")

mttf=log(38.9759037)-1/2*log(0.1777356/38.9759037+1) ;mttf

mediananttf=exp(mttf);mediananttf
```

OUTPUT:

```
>elnormAlt(x, method = "mle") $distribution

[1] "Lognormal"

>elnormAlt(x, method = "mle")$sample.size

[1] 35

>elnormAlt(x, method = "mle")$parameters

    mean        cv

38.9759037  0.1777356

>elnormAlt(x, method = "mle")$n.param.est

[1] 2

>elnormAlt(x, method = "mle")$method

[1] "mle"

>elnormAlt(x, method = "mle")$data.name
```

```
[1] "x"

>elnormAlt(x, method = "mle")$bad.obs

[1] 0

>attr(elnormAlt(x, method = "mle"),"class")

[1] "estimate"

>mttf=log(38.9759037)-1/2*log(0.1777356/38.9759037+1) ;mttf

[1] 3.660669

>medianttf=exp(mttf);medianttf

[1] 38.88734
```