R Notebook

Code

This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

Hide

```
# 1. Installation et chargement des packages
packages <- c("dplyr", "ggplot2", "readr")
installed <- rownames(installed.packages())
for (p in packages) {
  if (!(p %in% installed)) install.packages(p)
  library(p, character.only = TRUE)
}
```

```
G3;
Attachement du package : ‘dplyr’

gG3;Les objets suivants sont masqués depuis ‘package:stats’:

    filter, lag

gG3;Les objets suivants sont masqués depuis ‘package:base’:

    intersect, setdiff, setequal, union

g
```

Hide

```
# 2. Importation des fichiers de scores SIFT et FoldX
sift <- read_tsv(
  "https://raw.githubusercontent.com/HackBio-Internship/public_datasets/main/R/datasets/sift.
tsv",
  col_names = c("Protein", "Amino_Acid", "sift_Score"),
  skip = 1
)
```

```
`curl` package not installed, falling back to using `url()`
Rows: 155467 Columns: 3
── Column specification ────────────────────────────────────────────────────────
────
Delimiter: "\t"
chr (2): Protein, Amino_Acid
dbl (1): sift_Score

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Hide

```
foldx <- read_tsv(
  "https://raw.githubusercontent.com/HackBio-Internship/public_datasets/main/R/datasets/foldX.tsv",
  col_names = c("Protein", "Amino_Acid", "foldX_Score"),
  skip = 1
)
```

```
`curl` package not installed, falling back to using `url()`
Rows: 76575 Columns: 3
── Column specification ────────────────────────────────────────────────────────
Delimiter: "\t"
chr (2): Protein, Amino_Acid
dbl (1): foldX_Score

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Hide

```
print(foldx)
```

Protein <chr>	**Amino_Acid** <chr>	**foldX_Score** <dbl>
A5A607	E63D	1.842160
A5A607	K60Q	0.517100
A5A607	Y56C	0.728700
A5A607	Y56S	0.842120
P00350	A117S	-0.098540
P00350	A117T	0.923520
P00350	A175P	-1.836980
P00350	A175T	0.128240
P00350	A200S	0.722800
P00350	A216S	0.615700

1-10 of 76,575 rows Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
print(sift)
```

Protein <chr>	**Amino_Acid** <chr>	**sift_Score** <dbl>
A5A607	K60Q	0.1977985366
A5A607	Y56C	0.1749700816

Protein <chr>	Amino_Acid <chr>	sift_Score <dbl>
A5A607	Y56S	0.4155813244
A5A630	A16V	0.5167381034
A5A630	A47T	0.3872713385
A5A630	A64E	0.3094996492
A5A630	A64T	0.4117617609
A5A630	A64V	0.5101598129
A5A630	A89V	0.9196705090
A5A630	F22V	0.5440349580

1-10 of 155,467 rows Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
# 3. Création de la clé unique specific_Protein_aa pour chaque Dataset
sift <- sift %>% mutate(specific_Protein_aa = paste(Protein, Amino_Acid, sep = "_"))
foldx <- foldx %>% mutate(specific_Protein_aa = paste(Protein, Amino_Acid, sep = "_"))

# 4. Fusionner les deux datasets sur la clé unique
merged_df <- inner_join(sift, foldx, by = "specific_Protein_aa")
head(merged_df)
```

Protein.x <chr>	Amino_Acid.x <chr>	sift_Score <dbl>	specific_Protein_aa <chr>	Protein.y <chr>	Amino_Acid.y <chr>	foldX
A5A607	K60Q	0.1977985	A5A607_K60Q	A5A607	K60Q	
A5A607	Y56C	0.1749701	A5A607_Y56C	A5A607	Y56C	
A5A607	Y56S	0.4155813	A5A607_Y56S	A5A607	Y56S	
P00350	A117S	0.4015193	P00350_A117S	P00350	A117S	
P00350	A117T	0.2827685	P00350_A117T	P00350	A117T	
P00350	A175P	1.0000000	P00350_A175P	P00350	A175P	

6 rows

Hide

```
print(merged_df)
```

Protein.x <chr>	Amino_Acid.x <chr>	sift_Score <dbl>	specific_Protein_aa <chr>	Protein.y <chr>	Amino_Acid.y <chr>	fo
A5A607	K60Q	0.1977985366	A5A607_K60Q	A5A607	K60Q	
A5A607	Y56C	0.1749700816	A5A607_Y56C	A5A607	Y56C	
A5A607	Y56S	0.4155813244	A5A607_Y56S	A5A607	Y56S	

Protein.x <chr>	Amino_Acid.x <chr>	sift_Score <dbl>	specific_Protein_aa <chr>	Protein.y <chr>	Amino_Acid.y <chr>	fo
P00350	A117S	0.4015193272	P00350_A117S	P00350	A117S	
P00350	A117T	0.2827685257	P00350_A117T	P00350	A117T	
P00350	A175P	1.0000000000	P00350_A175P	P00350	A175P	
P00350	A175T	0.3271497925	P00350_A175T	P00350	A175T	
P00350	A200S	0.3311445160	P00350_A200S	P00350	A200S	
P00350	A216S	0.3589971518	P00350_A216S	P00350	A216S	
P00350	A216T	0.0215808724	P00350_A216T	P00350	A216T	

Hide

```
# 5.Sélection des mutations délètères selon les critères fonctionnels et structuraux
deleterious_mutations <- merged_df %>%
  filter(sift_Score < 0.05 & foldX_Score > 2)

# 6. Extraction des acides aminés initiaux et calcul des fréquences
top_impacts <- deleterious_mutations %>%
  mutate(original_aa = substr(Amino_Acid.x, 1, 1)) %>%
  group_by(original_aa) %>%
  summarise(n = n()) %>%
  arrange(desc(n))
# 7. affichage de l'acide aminé le plus fréquent
most_common_aa <- top_impacts$original_aa[1]
cat("'acide aminé avec le plus grand impact est :", most_common_aa, "\n\n")
```

```
'acide aminé avec le plus grand impact est : G
```

Hide

```
# 8. filtrage des acides aminés impliqués dans plus de 100 mutations délètéres
high_freq_aas <- top_impacts %>% filter(n > 100)
print(high_freq_aas)
```

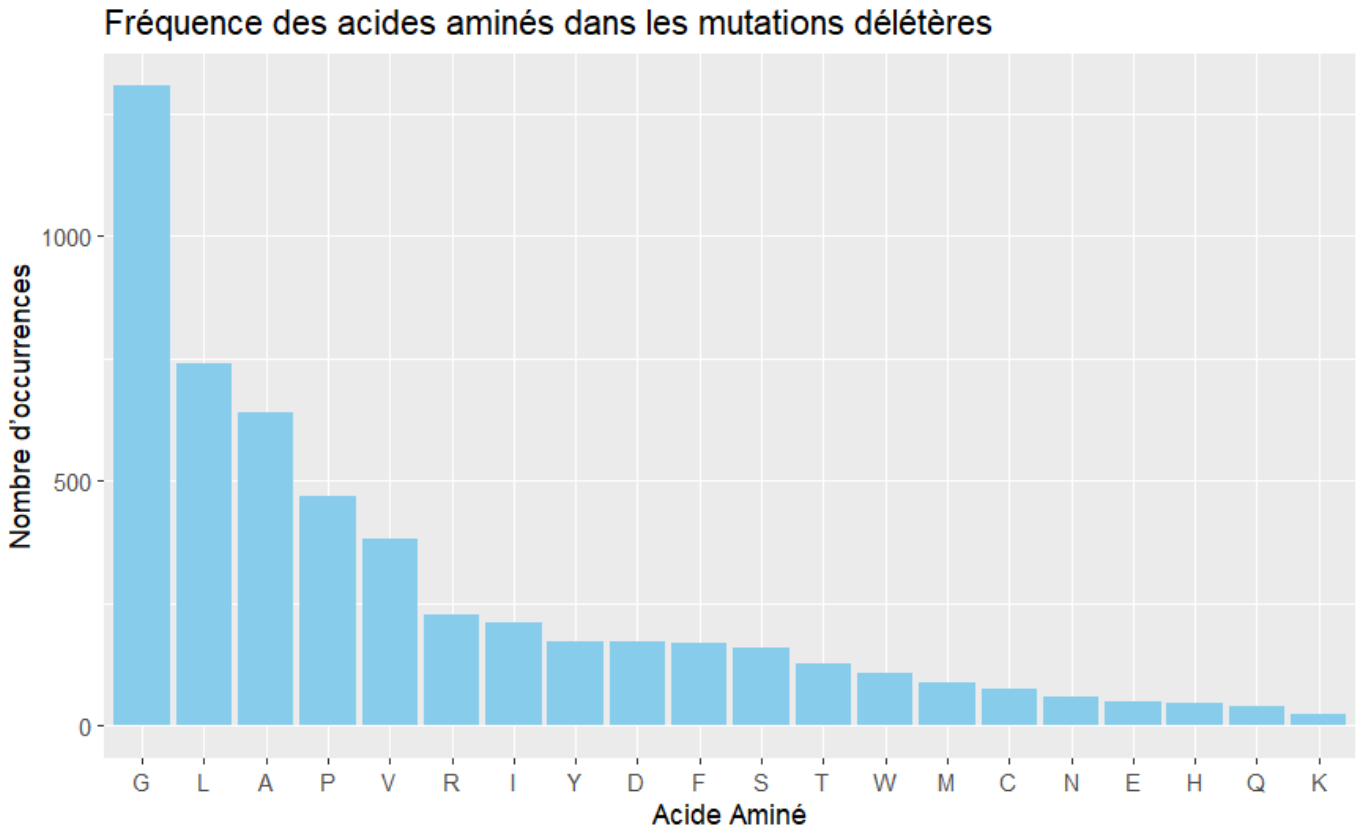
original_aa <chr>	n <int>
G	1307
L	739
A	640
P	470
V	380
R	227
I	212

original_aa <chr>	n <int>
Y	172
D	171
F	169

1-10 of 13 rows Previous **1** 2 Next

Hide

```
# 9. visualisation des fréquences en diagramme en barres
ggplot(top_impacts, aes(x = reorder(original_aa, -n), y = n)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(title = "Fréquence des acides aminés dans les mutations délétères",x = "Acide Aminé",
y = "Nombre d'occurrences")
```



Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.