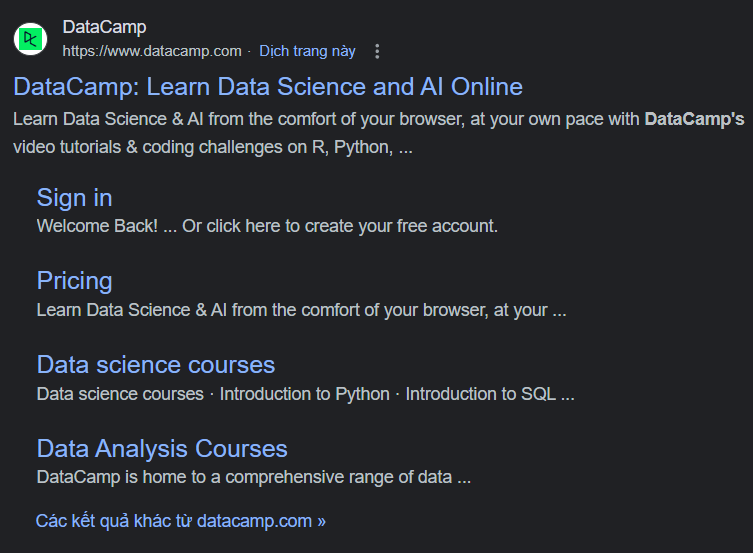
**Lab 11: TF-IDF**

**Họ và Tên**: Trần Nguyễn Gia Long

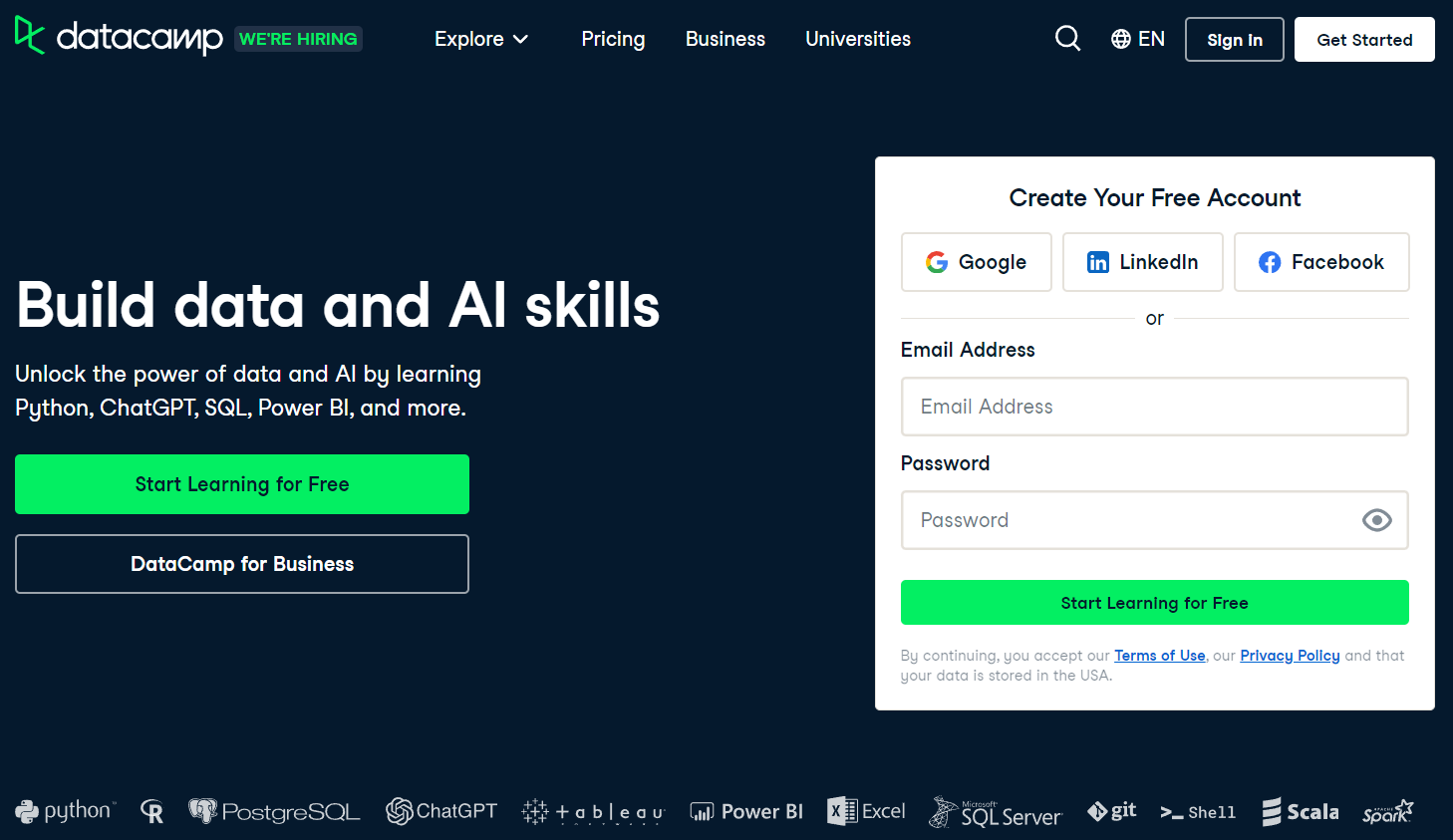
**MSSV**: 1050080059

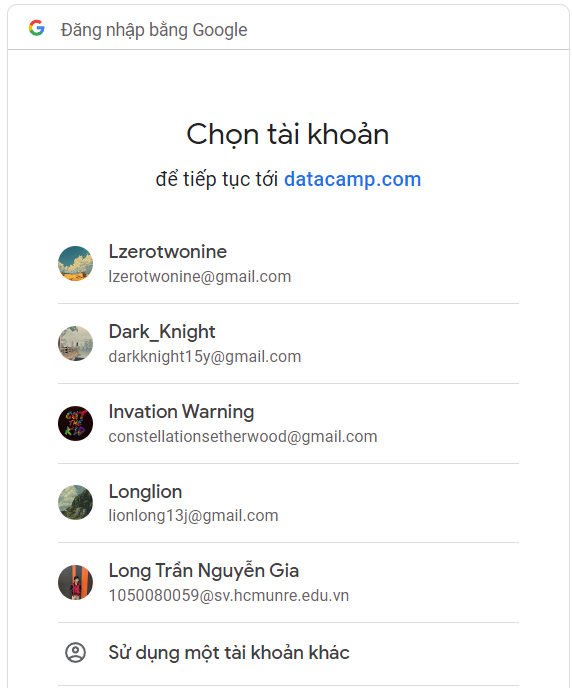
# 1. Cài đặt TF-IDF

- Sử dụng trang DataCamp để chạy TF-IDF.R.

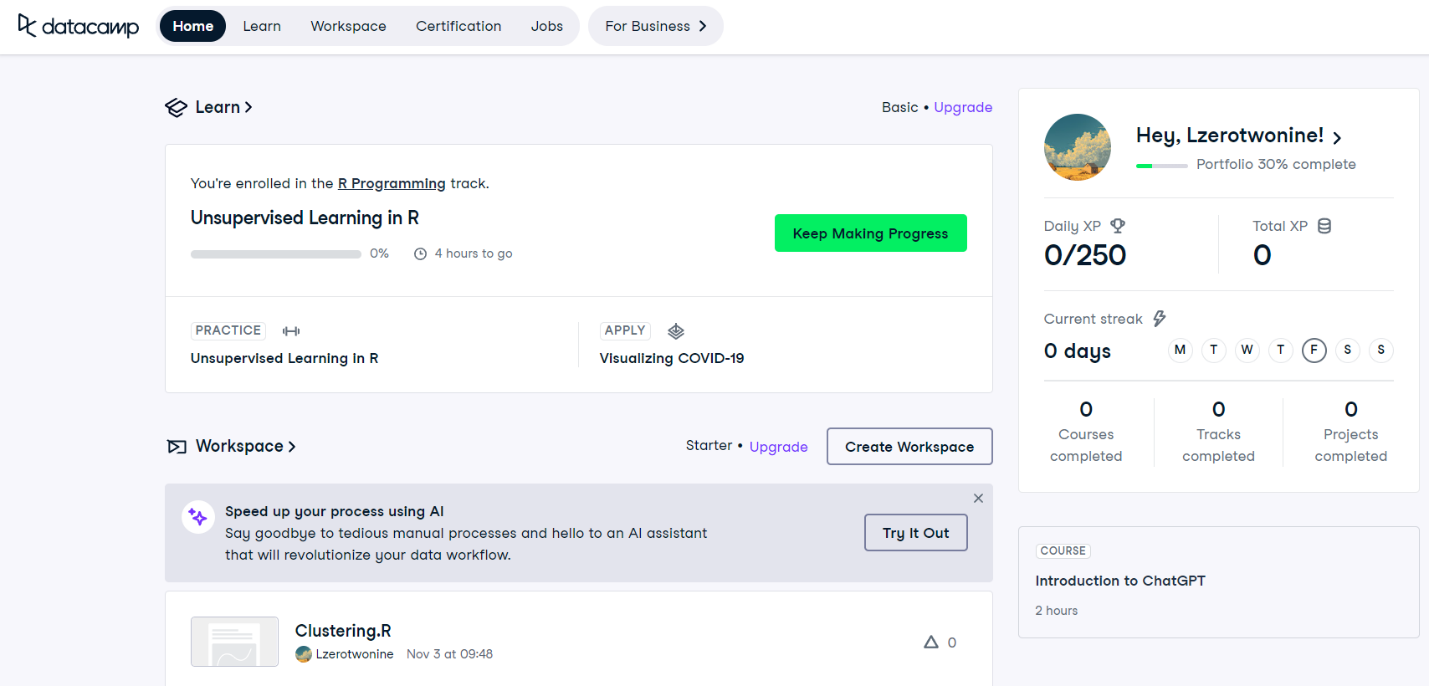


- Nhấp vào trang, tiến hành việc đăng nhập.

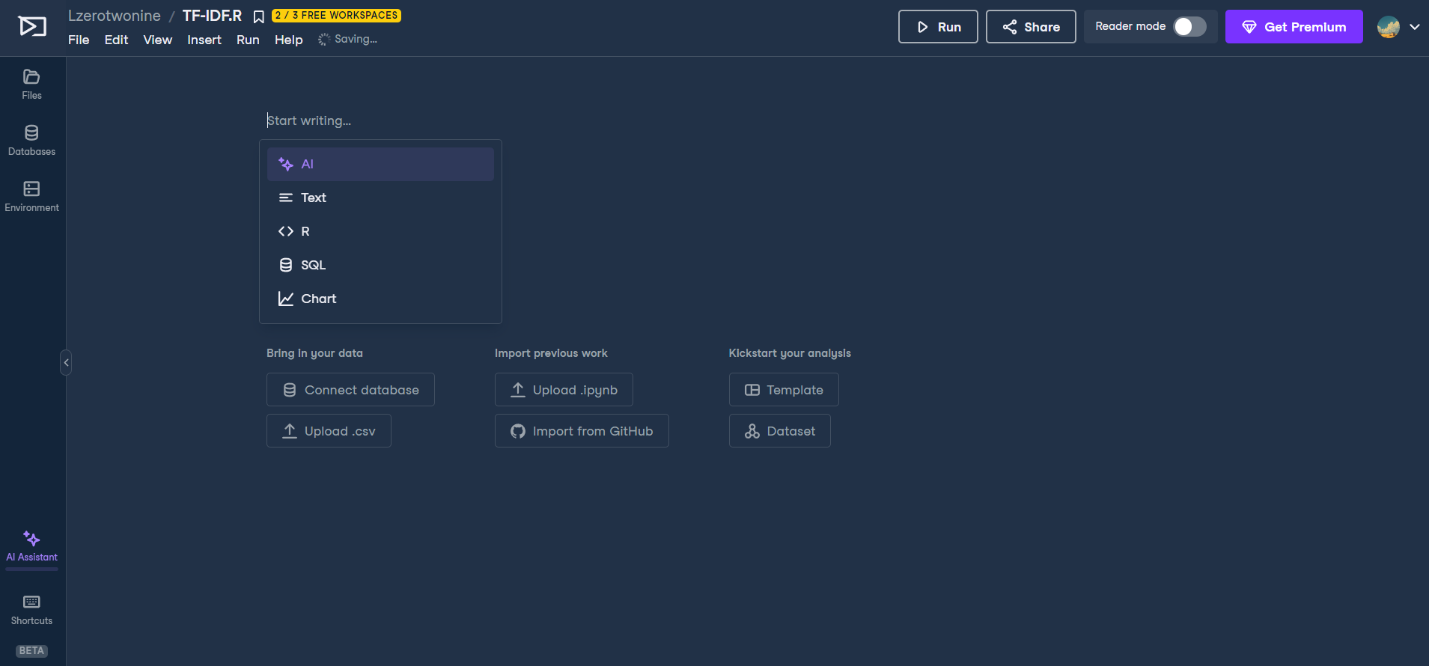




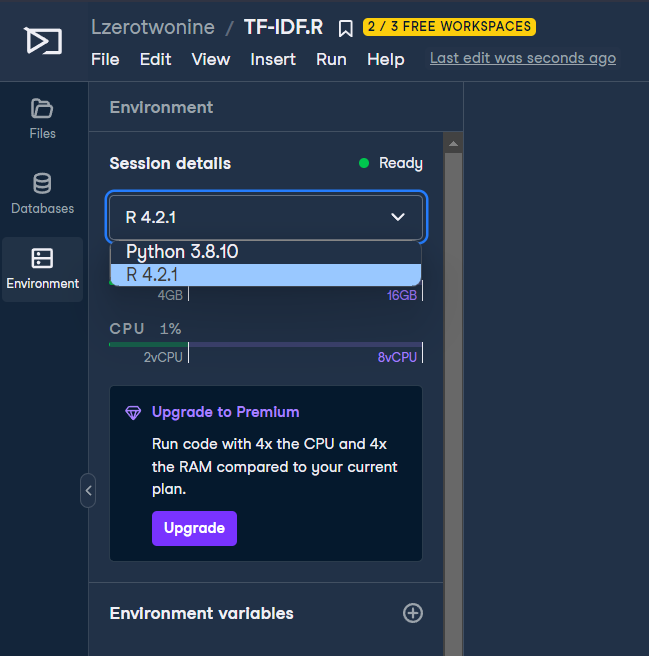
- Ở đây em chọn đăng nhập bằng tài khoản Google để vào nhanh chóng.



- Sau khi tạo tài khoản, ta bắt đầu tạo một workspace để dán và chạy code.



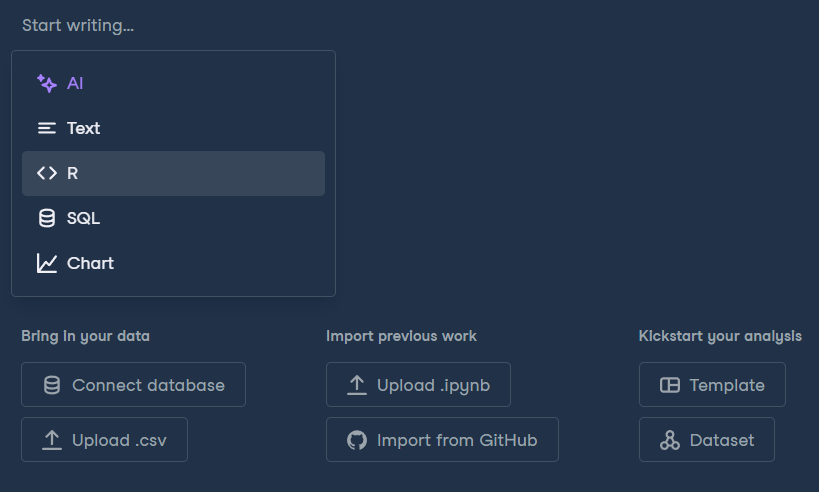
- Đây là giao diện của workspace.

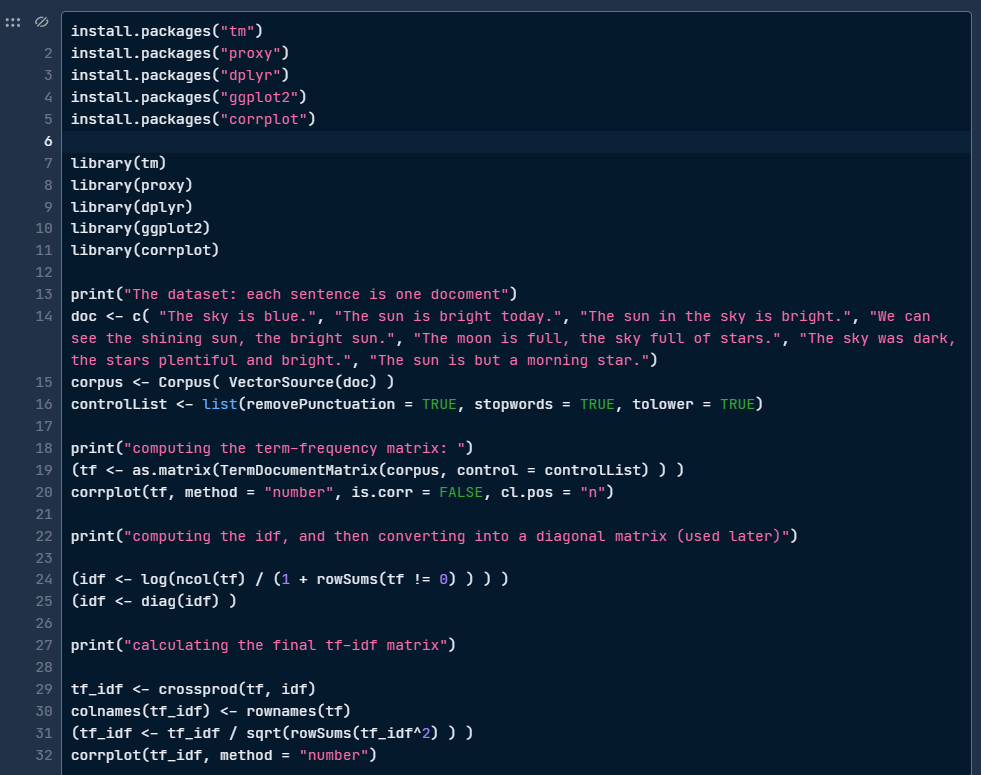


- Thay đổi ngôn ngữ lập trình thành R.

# 2. Chạy TF-IDF.

- Chọn viết code bằng R, sau đó dán đoạn code từ Lab 11 vào.





install.packages("tm")

install.packages("proxy")

install.packages("dplyr")

install.packages("ggplot2")

install.packages("corrplot")

library(tm)

library(proxy)

library(dplyr)

library(ggplot2)

library(corrplot)

* **tm**: Thư viện này cung cấp một công cụ khám phá trực quan trên ma trận tương quan, hỗ trợ sắp xếp biến tự động để giúp phát hiện các mẫu ẩn giữa các biến.
* **proxy**: Thư viện này cung cấp một khung mở rộng cho việc tính toán hiệu quả tự động và chéo-tính gần.
* **dplyr**: Đây là một ngữ pháp để thao tác dữ liệu, cung cấp một tập hợp nhất quán các động từ giúp ta giải quyết các thách thức thao tác dữ liệu phổ biến.
* **ggplot2**: ggplot2 là một hệ thống tạo đồ họa khai báo, dựa trên Ngữ pháp đồ họa. Khi ta cung cấp dữ liệu, cho ggplot2 biết cách ánh xạ các biến theo tính thẩm mỹ, sử dụng những nguyên tắc đồ họa nào và sẽ xử lý các chi tiết.
* **corrplot**: Thư viện này cung cấp một công cụ khám phá trực quan trên ma trận tương quan, hỗ trợ sắp xếp biến tự động để giúp phát hiện các mẫu ẩn giữa các biến.

print("The dataset: each sentence is one docoment")

doc <- c( "The sky is blue.", "The sun is bright today.", "The sun in the sky is bright.", "We can see the shining sun, the bright sun.", "The moon is full, the sky full of stars.", "The sky was dark, the stars plentiful and bright.", "The sun is but a morning star.")

corpus <- Corpus( VectorSource(doc) )

controlList <- list(removePunctuation = TRUE, stopwords = TRUE, tolower =TRUE)

* **Corpus(VectorSource(doc)):** Đầu tiên, đoạn mã tạo một “corpus” từ vector ‘doc’. Trong khai thác văn bản, một “corpus” là một tập hợp các văn bản. Trong trường hợp này, mỗi câu trong vector ‘doc’ được coi là một văn bản riêng biệt¹.
* **removePunctuation = TRUE:** Tiếp theo, đoạn mã xác định một danh sách các thao tác để thực hiện trên corpus. Tham số ‘removePunctuation = TRUE’ chỉ định rằng tất cả các dấu chấm câu nên được loại bỏ từ các văn bản.
* **stopwords = TRUE:** Tham số ‘stopwords = TRUE’ chỉ định rằng tất cả các “stop words” nên được loại bỏ từ các văn bản. "Stop words" là những từ phổ biến như “the”, “is”, “in”, v.v., thường được loại bỏ trong quá trình xử lý ngôn ngữ tự nhiên vì chúng ít có ý nghĩa phân loại⁹.
* **tolower =TRUE:** Tham số ‘tolower = TRUE’ chỉ định rằng tất cả các từ trong các văn bản nên được chuyển thành chữ thường. Điều này giúp đảm bảo rằng cùng một từ không được coi là hai từ riêng biệt chỉ vì chúng có chữ hoa.

print("computing the term-frequency matrix: ")

(tf <- as.matrix(TermDocumentMatrix(corpus, control = controlList) ) )

corrplot(tf, method = "number", is.corr = FALSE, cl.pos = "n")

print("computing the idf, and then converting into a diagonal matrix (used later)")

* **as.matrix(TermDocumentMatrix(corpus, control = controlList)):** Tạo một ma trận tần suất thuật ngữ (term-frequency matrix) từ corpus. Trong khai thác văn bản, một ma trận tần suất thuật ngữ là một ma trận mà mỗi hàng tương ứng với một thuật ngữ (hoặc từ) và mỗi cột tương ứng với một văn bản. Giá trị tại hàng i và cột j của ma trận là số lần thuật ngữ i xuất hiện trong văn bản j.
* **corrplot(tf, method = "number", is.corr = FALSE, cl.pos = "n")**: Tiếp theo, đoạn mã vẽ một biểu đồ để trực quan hóa ma trận tần suất thuật ngữ. Tham số ‘method = "number"’ chỉ định các giá trị trong ma trận nên được hiển thị như là các số, ‘is.corr = FALSE’ chỉ định ma trận không phải là một ma trận tương quan, ‘cl.pos = "n"’ chỉ định không nên vẽ một chú giải màu.

(idf <- log(ncol(tf) / (1 + rowSums(tf != 0) ) ) )

(idf <- diag(idf) )

* **log(ncol(tf) / (1 + rowSums(tf != 0) ) ):** Dòng này tính toán giá trị idf cho mỗi thuật ngữ. Giá trị idf của một thuật ngữ là một giá trị số được tính toán dựa trên số lượng văn bản mà thuật ngữ đó xuất hiện.
* **diag(idf):** Tiếp theo, đoạn mã chuyển đổi vector idf thành một ma trận đường chéo. Một ma trận đường chéo là một ma trận mà tất cả các phần tử ngoại trừ các phần tử trên đường chéo chính đều bằng 0. Các giá trị trên đường chéo chính của ma trận là các giá trị idf.

Hai dòng code này đang chuẩn bị để tính toán giá trị tf-idf cho mỗi thuật ngữ trong mỗi văn bản. Giá trị tf-idf của một thuật ngữ trong một văn bản là kết quả của tần suất thuật ngữ (tf) và tần suất tài liệu nghịch đảo (idf) của thuật ngữ đó.

print("calculating the final tf-idf matrix")

tf\_idf <- crossprod(tf, idf)

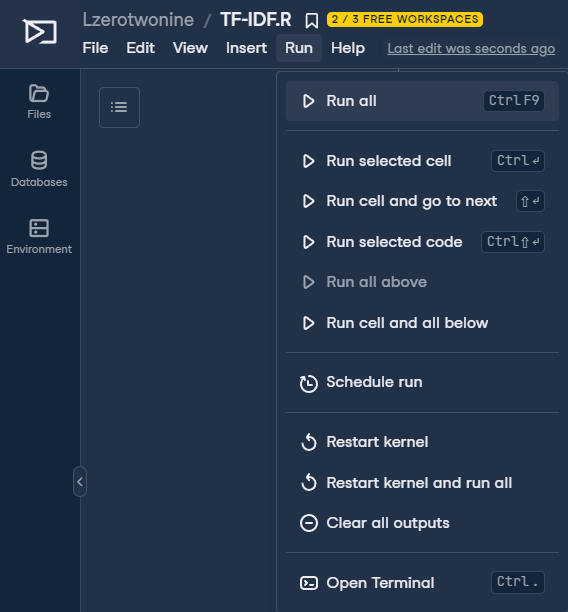
colnames(tf\_idf) <- rownames(tf)

(tf\_idf <- tf\_idf / sqrt(rowSums(tf\_idf^2) ) )

corrplot(tf\_idf, method = "number")

* **crossprod(tf, idf):** Đầu tiên, đoạn mã tính toán tích chéo giữa ma trận tần suất thuật ngữ ‘tf’ và ma trận đường chéo ‘idf’. Kết quả là một ma trận mới, trong đó mỗi phần tử là tích của tần suất thuật ngữ và tần suất tài liệu nghịch đảo (tf-idf) của thuật ngữ tương ứng.
* **colnames(tf\_idf) <- rownames(tf):** Tiếp theo, đoạn mã đặt tên cột của ma trận ‘tf\_idf’ bằng tên hàng của ma trận ‘tf’. Điều này đảm bảo rằng mỗi cột trong ma trận ‘tf\_idf’ tương ứng với một thuật ngữ.
* **tf\_idf <- tf\_idf / sqrt(rowSums(tf\_idf^2)):** Sau đó, đoạn mã chuẩn hóa ma trận ‘tf\_idf’ bằng cách chia mỗi phần tử trong ma trận cho căn bậc hai của tổng bình phương các phần tử trong cùng một hàng. Điều này giúp chuẩn hóa giá trị tf-idf, làm cho chúng không phụ thuộc vào độ dài của vector tf-idf.
* **corrplot(tf\_idf, method = "number"):** Cuối cùng, đoạn mã vẽ một biểu đồ để trực quan hóa ma trận ‘tf\_idf’. Tham số ‘method = "number"’ chỉ định rằng các giá trị trong ma trận nên được hiển thị như là các số.

- Sau đó là bấm chạy



# 3. Kết quả

