import numpy as np

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

x=np.array([[1],[2],[3],[4],[5]])

y=np.array([2,3,4,3.5,5])

model=LinearRegression()

model.fit(x,y)

print("Slope : ", model.coef\_[0])

print("Intercept : ", model.intercept\_)

y\_pred=model.predict(x)

print("Predictions")

for i in range(len(x)):

  print(f"X: {x[i][0]}, Y predicted: {y\_pred[i]}")

plt.scatter(x,y, label="original data points")

plt.plot(x,y\_pred,"r-", label="Regression line")

plt.xlabel("X axis")

plt.ylabel("y axis")

plt.legend()

plt.show()

Exp1 – Data Exploration using R

install.packages("LearnBayes")

library(LearnBayes)

data(mtcars)

print(mtcars[1:10,])

table(mtcars$mpg)

table(mtcars$cyl)

table(mtcars$gear)

barplot(table(mtcars$gear),xlab="gear",ylab="count")

sub\_disp\_mpg=mtcars$disp-mtcars$mpg

summary(sub\_disp\_mpg)

hist(sub\_disp\_mpg, main="")

barplot(table(mtcars$drat))

boxplot(mtcars$disp~mtcars$mpg)

Exp2 – Normal Population

d <- list(int.lo=c(-Inf, seq(66, 74, by=2)),

          int.hi=c(seq(66, 74, by=2), Inf),

          f=c(14, 30, 49, 70, 33, 15))

y <- c(rep(65,14), rep(67,30), rep(69,49),

       rep(71,70), rep(73,33), rep(75,15))

mean(y)

log(sd(y))

start <-c(70,1)

fit <-laplace(groupeddatapost, start, d)

fit

modal.sds <- sqrt(diag(fit$var))

proposal <- list(var=fit$var, scale=2)

fit2 <- rwmetrop(groupeddatapost,

                 proposal,

                 start,

                 10000, d)

fit2$accept

modal.sds <- sqrt(diag(fit$var))

proposal <- list(var=fit$var, scale=2)

fit2 <- rwmetrop(groupeddatapost,

                 proposal,

                 start,

                 10000, d)

fit2$accept

post.means <- apply(fit2$par, 2, mean)

post.sds <- apply(fit2$par, 2, sd)

cbind(c(fit$mode), modal.sds)

cbind(post.means, post.sds)

mycontour(groupeddatapost,

          c(69, 71, .6, 1.3), d,

          xlab="mu",ylab="log sigma")

points(fit2$par[5001:10000, 1],

       fit2$par[5001:10000, 2])

Exp3 – Circle area

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

radius=1

N=100000

x=np.random.uniform(low=-radius, high=radius, size=N)

y=np.random.uniform(low=-radius, high=radius, size=N)

R=np.sqrt(x\*\*2+y\*\*2)

box\_area=(2\*radius)\*\*2

is\_point\_inside=R<radius

N\_inside=np.sum(is\_point\_inside)

circle\_area=N\_inside\*box\_area/N

plt.scatter(x,y,s=5.0, cmap=plt.cm.Paired, edgecolors='none', c=is\_point\_inside)

plt.axis('equal')

print("The area of circle is ", circle\_area)

Exp4 – Priors

library(LearnBayes)

set.seed(123)

n <- 100

x <- rnorm(n, mean = 5, sd = 2)

y <- 2 \* x + rnorm(n, mean = 0, sd = 1)

fit <- lm(y ~ x)

newdata <- data.frame(x = seq(min(x), max(x), length.out = 100))

discrete\_prior <- function(x) {

  return(dnorm(x, mean = 5, sd = 2))

}

beta\_prior <- function(x) {

  return(dbeta(x, shape1 = 2, shape2 = 2))

}

hist\_prior <- function(x) {

  hist\_x <- hist(x, plot = FALSE)

  return(hist\_x$density)

}

pred\_discrete <- predict(fit, newdata = newdata, priorfun = discrete\_prior)

pred\_beta <- predict(fit, newdata = newdata, priorfun = beta\_prior)

pred\_hist <- predict(fit, newdata = newdata, priorfun = hist\_prior)

par(mfrow = c(3, 1))

plot(x, y, main = "Data and True Regression Line", col = "blue", pch = 16)

abline(coef(fit), col = "red", lwd = 2)

plot(newdata$x, pred\_discrete, type = "l", col = "blue", lwd = 2,

     main = "Discrete Prior Prediction")

plot(newdata$x, pred\_beta, type = "l", col = "blue", lwd = 2,

     main = "Beta Prior Prediction")

plot(newdata$x, pred\_hist, type = "l", col = "blue", lwd = 2,

     main = "Histogram Prior Prediction")

par(mfrow = c(1, 1))

# \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# Exp4-Priors-According to lab

#Using discrete prior

library('LearnBayes')

p <- seq(0.05, 0.95, by = 0.1)

prior <- c(1, 5.2, 8, 7.2, 4.6, 2.1, 0.7, 0.1, 0, 0)

prior <- prior / sum(prior)

plot(p, prior, type = "h", ylab="Prior Probability")

#The posterior for p:

data <- c(11, 16)

post <- pdisc(p, prior, data)

round(cbind(p, prior, post),2)

library(lattice)

PRIOR <- data.frame("prior", p, prior)

POST <- data.frame("posterior", p, post)

names(PRIOR) <- c("Type", "P", "Probability")

names(POST) <- c("Type","P","Probability")

data <- rbind(PRIOR, POST)

xyplot(Probability ~ P | Type, data=data,

       layout=c(1,2), type="h", lwd=3, col="black")

#Using a Beta Prior

quantile2 <- list(p=.9, x=.5)

quantile1 <- list(p=.5, x=.3)

(ab <- beta.select(quantile1,quantile2))

#Bayesian triplot:

a <- ab[1]

b <- ab[2]

s <- 11

f <- 16

curve(dbeta(x, a + s, b + f), from=0, to=1, xlab="p", ylab="Density", lty=1, lwd=4)

curve(dbeta(x, s + 1, f + 1), add=TRUE, lty=2, lwd=4)

curve(dbeta(x, a, b), add=TRUE, lty=3, lwd=4)

legend(.7, 4, c("Prior", "Likelihood", "Posterior"), lty=c(3, 2, 1), lwd=c(3, 3, 3))

#Posterior summaries:

1 - pbeta(0.5, a + s, b + f)

qbeta(c(0.05, 0.95), a + s, b + f)

#Simulating from posterior:

ps <- rbeta(1000, a + s, b + f)

hist(ps, xlab="p")

sum(ps >= 0.5) / 1000

quantile(ps, c(0.05, 0.95))

#Using a Histogram Prior

midpt <- seq(0.05, 0.95, by = 0.1)

prior <- c(1, 5.2, 8, 7.2, 4.6, 2.1, 0.7,

           0.1, 0, 0)

prior <- prior / sum(prior)

curve(histprior(x, midpt, prior), from=0, to=1, ylab="Prior density", ylim=c(0, .3))

curve(histprior(x,midpt,prior) \* dbeta(x, s + 1, f + 1), from=0, to=1, ylab="Posterior density")

p <- seq(0, 1, length=500)

post <- histprior(p, midpt, prior) \* dbeta(p, s + 1, f + 1)

post <- post / sum(post)

ps <- sample(p, replace = TRUE, prob = post)

hist(ps, xlab="p", main="")

#Prediction

#Want to predict the number of heavy sleepers in a future sample of 20.

#Discrete prior approach:

p <- seq(0.05, 0.95, by=.1)

prior <- c(1, 5.2, 8, 7.2, 4.6, 2.1, 0.7, 0.1, 0, 0)

prior <- prior / sum(prior)

m <- 20

ys <- 0:20

pred <- pdiscp(p, prior, m, ys)

cbind(0:20, pred)

#Continuous prior approach:

ab <- c(3.26, 7.19)

m <- 20

ys <- 0:20

pred <- pbetap(ab, m, ys)

#Simulating predictive distribution:

p <- rbeta(1000, 3.26, 7.19)

y <- rbinom(1000, 20, p)

table(y)

freq <- table(y)

ys <- as.integer(names(freq))

predprob <- freq / sum(freq)

plot(ys, predprob, type="h", xlab="y", ylab="Predictive Probability")

dist <- cbind(ys, predprob)

#Construction of a prediction interval:

covprob <- .9

discint(dist, covprob)