## Software Tools (BINF*6210) Assignment #2

## Vlad Bularca

## October 25, 2024

## Introduction -------

# For Assignment #2 I will be building and testing a supervised Machine learning model that will make gene classifications based on sequence features. I will be analyzing NCBI GenBank sequence data for Orthomyxoviridae, a family of negative-sense RNA viruses, commonly known as influenza viruses. Within this family, are the commonly known Influenza type A and type B viruses (Bouvier & Palese, 2008). The classifier will categorize a DNA sequence as Polymerase Basic Protein 1 (PB1) or Polymerase Basic Protein 2 (PB2). I specifically chose 2 genes that are protein coding and closely related to see if the basic model we learned about can distinguish between the two. PB1 and PB2 are 2 of the proteins that make up the functional ribonucleoprotein (RNP) complex of influenza viruses (Hiromoto et al., 2000) The RNP is responsible for transcription and replication of viral RNA and is a heterotrimer polymerase complex made up of the PB2, PB1 and Polymerase Acidic (PA) proteins and is bound to viral RNA and the nucleoprotein (NP) (Hiromoto et al., 2000).

# The goal of this analysis is to see if different sequences can be separated into gene type based on nucleotide proportion. Additionally, I will create a second classifier that will consider two sequence features, gene type and species name. Since PB1 and PB2 are subunit proteins within the RNA polymerase complex, my hypothesis is that the genes may have similar properties and share high sequence similarity which may make it difficult to distinguish the two. Additionally, both proteins interact with the nucleoprotein, so it is expected that they have regions of similarity that translate to similar structural motifs (Gorman et al., 1990). The ability to categorize sequences into gene and species classes would be valuable when investigating DNA sequences isolated from influenza samples. This can also be used as a beginning step in more complex problems that investigate mutation and virulence.

```r
## Packages used -------

library(tidyverse)

library(viridis)

# + scale_color/fill_viridis_c/d()

theme_set(theme_light())

library(Biostrings)

library(rentrez)

library(seqinr)

library(dbplyr)

conflicted::conflicts_prefer(dplyr::filter())

conflicted::conflicts_prefer(dplyr::rename())

library(randomForest)

options(timeout = 300)


## Data Retrieval & Data Manipulation -------


# Retrieving 5000 PB1 sequences from NCBI GenBank for the Orthomyxoviridae family and
filtering for sequences between a length of 2200 and 2400 to remove any outliers. First, the
specific IDs were retrieved and then fetched using entrez functions. Data was retrieved on Oct
22, 2024.


# PB1_search <- entrez_search(db = "nuccore", term = "(Orthomyxoviridae[ORGN] AND
PB1[GENE] AND 2200:2400[SLEN]", retmax = 5000, use_history = TRUE)

# PB1_fetch <- entrez_fetch(db = "nuccore", web_history = PB1_search$web_history, rettype =
"fasta", retmax = 5000)

# Converting this output to a fasta file and then converting this to a data frame so I can further
process and visualize the data:
```

```
# write(PB1_fetch, "PB1_fetch.fasta", sep = "\n")

stringSet <- readDNAStringSet ("Assignment_2_Data/PB1_fetch.fasta")

dfPB1 <- data.frame(PB1_Title = names(stringSet), PB1_Sequence = paste(stringSet))

dfPB1$Species_Name <- word(dfPB1$PB1_Title, 2L, 3L)

View(dfPB1)

# There may be duplicate sequence copies in the data I pulled, so I am removing any records
that have the same sequence and title combination. This is done to ensure that I am using
distinct data points for my classification model.

dfPB1clean <- dfPB1[!duplicated(dfPB1[, c("PB1_Sequence", "PB1_Title")]), ]

# Adding a new column with the gene name, one with the sequence length and then
rearranging the columns:

dfPB1clean$Gene_Name <- "PB1"

dfPB1clean$Sequence_Length <- nchar(dfPB1clean$PB1_Sequence)

dfPB1clean <- dfPB1clean[, c("PB1_Title", "Gene_Name", "Species_Name", "PB1_Sequence",
"Sequence_Length")]

View(dfPB1clean)
```

```r
# Filtering out any missing sequences:

dfPB1clean <- dfPB1clean %>%

  filter(!is.na(PB1_Sequence))


# Retrieving 5000 PB2 sequences from NCBI GenBank for the Orthomyxoviridae family and filtering for sequences between a length of 2200 and 2400 to remove any outliers. First, the specific IDs were retrieved and then fetched using entrez functions. Data was retrieved on Oct 22, 2024.

# I will also do the same data manipulation/filtering steps from above for PB2:


# PB2_search <- entrez_search(db = "nuccore", term = "(Orthomyxoviridae[ORGN] AND PB2[GENE] AND 2200:2400[SLEN]", retmax = 5000, use_history = TRUE)


# PB2_fetch <- entrez_fetch(db = "nuccore", web_history = PB2_search$web_history, rettype = "fasta", retmax = 5000)


# write(PB2_fetch, "PB2_fetch.fasta", sep = "\n")


stringSet <- readDNAStringSet("Assignment_2_Data/PB2_fetch.fasta")


dfPB2 <- data.frame(PB2_Title = names(stringSet), PB2_Sequence = paste(stringSet))


dfPB2$Species_Name <- word(dfPB2$PB2_Title, 2L, 3L)


View(dfPB2)


dfPB2clean <- dfPB2[!duplicated(dfPB2[, c("PB2_Sequence", "PB2_Title")]), ]
```

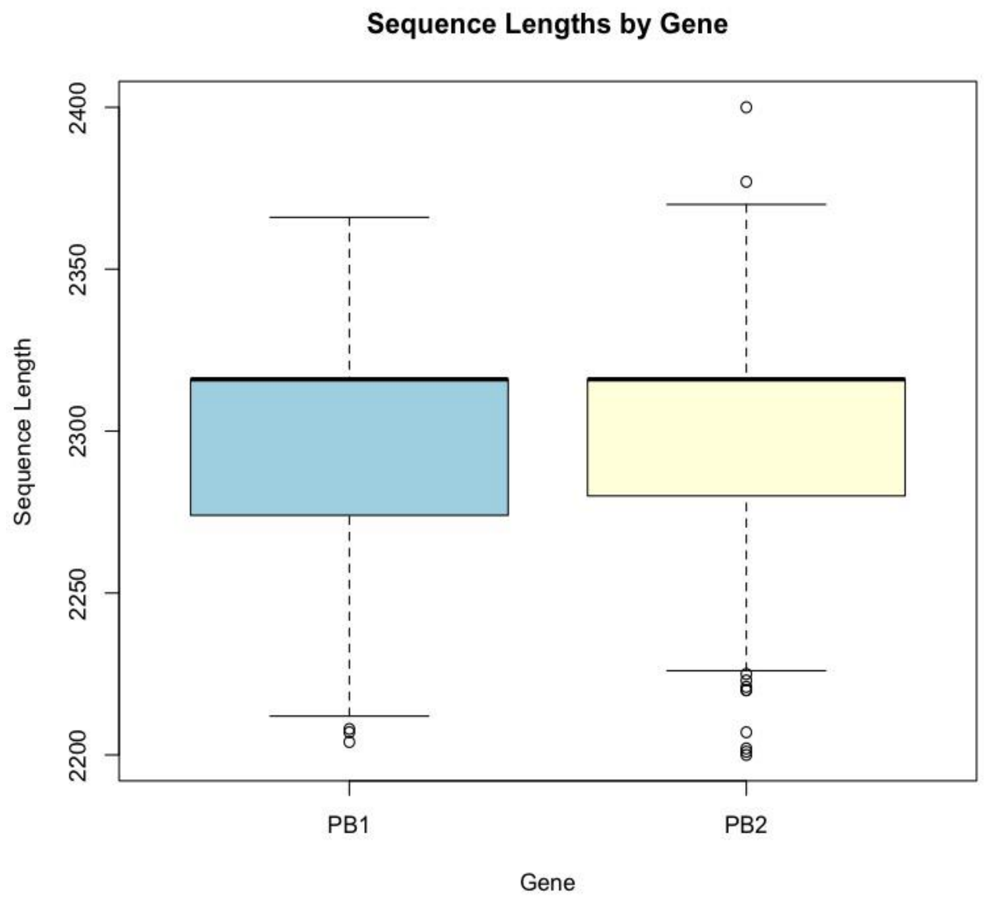
```r
# Adding a new column that indicates the gene name and a column that indicates the length of
each sequence:

dfPB2clean$Gene_Name <- "PB2"

dfPB2clean$Sequence_Length <- nchar(dfPB2clean$PB2_Sequence)

dfPB2clean <- dfPB2clean[, c("PB2_Title", "Gene_Name", "Species_Name", "PB2_Sequence",
"Sequence_Length")]

View(dfPB2clean)

dfPB2clean <- dfPB2clean %>%
  filter(!is.na(PB2_Sequence))

# There are no gaps or Ns in these sequences so I will not have to do that additional filtering
step to process the data. I have also specified the length of each sequence when I pulled the
data from NCBI.
# Next, I will be merging the PB1 data and PB2 data into a new data frame. First, I will be
changing the column names so that they are the same, this makes the merging step easier.

dfPB2clean2 <- dfPB2clean %>%
   rename(Sequence = PB2_Sequence, Title = PB2_Title)

dfPB1clean2 <- dfPB1clean %>%
   rename(Sequence = PB1_Sequence, Title = PB1_Title)
```

```
dfPBmerged <- rbind(dfPB1clean2, dfPB2clean2)
```

```
# Validating that there are no NAs in the data and that all sequences have a length of 2200-
2400:

sum(is.na(dfPBmerged$Sequence))

summary(nchar(dfPBmerged$Sequence))

summary(nchar(dfPBmerged$Sequence[dfPBmerged$Gene_Name == 'PB2']))

summary(nchar(dfPBmerged$Sequence[dfPBmerged$Gene_Name == 'PB1']))
```

```
# After summarizing the data based on gene name, I can see that the mean sequence lengths
for the genes are very similar. I did use the same filter when I pulled the data for both genes
from NCBI because it is documented that both proteins are 755-760 amino acids long.
```

```
# Below I am creating a box plot to visualize the results I got from using the summary function
above. This steps also allows me to validate these results and to see if there are any outliers.
Each gene will have its own box plot but will share the same y-axis.

boxplot(Sequence_Length ~ Gene_Name, data = dfPBmerged,

        main = "Sequence Lengths by Gene",

        xlab = "Gene",

        ylab = "Sequence Length",

        col = c("lightblue", "lightyellow"))

axis(2, at = seq(2200, 2400, by = 25), labels = seq(2200, 2400, by = 25))
```



# Based on the box plots, I can validate that the mean lengths are almost identical. I also see that PB2 has a longer maximum sequence length while PB1 has a lower minimum sequence length. Additionally, the PB2 gene has a few outliers that are longer than the maximum while PB1 does not. It's possible that the outliers identified in the box plots represent natural variation in the gene sequence. I will include these in the next part of my analysis, however, I may return to them later on. It is also possible some of these sequences have varying lengths due to indels.

# In this next step, I am making sure that dfPBmerged is a data frame and that the values in the sequence column are converted to a DNAStringSet set so that I can use the Biostrings package:

```
dfPBmerged <- as.data.frame(dfPBmerged)

dfPBmerged$Sequence <- DNAStringSet(dfPBmerged$Sequence)
```

```
# In this next step I am determining the nucleotide frequency as well as nucleotide proportions and kmer proportions for the DNA sequences in my data frame. This is important as it gives insights into the features of each sequence and can be used to train my classifier model.

dfPBmerged <- cbind(dfPBmerged, as.data.frame(letterFrequency(dfPBmerged$Sequence, letters = c("A", "C","G", "T"))))

dfPBmerged$Aprop <- (dfPBmerged$A) / (dfPBmerged$A + dfPBmerged$T + dfPBmerged$C + dfPBmerged$G)
dfPBmerged$Tprop <- (dfPBmerged$T) / (dfPBmerged$A + dfPBmerged$T + dfPBmerged$C + dfPBmerged$G)
dfPBmerged$Gprop <- (dfPBmerged$G) / (dfPBmerged$A + dfPBmerged$T + dfPBmerged$C + dfPBmerged$G)
dfPBmerged$Cprop <- (dfPBmerged$C) / (dfPBmerged$A + dfPBmerged$T + dfPBmerged$C + dfPBmerged$G)

# Adding 3-mer (trinucleotide) proportions for each sequence:

dfPBmerged <- cbind(dfPBmerged, as.data.frame(trinucleotideFrequency(dfPBmerged$Sequence, as.prob = TRUE)))

## Creating & Testing the Classifier Model -----

# Changing the DNAStringSet sequence data back into character type so I can use it to create my training and validation data sets.
dfPBmerged$Sequence <- as.character(dfPBmerged$Sequence)

# Creating a validation data set with 750 sequences for each gene (PB1 & PB2):
```

```
set.seed(123)

dfValidation <- dfPBmerged %>%
  group_by(Gene_Name) %>%
  sample_n(750)


table(dfValidation$Gene_Name)


# Creating a training data set with 1250 sequences for each gene filtering so that this data set
does not include any of the Titles that were used in the validation data set:

set.seed(321)


dfTraining <- dfPBmerged %>%
  filter(!Title %in% dfValidation$Title) %>%
  group_by(Gene_Name) %>%
  sample_n(1250)


table(dfTraining$Gene_Name)


# Creating 2 classifiers that use the training data to predict which gene each sequence
represents. The first one uses columns 6-9, which represent the nucleotide (A, T, G, C)
frequencies for each sequence. The second one uses columns 10-13, which represent the
nucleotide proportions for each sequence. I am using a ntree value of 500 here because the
data set is larger compared to others that I have worked with previously and to increase
predictive performance. I will use the classifier that has the best results to predict the gene
features for my validation data set.
```

```
gene_classifier2 <- randomForest::randomForest(x = dfTraining[, 6:9], y =
as.factor(dfTraining$Gene_Name), ntree = 500, importance = TRUE)

gene_classifier2


gene_classifier <- randomForest::randomForest(x = dfTraining[, 10:13], y =
as.factor(dfTraining$Gene_Name), ntree = 500, importance = TRUE)


gene_classifier


# The classifier results exceeded my expectations. I was expecting it to be more difficult to
distinguish the genes based on sequence properties since they are very similar in length and
have a lot of the same properties. The nucleotide frequency classifier produced 4 errors for PB1,
and 2 for PB2, while the second classifier had 0 errors.

# Using the classifier to predict the gene type for unseen data from the validation data set:


predictValidation <- predict(gene_classifier, dfValidation[, c(2, 10:13)])


validation_results <- table(observed = dfValidation$Gene_Name, predicted = predictValidation)


View(validation_results)


# Based on this table, the classifier is also capable of classifying unseen data. The classifier made
only 1 error and predicted 749 sequences correctly for PB1. For PB2, it predicted all 750
sequences correctly.

# To take this analysis one step further, I will check to see if the classifier can categorize
sequences base on two variables, gene name and species name. By adding the species name
component, there will be 4 distinct groups that each sequence can fall into: Influenza A PB1,
Influenza A PB2, Influenza B PB1, Influenza B PB2.
```

# Summarizing the data based on species name and gene name:

```
table(dfPBmerged$Species_Name, dfPBmerged$Gene_Name)
```

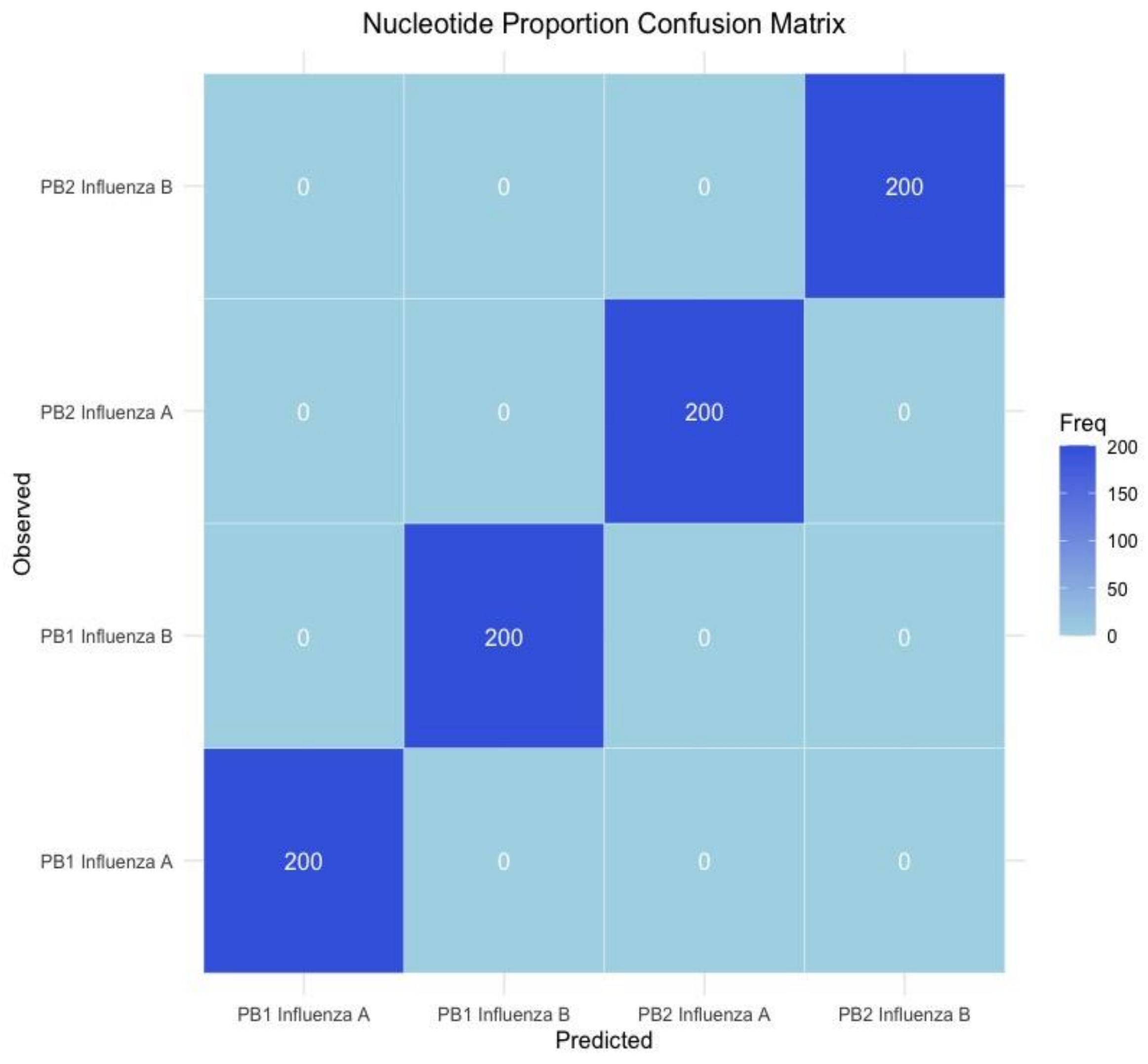
# I will use 300 sequences for each category for the training set and 200 sequences for the validation data set. I am filtering so that my validation and training data sets have only Influenza A and Influenza B sequences since there a few records with other species labels.

```
set.seed(111)
dfValidation2 <- dfPBmerged %>%
  filter(Species_Name == "Influenza A"|Species_Name == "Influenza B") %>%
  group_by(Gene_Name, Species_Name) %>%
  sample_n(200)

table(dfValidation2$Gene_Name,dfValidation2$Species_Name)

set.seed(222)
dfTraining2 <- dfPBmerged %>%
  filter(Species_Name == "Influenza A"|Species_Name == "Influenza B") %>%
  filter(!Title %in% dfValidation2$Title) %>%
  group_by(Gene_Name, Species_Name) %>%
  sample_n(300)

table(dfTraining2$Gene_Name,dfTraining2$Species_Name)
```

# Creating a new column that concatenates the gene name and species name into a single string since the classifier cannot take two columns:

```
dfTraining2$Combined <- paste(dfTraining2$Gene_Name, dfTraining2$Species_Name)
```

```
gene_classifier3 <- randomForest::randomForest(x = dfTraining2[, 10:13], y =
as.factor(dfTraining2$Combined), ntree = 500, importance = TRUE)


gene_classifier3


# The results from the training data look good, there was only 1 misclassification, where 1 PB1
Influenza A sequence was misclassified as a PB2 Influenza A sequence.


dfValidation2$Combined <- paste(dfValidation2$Gene_Name, dfValidation2$Species_Name)


predictValidation2 <- predict(gene_classifier3, dfValidation2[, c(2, 3, 10:13)])


validation_results2 <- table(observed = (dfValidation2$Combined), predicted =
predictValidation2)


# The classifier was able to predict the sequences perfectly with no error based on the gene
type and species name for unseen data from the validation data set.


confusion_df <- as.data.frame(validation_results2)


# Plotting the resulting data frame as a confusion matrix:
ggplot(data = confusion_df, aes(x = predicted, y = observed)) +
  geom_tile(aes(fill = Freq), color = "white") +
  scale_fill_gradient(low = "lightblue", high = "royalblue") +
  geom_text(aes(label = Freq), color = "white") +
  labs(title = "Nucleotide Proportion Confusion Matrix", x = "Predicted", y = "Observed") +
  theme_minimal() + theme(plot.title = element_text(hjust = 0.5))
```



# Since all the sequences were categorized correctly, the confusion matrix does not show a lot of variation in the colour gradient. However, it is a useful way to visualize how the predicted sequence results compare to the observed values.

## Discussion/Conclusion -----

# Overall, my initial hypothesis was incorrect, and the gene classifier I created was able to distinguish between PB1 and PB2 genes almost perfectly based on nucleotide proportion. Additionally, the second classifier I made was able to distinguish sequences by gene and species type with no errors. My analysis and results highlight the power of the random forest regression algorithm. Evidently, the length of a sequence is only one factor when it comes to understanding its function and the species it originates from. When sequences are analyzed using different properties such as nucleotide proportion or k-mers, this information can be used to train a model that predicts which group a new sequence would fall into. When looking at PB1 and PB2 specifically, the differences in their sequence properties can be tied to each protein's structural differences and distinct roles in viral replication. PB2 contains a cap-binding domain that is crucial for its role in mRNA synthesis, and PB1 is the main catalytic component responsible for polymerase activity within the RNA polymerase complex (Gorman et al., 1990).

# One of the potential limitations of this analysis is the quality of the data retrieved from NCBI GenBank. It is possible that species or gene names can be incorrectly assigned since it is a public repository. In this case, this was not a major concern because the classifiers were able to categorize sequences with minimal errors. However, in a different analysis, where a lot of values are being misclassified when training or validating a model, this could be a caveat. The classifiers I created for this assignment are limited in scope, If I were to expand on this project, I would include sequence data for multiple influenza species and a wide range of viral genes. This would allow me to train the model to predict a more complex combination of features. One use case for this would be to study which genes and species are present in a sample from an individual's microbiome. Additionally, these classifiers can be used as an informative step before doing any further analysis, depending on the specific biological question.

## Acknowledgements ----

# I would like to thank Brittany our amazing TA for taking a whole class to give us feedback on our first assignment and for all the individual feedback as well. This is very important so that we can all improve and become better bioinformaticians.

# I would also like to shout out Dr. Cottenie for being a great lecturer and providing us so many resources so that we can all excel in this course.

# Lastly, I would like to thank all my classmates for giving me panic attacks every time we talk about how our assignments are going (specifically Isha & Avery). In all seriousness, everyone in our class is amazing and they are always willing to discuss ideas and provide constructive feedback.

## References -----

## Bouvier, N. M., & Palese, P. (2008). The biology of influenza viruses. Vaccine, 26 Suppl 4(Suppl 4), D49–D53. https://doi.org/10.1016/j.vaccine.2008.07.039

## Gorman, O. T., Donis, R. O., Kawaoka, Y., & Webster, R. G. (1990). Evolution of influenza A virus PB2 genes: implications for evolution of the ribonucleoprotein complex and origin of human influenza A virus. Journal of virology, 64(10), 4893–4902. https://doi.org/10.1128/JVI.64.10.4893-4902.1990

## Hiromoto, Y., Saito, T., Lindstrom, S. E., Li, Y., Nerome, R., Sugita, S., Shinjoh, M., & Nerome, K. (2000). Phylogenetic analysis of the three polymerase genes (PB1, PB2 and PA) of influenza B virus. The Journal of general virology, 81(Pt 4), 929–937. https://doi.org/10.1099/0022-1317-81-4-929