Project 1

Will Graham; Richelle Lee; Robin Lin

2022-10-06

Ctrl + Shift + H: Set Working Directory

a <- scan("pg10.txt",what="character",skip=104) ## skip contents  
n <- length(a)  
a <- a[-((n-2886):n)] ## strip license  
a <- a[-grep("[0123456789]:[0123456789]",a)] ## strip out verse numbers  
  
# a <- a[1:100] # test

A Function Splitting Punctuation Marks

indexes\_split\_punct <- function(string){  
 ## Separating words with punctuation marks  
 indexes <- grep("[',','.','?','!',':',';']",a) ## obtain the indexes of words containing non-letter characters   
 return(indexes)  
}  
  
get\_punct <- function(r){  
 return(substr(r, nchar(r), nchar(r)))  
}  
  
split\_punct <- function(b, test){  
 punct\_indexes <- indexes\_split\_punct(b)  
 punct <- lapply(b[punct\_indexes] , FUN = get\_punct)  
 new\_b <- rep(0, (length(punct\_indexes) + length(b)))  
 final\_punct\_indexes <- punct\_indexes + 1:length(punct\_indexes)  
 new\_b[final\_punct\_indexes] <- punct  
 no\_punct\_list <- gsub('[[:punct:] ]+','',b)  
 if (test == TRUE){  
 b <- b[1:100]  
 new\_b <- new\_b[1:115]  
 }  
 new\_b[-final\_punct\_indexes] <- no\_punct\_list  
 return(new\_b)  
}

#Lowering capitals  
new\_b <- split\_punct(a, FALSE)  
lower\_new\_b <- tolower(new\_b)  
unique\_words <- unique(lower\_new\_b)  
index\_match <- match(lower\_new\_b, unique\_words)  
tabulate\_index\_match <- tabulate(index\_match)  
sort\_results <- sort(tabulate\_index\_match, index.return = TRUE, decreasing = TRUE)  
top\_500 <- sort\_results$ix[1 : 500]  
result\_list <- unique\_words[top\_500]

common\_word\_match <- match(lower\_new\_b, result\_list)  
first\_column <- common\_word\_match[1 : (length(common\_word\_match) - 2)]  
second\_column <- first\_column[2 : (length(common\_word\_match) - 1)]  
third\_column <- first\_column[3 : length(common\_word\_match)]  
T\_array <- data.frame(first\_column, second\_column, third\_column)  
  
T\_array <- T\_array[is.na(rowSums(T\_array)) == FALSE,]  
  
  
Index\_to\_common\_words <- function(x){  
 return(result\_list[x])  
}  
matrix\_x <- cbind(c(lapply(T\_array$first\_column, FUN = Index\_to\_common\_words)), c(lapply(T\_array$second\_column, FUN = Index\_to\_common\_words)), c(lapply(T\_array$third\_column, FUN = Index\_to\_common\_words)))  
  
Inital\_W\_map <- unique(T\_array$first\_column)  
Intial\_word\_probability <- tabulate(match(T\_array$first\_column, unique(T\_array$first\_column)))  
Initail\_prob <- Intial\_word\_probability/sum(Intial\_word\_probability)  
  
second\_W\_map <- unique(T\_array[1:2])  
second\_W\_map\_conc <- paste(second\_W\_map[[1]], second\_W\_map[[2]])  
T\_array\_conc <- paste(T\_array[[1]], T\_array[[2]])  
df2 <- data.frame(c(second\_W\_map\_conc), rep(0, length(second\_W\_map\_conc[[1]])))  
colnames(df2) <- c("Concat\_w1\_w2", "Freq")  
  
match\_indexes <- match(T\_array\_conc, second\_W\_map\_conc)  
tabulate\_match\_index <- tabulate(match\_indexes)  
df2['Freq'] <- tabulate\_match\_index  
  
third\_W\_map <- unique(T\_array[1:3])  
third\_W\_map\_conc <- paste(third\_W\_map[[1]], third\_W\_map[[2]], third\_W\_map[[3]])  
T\_array\_conc2 <- paste(T\_array[[1]], T\_array[[2]], T\_array[[3]])  
df3 <- data.frame(c(third\_W\_map\_conc), rep(0, length(third\_W\_map\_conc[[1]])))  
colnames(df3) <- c("Concat\_w1\_w2\_w3", "Freq")  
  
match\_indexes2 <- match(T\_array\_conc2, third\_W\_map\_conc)  
tabulate\_match\_index2 <- tabulate(match\_indexes2)  
df3['Freq'] <- tabulate\_match\_index2

PI <- grep("[',','.','?','!',':',';']", result\_list)  
  
word\_1 <- function(){  
 while (TRUE){  
 ret <- sample(Inital\_W\_map, 1, prob = Initail\_prob)  
 if (!(ret %in% PI)) {  
 return(as.numeric(ret))  
 }  
 }  
}  
  
word\_2 <- function(w1){  
 new\_ops <- second\_W\_map[second\_W\_map[1] == w1,]  
 freq <- c(1:length(new\_ops[[1]]))  
 for (i in c(1:length(new\_ops[[1]]))){  
 freq[i] <- subset(df2, df2$Concat\_w1\_w2 == paste(new\_ops[i,1], new\_ops[i,2]), 2)[[1]]  
 }  
 sample\_1 <- as.numeric(sample(as.numeric(as.character(as.factor(new\_ops[,2]))), 1, prob = freq))  
 return(sample\_1)  
}  
  
word\_3 <- function(w1, w2){  
 new\_ops <- third\_W\_map[third\_W\_map[1] == w1 & third\_W\_map[2] == w2,]  
 freq <- c(1:length(new\_ops[[1]]))  
 for (i in c(1:length(new\_ops[[1]]))){  
 freq[i] <- subset(df3, df3$Concat\_w1\_w2\_w3 == paste(new\_ops[i,1], new\_ops[i,2], new\_ops[i,3]),2)[[1]]  
 }  
 sample\_1 <- as.numeric(sample(as.numeric(as.character(as.factor(new\_ops[,3]))), 1, prob = freq))  
 return(sample\_1)  
}

words\_generator\_50 <- function(){  
 sentence <- as.numeric(rep (0,50))  
 w1 <- word\_1()  
 sentence[1] <- w1  
 w2 <- word\_2(w1)  
 sentence[2] <- w2  
 for (i in c(3:length(sentence))){  
 if(paste(sentence[i - 2], sentence[i - 1]) %in% df2[1]){  
 sentence[i] <- word\_3(sentence[i - 2], sentence[i - 1])  
 } else if (sentence[i - 1] %in% second\_W\_map[[1]]){  
 sentence[i] <- word\_2(sentence[i - 1])  
 } else {  
 sentence[i] <- word\_1()  
 }  
 }  
  
 sentence2 <- c(1:50)  
 for (i in c(1:length(sentence2))){  
 sentence2[i] <- result\_list[as.numeric(sentence[i])]  
 }  
 return(sentence2)  
}

test <- words\_generator\_50()  
cat(test)

## he slew all israel , saying , and he went in jerusalem dwelt there fell upon princes of the holy unto them many , and bread , whom if his servant , serve , shall be thou keep not there be broken : i will go to eat bread shall