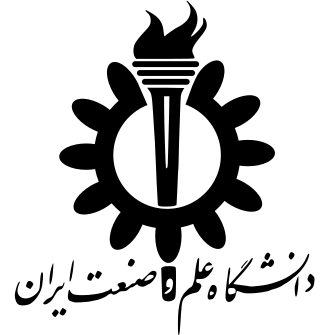
**به نام خدا**

****

**پروژه پایانی درس مبانی انفورماتیک زیستی**

**تحلیل و بصری‌سازی تغییرات تعداد کپی(CNV) در ژنوم انسان**

اعضای گروه:

هلیا شمس‌زاده امیری

بهاره کاوسی نژاد

مهدی معدنی‌پور

بهمن 1403

**مستندات پروژه: تغییرات تعداد کپی(CNV) و ارتباط آن‌ با ژن‌ها**

**بررسی کلی پروژه**

این پروژه به تحلیل تغییرات تعداد کپی DNA (CNVs) و ارتباط احتمالی آن‌ها با ژن‌های خاص می‌پردازد. اهداف شامل شناسایی الگوهای CNVs در داده‌های مورد و کنترل، انجام تحلیل‌های آماری برای برجسته‌سازی نواحی مهم CNV، نگاشت این نواحی به ژن‌ها و استفاده از یادگیری ماشین برای پیش‌بینی ارتباطات CNV و شناسایی ژن‌های کلیدی مرتبط با بیماری‌ها می‌باشد.

**اهداف پروژه**

1. شناسایی نواحی آماری معنادار CNV در داده‌های مورد و کنترل.
2. نگاشت نواحی معنادار به ژن‌های همپوشان بر اساس موقعیت‌های کروموزومی.
3. استفاده از یادگیری ماشین برای پیش‌بینی ارتباطات CNV و اولویت‌بندی ژن‌های مرتبط با بیماری.
4. ارائه بصری‌سازی‌ها و بینش‌های مربوط به الگوهای مشاهده‌شده.

**نیازمندی‌های داده**

* **داده‌های :** caseشامل CNVها از افراد مبتلا.
* **داده‌های :** controlشامل CNVها از افراد غیرمبتلا.
* **لیست ژن‌ها:** لیستی از ژن‌ها با موقعیت‌های کروموزومی آن‌ها.
* **ستون‌های کلیدی:**
  + داده‌های CNV: Chromosome، Start، End، Type، Patient\_ID.
  + داده‌های ژن: Gene\_ID، Chromosome، Gene\_Start، Gene\_End.

**جریان کاری پروژه**

**1. پیش‌پردازش داده‌ها**

* تغییر نام و استانداردسازی نام ستون‌ها در تمام مجموعه‌داده‌ها.
* حذف ستون‌های غیرمرتبط (مانند متادیتای غیرمورد استفاده).
* اعتبارسنجی یکپارچگی داده‌ها مانند نام‌های کروموزوم و محدوده‌های مختصات.

# Load the data

case\_data <- read\_csv('PC\_case.csv', col\_names = FALSE)

control\_data <- read\_csv('PC\_control.csv', col\_names = FALSE)

gene\_list <- read\_csv('geneList.csv', col\_names = FALSE)

# Rename columns for clarity

colnames(case\_data) <- c('Chromosome', 'Start', 'End', 'Type', 'Patient\_ID')

colnames(control\_data) <- c('Chromosome', 'Start', 'End', 'Type', 'Patient\_ID')

colnames(gene\_list) <- c('Gene\_ID', 'Unused', 'Chromosome', 'Gene\_Start', 'Gene\_End')

# Drop unused column in gene\_list

gene\_list <- gene\_list %>% select(-Unused)

**2. تحلیل آماری**

**گام 1: تجمیع داده‌ها**

* گروه‌بندی داده‌های CNV بر اساس Chromosome، Start، End و Type.
* شمارش رخدادها در موارد و کنترل‌ها.

# Group case and control data by chromosomal regions

case\_regions <- case\_data %>%

  group\_by(Chromosome, Start, End, Type) %>%

  summarise(Case\_Count = n(), .groups = 'drop')

control\_regions <- control\_data %>%

  group\_by(Chromosome, Start, End, Type) %>%

  summarise(Control\_Count = n(), .groups = 'drop')

**گام 2: آزمون دقیق فیشر**

* ادغام نواحی موارد و کنترل‌ها.
* محاسبه معناداری هر ناحیه با استفاده از آزمون دقیق فیشر.

# Merge case and control regions

merged\_regions <- full\_join(case\_regions, control\_regions, by = c('Chromosome', 'Start', 'End', 'Type')) %>%

  replace\_na(list(Case\_Count = 0, Control\_Count = 0))

# Calculate total case and control patient counts

total\_case\_patients <- n\_distinct(case\_data$Patient\_ID)

total\_control\_patients <- n\_distinct(control\_data$Patient\_ID)

# Perform Fisher's exact test

fisher\_test <- function(case\_count, control\_count, total\_case, total\_control) {

  case\_absence <- total\_case - case\_count

  control\_absence <- total\_control - control\_count

  contingency\_table <- matrix(c(case\_count, control\_count, case\_absence, control\_absence), nrow = 2)

  p\_val <- fisher.test(contingency\_table, alternative = "greater")$p.value

  return(p\_val)

}

merged\_regions <- merged\_regions %>%

  rowwise() %>%

  mutate(P\_Value = fisher\_test(Case\_Count, Control\_Count, total\_case\_patients, total\_control\_patients))

# Filter significant regions

significant\_regions <- merged\_regions %>%

  filter(P\_Value < 0.05)

این قسمت از کد برای تحلیل داده‌های تغییرات تعداد کپی (CNV) در بیماران مورد و شاهد طراحی شده است. ابتدا داده‌های مناطق CNV در دو گروه بر اساس کروموزوم، موقعیت شروع، پایان و نوع، ادغام شده و مقادیر خالی به صفر تبدیل می‌شوند. سپس تعداد کل بیماران در هر گروه محاسبه می‌شود. برای هر منطقه، با استفاده از آزمون دقیق فیشر، بررسی می‌شود که آیا حضور تغییرات CNV در گروه مورد به طور معناداری بیشتر از گروه شاهد است یا خیر. نتایج این آزمون به صورت مقدار احتمال (P-Value) ذخیره می‌شوند که نشان‌دهنده معناداری آماری تفاوت‌ها است.

**3. نگاشت نواحی معنادار به ژن‌ها**

* شناسایی ژن‌های همپوشان برای هر ناحیه CNV معنادار.
* ذخیره ارتباطات ژنی به صورت لیستی جداشده با ";"

# Map significant regions to genes

map\_to\_genes <- function(chromosome, start, end, genes) {

  overlapping\_genes <- genes %>%

    filter(Chromosome == paste0('chr', chromosome),

           Gene\_End >= start,

           Gene\_Start <= end)

  paste(overlapping\_genes$Gene\_ID, collapse = ";")

}

significant\_regions <- significant\_regions %>%

  rowwise() %>%

  mutate(Associated\_Genes = map\_to\_genes(Chromosome, Start, End, gene\_list))

# Save results to a CSV file

write\_csv(significant\_regions, 'Significant\_Regions\_with\_Genes.csv')

# Print completion message

print("Significant regions with associated genes saved to 'Significant\_Regions\_with\_Genes.csv'.")

در ابتدا، تابع map\_to\_genes بررسی می‌کند که هر منطقه در کدام کروموزوم قرار دارد و بازه شروع و پایان آن چیست. سپس ژن‌هایی که موقعیت آن‌ها با بازه منطقه همپوشانی دارند، از دیتافریم ژن‌ها استخراج می‌شوند. شناسه ژن‌های همپوشان به صورت رشته‌ای که با ";" جدا شده است، بازگردانده می‌شود. این تابع برای تمام مناطق مهم اعمال شده و ژن‌های مرتبط در ستون جدیدی به نام Associated\_Genes ذخیره می‌شوند. در نهایت، داده‌های مناطق مهم به همراه ژن‌های مرتبط در یک فایل CSV ذخیره می‌شوند.

1. **بصری‌سازی نتایج**

# Visualizations

library(ggplot2)

**نمودار میله‌ای: تعداد موارد در مقابل کنترل‌ها بر اساس کروموزوم**

# Bar Plot: Case vs Control Counts

ggplot(significant\_regions, aes(x = Chromosome)) +

  geom\_bar(aes(y = Case\_Count), stat = "identity", fill = "blue", alpha = 0.6) +

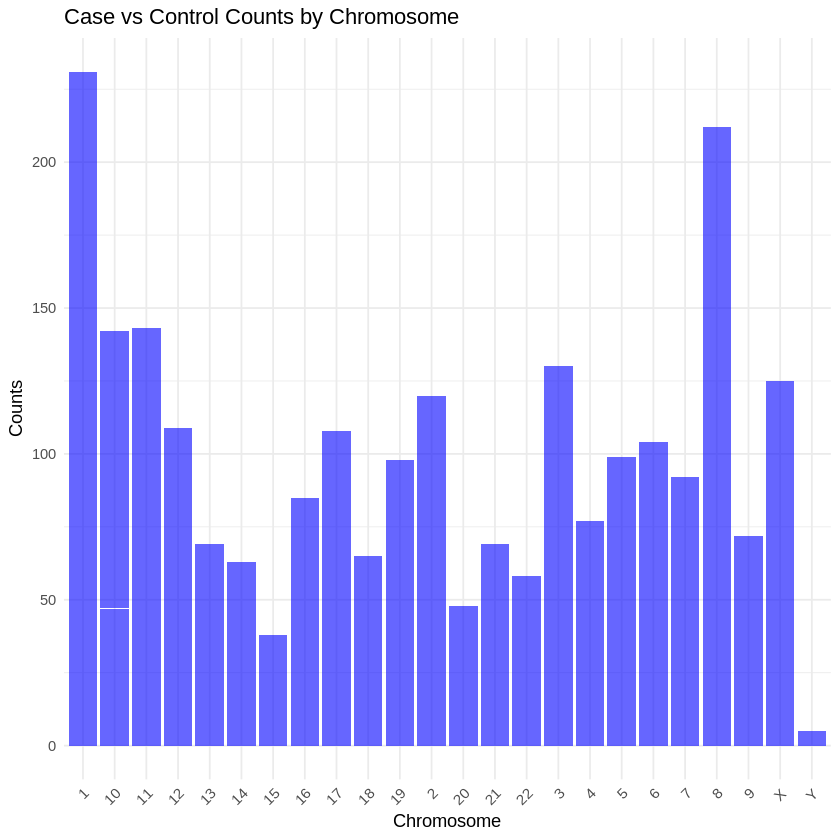
  geom\_bar(aes(y = Control\_Count), stat = "identity", fill = "orange", alpha = 0.6) +

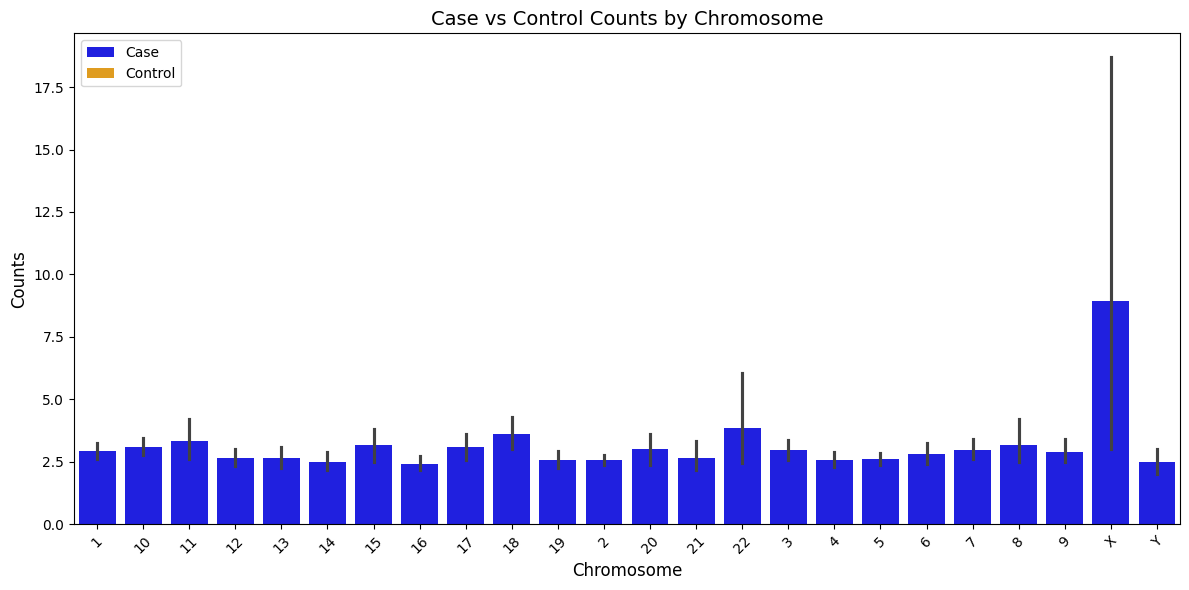
  labs(title = "Case vs Control Counts by Chromosome", y = "Counts", x = "Chromosome") +

  theme\_minimal() +

  theme(axis.text.x = element\_text(angle = 45, hjust = 1))

**نتایج**:





**نمودار پراکندگی: موقعیت کروموزومی در مقابل مقدار P**

# Scatter Plot: Chromosome Position vs P-Value

significant\_regions <- significant\_regions %>%

  mutate(Log\_P\_Value = -log10(P\_Value))

ggplot(significant\_regions, aes(x = Start, y = Log\_P\_Value, color = as.factor(Chromosome))) +

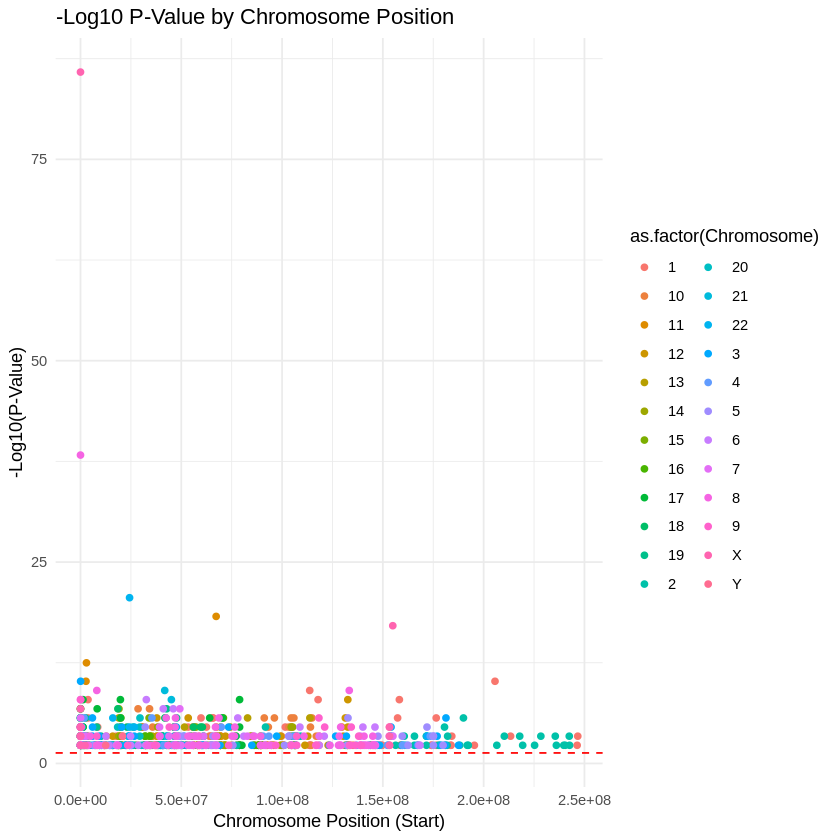
  geom\_point() +

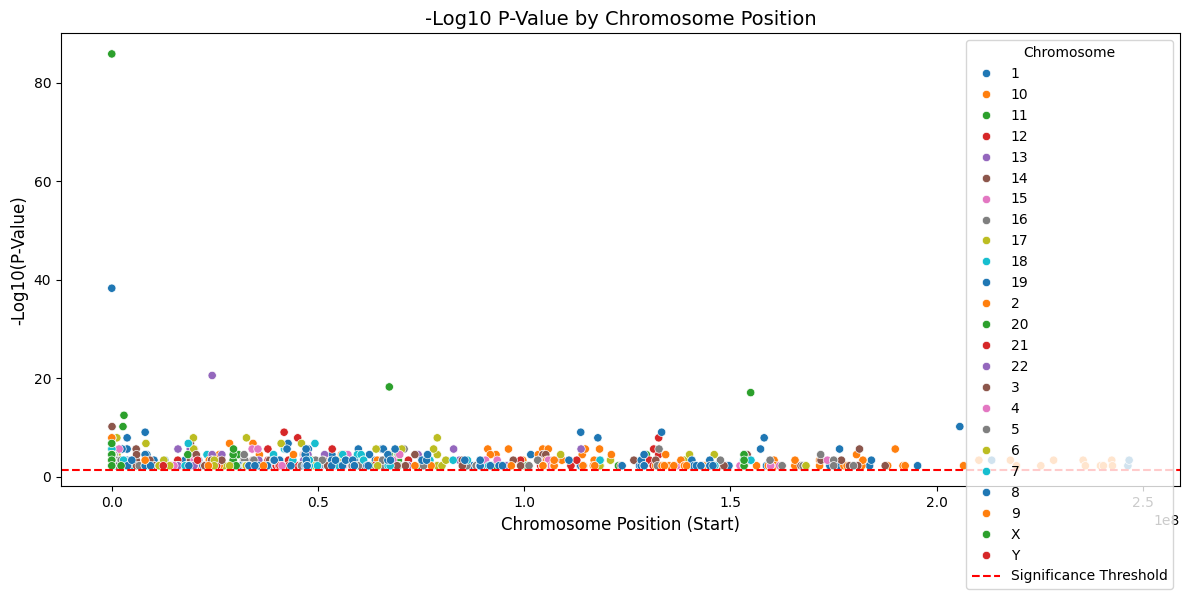
  geom\_hline(yintercept = -log10(0.05), linetype = "dashed", color = "red") +

  labs(title = "-Log10 P-Value by Chromosome Position", y = "-Log10(P-Value)", x = "Chromosome Position (Start)") +

  theme\_minimal()

**نتایج**:





**5. Machine learning pipeline**

**ایجاد ویژگی‌ها**

* ترکیب داده‌های مورد و کنترل با یک ستون target.
* نگاشت نواحی CNV به ژن‌های مرتبط.
* کدگذاری ویژگی‌های دسته‌ای.

# Combine case and control data

case\_data <- mutate(case\_data, target = 1)

control\_data <- mutate(control\_data, target = 0)

combined\_data <- bind\_rows(case\_data, control\_data)

# Map significant regions to genes

map\_to\_genes <- function(chromosome, start, end, genes) {

  overlapping\_genes <- genes %>%

    filter(Chromosome == paste0("chr", chromosome),

           Gene\_End >= start,

           Gene\_Start <= end)

  paste(overlapping\_genes$Gene\_ID, collapse = ";")

}

combined\_data <- combined\_data %>%

  rowwise() %>%

  mutate(Associated\_Genes = map\_to\_genes(Chromosome, Start, End, gene\_data))

# Export the combined data to a CSV file

write\_csv(combined\_data, "combined\_data.csv")

# Feature engineering

combined\_data <- combined\_data %>%

  mutate(VariationLength = End - Start) %>%

  select(-Patient\_ID)

# One-Hot Encoding

encoded\_data <- combined\_data %>%

  mutate(across(c(Chromosome, Type, Associated\_Genes), as.factor)) %>%

  model.matrix(~ . - 1, data = .) %>%

  as.data.frame()

**آموزش مدل**

* آموزش و ارزیابی مدل جنگل تصادفی.

# Train a model

set.seed(42)

X <- encoded\_data %>% select(-target)

y <- encoded\_data$target

train\_index <- createDataPartition(y, p = 0.8, list = FALSE)

X\_train <- X[train\_index, ]

X\_test <- X[-train\_index, ]

y\_train <- y[train\_index]

y\_test <- y[-train\_index]

model <- randomForest(X\_train, as.factor(y\_train))

# Evaluate the model

y\_pred <- predict(model, X\_test)

confusion\_matrix <- confusionMatrix(as.factor(y\_pred), as.factor(y\_test))

print(confusion\_matrix)

# Feature importance

feature\_importance <- data.frame(Feature = colnames(X\_train), Importance = importance(model)) %>%

  arrange(desc(Importance))

# Identify significant genes

gene\_columns <- grep("Associated\_Genes\_", colnames(encoded\_data), value = TRUE)

significant\_genes <- encoded\_data %>%

  filter(target == 1) %>%

  select(all\_of(gene\_columns)) %>%

  summarise(across(everything(), sum)) %>%

  pivot\_longer(everything(), names\_to = "Gene", values\_to = "Count") %>%

  arrange(desc(Count))

write\_csv(significant\_genes, "significant\_genes.csv")

# Gene counts

significant\_genes <- read\_csv("significant\_genes.csv")

# Split the genes and count occurrences

gene\_counts <- significant\_genes %>%

  separate\_rows(Gene, sep = ";") %>%

  count(Gene, name = "Count") %>%

  arrange(desc(Count))

write\_csv(gene\_counts, "gene\_counts.csv")

# Plot the top 20 genes with the highest counts

top\_genes <- head(gene\_counts, 20)

ggplot(top\_genes, aes(x = reorder(Gene, Count), y = Count)) +

  geom\_bar(stat = "identity", fill = "skyblue") +

  coord\_flip() +

  labs(title = "Top 20 Genes with Highest Counts", x = "Gene", y = "Count") +

  theme\_minimal()

برای ایجاد و ارزیابی یک مدل یادگیری ماشین با استفاده از Random Forest Classifierابتدا داده‌های ورودی (ویژگی‌ها) و خروجی (هدف) از دیتافریم combined\_data\_encoded استخراج می‌شوند. ستون هدف (target) به عنوان برچسب کلاس مشخص شده و سایر ستون‌ها به عنوان ویژگی‌ها (X) در نظر گرفته می‌شوند. سپس داده‌ها به دو بخش آموزش (80%) و آزمون (20%) تقسیم می‌شوند.

مدل Random Forest Classifier ایجاد شده و با داده‌های آموزشی ، آموزش داده می‌شود. پس از آن، مدل برای پیش‌بینی کلاس‌ها در داده‌های آزمون (X\_test) استفاده می‌شود و خروجی پیش‌بینی‌شده (y\_pred) با مقادیر واقعی (y\_test) مقایسه می‌شود. در نهایت، متریک‌های ارزیابی مدل شامل دقت (Precision)، بازخوانی (Recall)، امتیاز F1 و دقت کل (Accuracy) با استفاده از تابع classification\_report چاپ می‌شوند. این ارزیابی نشان می‌دهد که مدل چقدر در دسته‌بندی داده‌ها عملکرد خوبی داشته است.

نتایج:

