Week 3 - Data Viz. & Intro to Statistical Inference

Chris Lo

6/23/2020

Package list: (Use the library function to load in additional libraries)

If you're missing packages, run install.packages("packagename").

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.2.1     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Loading the data

We're loading the same dataset from Homework 2.

```
# Unlike last time where we used the read.delim() function of base R,
# we'll be using the readr package's read_csv() function.
CCLE_metadata <- read_csv("sample_info.csv")

# These next lines might take a little while to run - try not to run them too many times!
CCLE_expression <- read_csv("CCLE_expression-subset.csv")
CCLE_mutations <- read_csv("CCLE_mutations-subset.csv")
```

```
# Our goal here is to extract the relevant parts from the mutation data
# to merge it with the CCLE_metadata.

kras_mutations <- CCLE_mutations %>%
  # Filter for only KRAS mutations
  filter(Hugo_Symbol == "KRAS") %>%
  # Select for the columns we need to merge our data
  select(Hugo_Symbol, DepMap_ID, Protein_Change, Codon_Change)

# To make our lives easier, we're going to merge the CCLE_metadata
# with the kras_mutations data. By doing this, we're annotating
```

```r
# each cell line with the Protein_Change or Codon_Change in KRAS that
# it has.
CCLE_metadata <- merge(CCLE_metadata, kras_mutations, all.x = T)
```

```r
# Create a new CCLE_combined dataframe that brings together the metadata and expression data.
CCLE_combined <- CCLE_expression %>%
  # Rename the 1st column to DepMap_ID (because that's what it actually represents)
  # This will also allow the merge function to merge by the DepMap_ID.
  rename(DepMap_ID = X1) %>%
  # Use the merge function to combine all of the expression data with the metadata
  merge(CCLE_metadata) %>%
  # Add a column of TRUE or FALSE, indicating if KRAS is mutated or not
  # If codon change is NOT NA (the ! means NOT), then set the value to TRUE.
  mutate(KRAS_mutated = !is.na(Codon_Change)) # *** #

KRAS_data <- CCLE_combined %>%
  # Pick the columns of interest (renaming the KRAS data to KRAS_expression)
  select(KRAS_expression = `KRAS (3845)`, CCLE_Name, KRAS_mutated, lineage)
```

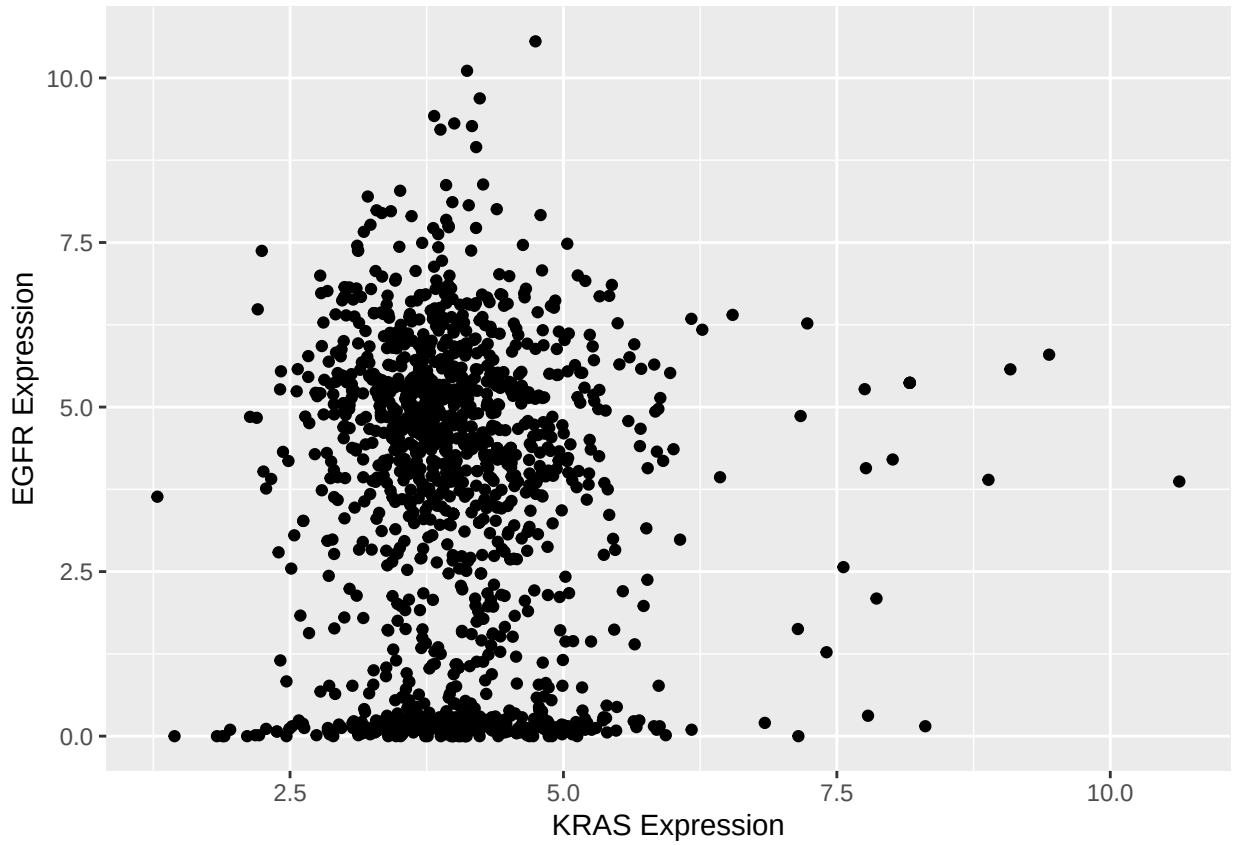
Part 1: More practice with tidyverse, ggplot

Q1. Two perspectives to look at expression of two genes, KRAS and EGFR.

- Using the dataframe `CCLE_combined`, create a scatterplot between KRAS expression and EGFR expression.
- Then, create a boxplot comparing KRAS expression and EGFR expression.
- These are two visualizations to display *bivariate* (as opposed to univariate data or multivariate data). In what circumstance might you use one over the other?
- Bonus: Can you figure out how to labels and axis nicer to explain what you are displaying clearly?

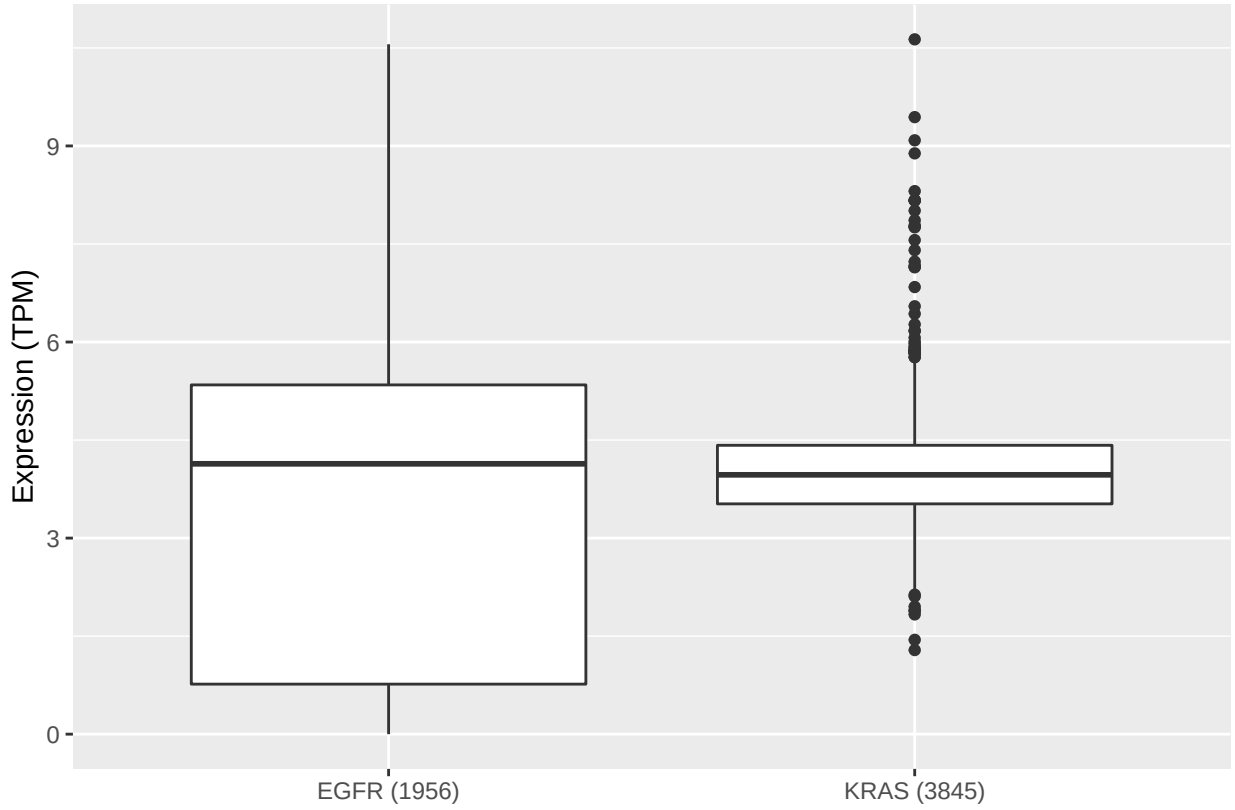
```r
#An an important aspect of dealing with column names with spaces: how does one select them?
#CCLE_combined$`KRAS (3845)` #it is done using the back-tick symbol, to the left of your 1 on your keyb
#select(CCLE_combined, `KRAS (3845)`) #this applies to Tidyverse and ggplot.

ggplot(CCLE_combined, aes(x = `KRAS (3845)`, y = `EGFR (1956)`)) + geom_point() + labs(x = "KRAS Expres
```




```
CCLE_combined_long <- CCLE_combined %>%
  select(`KRAS (3845)`, `EGFR (1956)`) %>%
  gather(gene, expression, `KRAS (3845)`, `EGFR (1956)`)

ggplot(CCLE_combined_long, aes(x = gene, y = expression)) + geom_boxplot() + labs(x = "", y = "Expressi
```



Part 2: Understanding how the T-test really works.

Q2. We are going to expand on Q3 from last homework. Perform a t-test to see if there is a difference in KRAS expression between KRAS wild-type and KRAS mutant cell lines using the dataframe `KRAS_data`. Use the function `t.test` can take two numeric vectors and perform a t.test between them. Look at the results: Explain what the t-statistic means, what the null hypothesis is, what the p-value means, and whether you reject the null hypothesis using a cutoff of .05. What is the effect size (difference in mean)?

```
t.test(KRAS_data$KRAS_expression[KRAS_data$KRAS_mutated == TRUE],
       KRAS_data$KRAS_expression[KRAS_data$KRAS_mutated == FALSE])
```

```
##
##  Welch Two Sample t-test
##
## data:  KRAS_data$KRAS_expression[KRAS_data$KRAS_mutated == TRUE] and KRAS_data$KRAS_expression[KRAS_
## t = 5.4772, df = 352, p-value = 8.238e-08
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.2293811 0.4863982
## sample estimates:
## mean of x mean of y
##  4.335247  3.977358
```

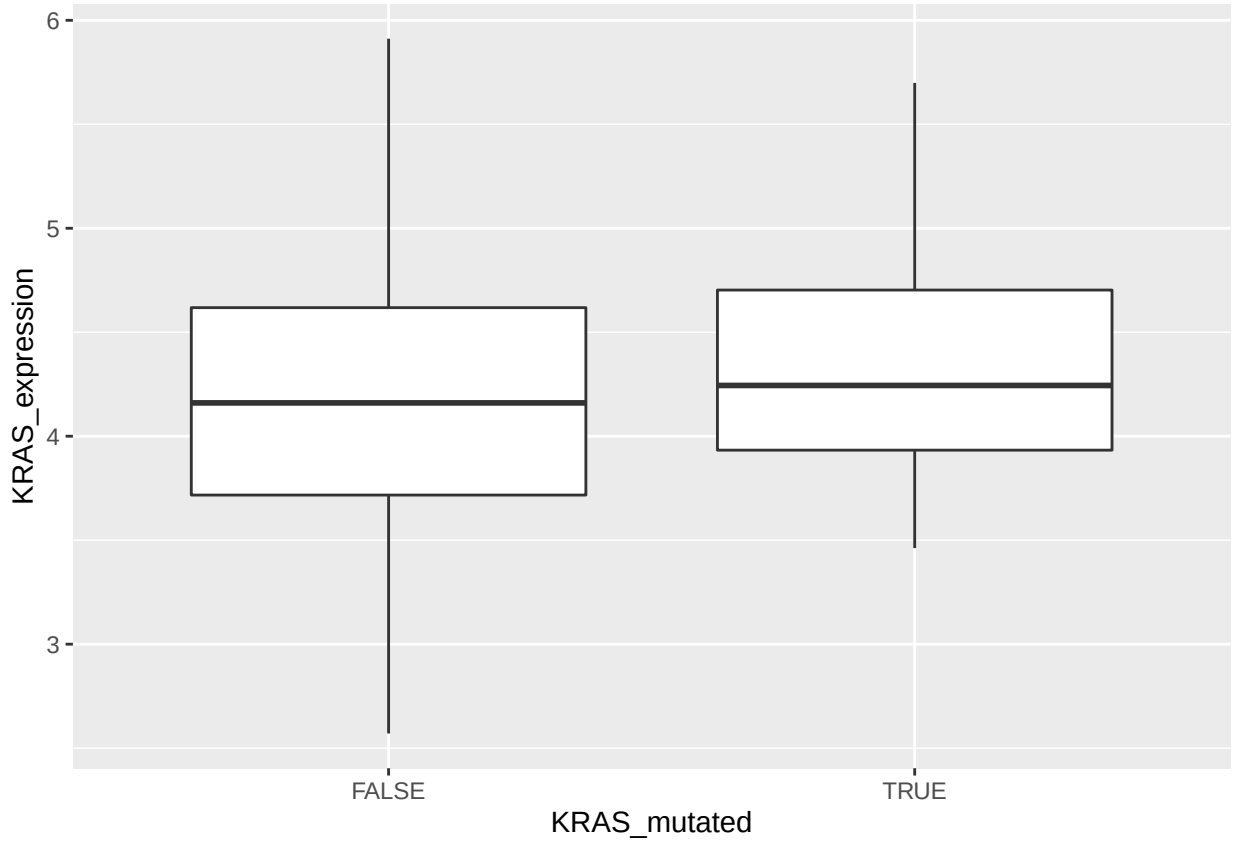
The null hypothesis is that the difference between the two group's mean is 0. The t-statistic is a random variable under the assumption that the null hypothesis is true. It follows the T-distribution. The p-value is

the probability of observing the result *assuming* that the null hypothesis is true. If this probability is small, then there is strong evidence that the the null hypothesis is false, favoring the alternative hypothesis that there is a difference between the two group's mean. The effect size is 4.33 - 3.97 = .36

Q3. Now, we want to narrow the scope to colorectal cell lines: within colorectal cell lines, is KRAS expression different between KRAS KRAS wild-type and mutant?

- Perform a T-test. Interpret the p-value: do you reject the null hypothesis using a cutoff of .05? If you don't reject, what can you say about the data? What is the effect size?

```
ggplot(KRAS_data %>% filter(lineage == "colorectal"), aes(KRAS_mutated, KRAS_expression)) + geom_boxplo
```




```
t.test(KRAS_data$KRAS_expression[KRAS_data$KRAS_mutated == TRUE & KRAS_data$lineage == "colorectal"],
       KRAS_data$KRAS_expression[KRAS_data$KRAS_mutated == FALSE & KRAS_data$lineage == "colorectal"])
```

```
##
##  Welch Two Sample t-test
##
## data:  KRAS_data$KRAS_expression[KRAS_data$KRAS_mutated == TRUE & KRAS_data$lineage ==  and KRAS_data
## t = 0.81984, df = 46.175, p-value = 0.4165
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.1963419  0.4662310
## sample estimates:
## mean of x mean of y
##  4.293837  4.158892
```

p value is greater than .05. Failing to reject the null indicates that our sample did not provide sufficient evidence to conclude that the effect exists. However, at the same time, that lack of evidence doesn't prove that the effect does not exist. It may be that the sample size is too small, or by chance the sample didn't show the effect size. 4.29 - 4.15 = .14

Q4. Now, we are going to compute the T-test by hand to understand better how it works. Using the EdX module "Central Limit Theorem", section "t-tests in Practice" as a guidance,

- Compute the T-statistic for Q2. How does it compare to your answer in Q2?
- Then, see where the T-statistic lie relative to the null hypothesis distribution of T distribution to compute the p-value. How does it compare to your answer in Q2? (hint: in the sections online, the T-statistic is compared to Normal(mean=0, sd=1), but the T-statistic should technically compared to T distribution(mean=0, sd=1). To do this properly, use `pt(t_statistic, df = 46)`, where `t_statistic` is your answer to the first part of the question).
- BONUS: Let's see this visually: take 10,000 random samples from the T distribution using the command `rt(n = 1000, df = 46)`, plot a histogram using ggplot, and use `geom_vline(xintercept = ___)` to create a vertical line where the T-statistic lie. Describe this graph you created: visually, where is the null hypothesis, where is the T-statistic, and where is the p-value?

```
set.seed(666)
mutated_exp <- KRAS_data$KRAS_expression[KRAS_data$KRAS_mutated == TRUE]
wt_exp <- KRAS_data$KRAS_expression[KRAS_data$KRAS_mutated == FALSE]


obs_mean <- mean(mutated_exp)
wt_mean <- mean(wt_exp)
se <- sqrt(var(mutated_exp) / length(mutated_exp) + var(wt_exp) / length(wt_exp))
t_statistic <- (obs_mean - wt_mean) / se
print(paste0("Computed t-statistic: ", t_statistic))
```

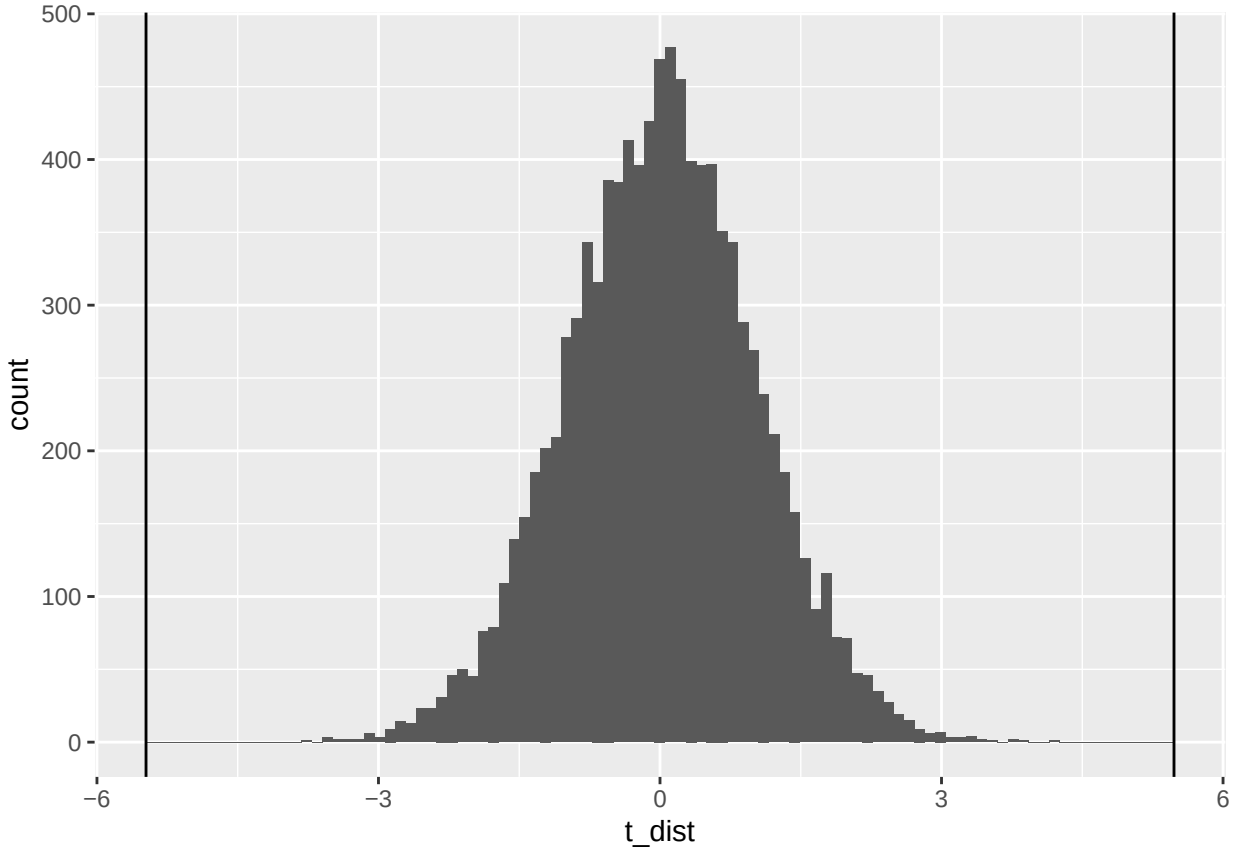
```
## [1] "Computed t-statistic: 5.4772321925207"
```

```
p_value <- (1 - pt(t_statistic, df = 46)) * 2

print(paste0("Computed p-value: ", p_value))
```

```
## [1] "Computed p-value: 1.74505421468574e-06"
```

```
null_distribution <- data.frame(t_dist = rt(n = 10000, df = 46))
ggplot(null_distribution, aes(x = t_dist)) + geom_histogram(bins = 100) + geom_vline(xintercept = t_sta
```



The distribution represents the null hypothesis. The vertical lines represents the T-statistic, one for each tail, and any area to the right and left of each T-statistic is the p-value.