######################### Bike Sharing Problem #######################################

[Description:](https://www.kaggle.com/c/bike-sharing-demand/forums/t/13228/kaggle-scripts/69563#post69563)

Bike sharing systems are a means of renting bicycles where the process of obtaining membership, rental, and bike return is automated via a network of kiosk locations throughout a city. Using these systems, people are able rent a bike from a one location and return it to a different place on an as-needed basis. Currently, there are over 500 bike-sharing programs around the world.

The data generated by these systems makes them attractive for researchers because the duration of travel, departure location, arrival location, and time elapsed is explicitly recorded. Bike sharing systems therefore function as a sensor network, which can be used for studying mobility in a city. In this competition, participants are asked to combine historical usage patterns with weather data in order to forecast bike rental demand in the Capital Bikeshare program in Washington, D.C

Data: Saved…

# Bike sharing problem

###########################################################################################

###########################################################################################

#1. Check current directory

getwd()

#2. Set current directory as the folder where Bike Sharing data is downloaded

setwd("C:\\Users\\Arpit\\Desktop\\Acadgild")

###########################################################################################

###########################################################################################

#3. import train data

train<-read.csv("train.csv")

#4.Import test data

test<-read.csv("test.csv")

#5. Check structure & dimention of train data

str(train)

dim(train)

#6. Check structure of test data

str(test)

dim(test)

#7. Check the summary of train data

summary(train)

#8. Check the summary of test data

summary(test)

###########################################################################################

###########################################################################################

#Combine both train and test data to undetstand the distribution of independent variable

#9. Create new blank columns in test data which are not present there (casual, registered, count)

test$casual<-0

test$registered<-0

test$count<-0

#10. Check before combining train and test data set whether the structure is similar for both.

str(train)

str(test)

#11. Create a new dataframe by combining train and test dataset -"BikeData"

BikeData<-rbind(train,test)

###########################################################################################

###########################################################################################

#12. Check the structure of BikeData

str(BikeData)

dim(BikeData)

#13. Check the summary of BikeData

summary(BikeData)

###########################################################################################

###########################################################################################

#14. Find Missing value (if any)

table(is.na(BikeData))

sapply(BikeData,function(train)sum(train==""))

###########################################################################################

###########################################################################################

#Uni-Variate Analysis.

#15. Create histigram chart for all the numeric columns

hist(BikeData$season) #season should be a factor

hist(BikeData$holiday) #holiday should a factor, there are very few holiday

hist(BikeData$workingday) #workingday should be a factor

hist(BikeData$weather)#wearther should be a factor

str(BikeData)

hist(BikeData$windspeed)

hist(BikeData$humidity)

hist(BikeData$atemp)

#16. Change the layout of plot using par function. Then again plot the histograms

par(mfrow=c(2,2))

#17. Write insights as comment for each of the histogram

###########################################################################################

###########################################################################################

#Uni-Variate Analysis - Categorical

#Convert discrete variable into factor (season, weather, holiday, working day)

BikeData$season = as.factor(BikeData$season)

BikeData$workingday =as.factor(BikeData$workingday)

BikeData$holiday=as.factor(BikeData$holiday)

BikeData$weather=as.factor(BikeData$weather)

#Create table for all the factors, return the count

table(BikeData$season)

table(BikeData$workingday)

#Create proportion or % for all the factors and write insight as comment

prop.table(table(BikeData$workingday))

barplot(table(BikeData$season),main = "season",xlab = "no. of seasons",ylab = "no. of users")

barplot(table(BikeData$workingday),main = "workingday",xlab = "no. of workingday",ylab = "no. of users")

barplot(table(BikeData$holiday),main = "holiday",xlab = "no. of holiday",ylab = "no. of users")

###########################################################################################

###########################################################################################

#Bivariate analysis

#Draw scatter plot for all the numeric Variable and Count

# Feature Engineering

#Year

#Create the Year column from datetime column

BikeData$year <- substr(BikeData$datetime,1,4)

#convert that to factor

BikeData$year <- as.factor(BikeData$year)

#Check the unique data in BikeData$year

unique(BikeData$year)

#Month

#Create the Month column from datetime column

BikeData$month <- substr(BikeData$datetime,6,7)

#convert that to factor

BikeData$month <- as.integer(BikeData$month)

#Check the unique data in BikeData$Month

unique(BikeData$month)

#Date

#Create the days of week column from datetime column

date<-substr(BikeData$datetime,1,10)

?weekdays

days<-weekdays(as.Date(date))

BikeData$date<-days

#convert that to factor

BikeData$date <- as.factor(BikeData$date)

#Check the unique data in BikeData$day

unique(BikeData$date)

#Hour

BikeData$hour <- substr(BikeData$datetime,12,13)

#convert that to factor

BikeData$hour <- as.factor(BikeData$hour)

#Check the unique data in BikeData$Hour

unique(BikeData$hour)

#Bivariate analysis

#1. Trend by Weather using boxplot()

par(mfrow=c(1,1))

?boxplot

#count

boxplot(BikeData$count~BikeData$weather,main="Weather Trend",xlab="weather",ylab="No. of users")

#registered

boxplot(BikeData$registered~BikeData$year,xlab="year", ylab="registered users")

#. Trend by Humidity using boxplot()

boxplot(BikeData$count~BikeData$humidity, main= "Humidity Trend",xlab="Humidity",ylab="No. of Uses")

#. Check the Correlation for all the numerical variable - (registered,casual,count,temp,humidity,atemp,windspeed)

boxplot(BikeData$count~BikeData$year,xlab="year", ylab="Total users")

boxplot(BikeData$registered~BikeData$year,xlab="year", ylab="registered users")

# Demand with registered users increases over year

boxplot(BikeData$casual~BikeData$year,xlab="year", ylab="casual users")

#1.Trend by Hour

# Find total count of User Trend by Hour

boxplot(BikeData$count~BikeData$hour, xlab ="Hour", ylab = "Count of User")

#High User:7 to 9 and 17-1

#######################################################################

#Data Manipulation Exercise

# Load the dplyr package

library(dplyr)

# Call both head() and summary() on BikeData

summary(BikeData)

str(BikeData)

# tbl - tibble (Special type of dataframe)

BikeData<- tbl\_df(BikeData)

glimpse(BikeData)

# Glimpse at BikeData

######################################################################################

# Five verbs of dplyr - select, filter, arrange, mutate, summarize

# The dplyr package contains five key data manipulation functions, also called verbs:

# 1. select(), -> select specific column from a tbl,

# 2. filter(), -> filter specific rows which matches the logical condition

# 3. arrange(), -> that reorders the rows according to single or multiple variables,

# 4. mutate(), -> add columns from existing data,

# 5. summarise(), which reduces each group to a single row by calculating aggregate measures.

#######################################

#######################################

#select()

tbl <- select (BikeData,temp,atemp)

glimpse(tbl)

################################

#starts\_with("X"): every name that starts with "h",

select(BikeData,starts\_with("h"))

#ends\_with("X"): every name that ends with "day",

select(BikeData.ends\_with("h"))

#contains("X"): every name that contains "day",

#matches("X"): every name that matches "e", where "X" can be a regular expression,

#num\_range("x", 1:5): the variables named x01, x02, x03, x04 and x05,

#one\_of(x): every name that appears in x, which should be a character vector.

#mutate

#Data Manipulation Exercise

# Load the dplyr package

library(dplyr)

# Call both head() and summary() on BikeData

head(BikeData)

summary(BikeData)

str(BikeData)

# tbl - tibble (Special type of dataframe)

# Convert the hflights data.frame into a hflights tbl

BikeData <- tbl\_df(BikeData)

# Glimpse at BikeData

glimpse(BikeData)