Code

Correlation of GCMS Analysis Date with Harvest Date

Library Imports

Hide

```
library(readxl)
library(stringr)
library(dplyr)
library(questionr)
library(xlsx)
library(ggplot2)
```

Problem

We found that there is correlation between harvest date and samples total volatile content (STVC). We would like to know whether this is a true biological phenomenon or whether it is an artifact of some sort. To do that, we would like to see if there is a correlation between the date the volatile data was collected and the harvest date. If we see a high correlation between the two values, this means that harvest date is largely explained by the date on which the volatile was quantified. If that is the case, we do not have a strong position to propose that there is a biological relationship between harvest date and other attributes.

Data Loading

We have information about when (i.e., `YYYY-MM-DD` formatted date) the samples were analyzed on the GC-MS machine. We will be loading that.

Hide

```
date_data <- read_excel('data/raw/apple_quality_2017_analysis_date.xlsx')
dim(date_data)
```

```
[1] 140830      3
```

There are 3 columns and 140830 rows.

Data Check & Curation

Let’s see what the column names are for the data:

Hide

```
colnames(date_data)
```

```
[1] "Peak Number"      "Date of analysis" "FileName"
```

```
Peak Number

Date of analysis

FileName
```

We need to change the column names so that they are easier to work with. Also, we probably do not need all of these columns. We only really need the `Date` and `FileName` column.

Hide

```
colnames(date_data) <- c("Peak_Number", "Date", "FileName")
date_data <- date_data[, -1]
colnames(date_data)
```

```
[1] "Date"     "FileName"
```

```
Date

FileName
```

There we go! We only have the `Date` and `FileName` column now.

Investigating Data Format

Let's take a look at the data in the file to see what the format is:

Hide

```
head(date_data)
```

Date <S3: POSIXct>	FileName <chr>
2018-04-18	AAFC Apple Quality Volatiles 2016 Blank1_1.csv
2018-04-19	AAFC Apple Quality Volatiles 2016 Blank1_2.csv
2018-04-20	AAFC Apple Quality Volatiles 2016 Blank1_3.csv
2018-04-23	AAFC Apple Quality Volatiles 2016 Blank1_4.csv
2018-04-24	AAFC Apple Quality Volatiles 2016 Blank1_5.csv
2018-04-19	AAFC Apple Quality Volatiles 2016 Blank2_1.csv

6 rows

It looks like the date is `YYY-MM-DD` format. The FileName is the same FileName that we had previously from which we extracted the nursery id and then finally extracted the `apple id`.

Removing unwanted rows from the `date_data`

The rows with Blanks in the `FileName` column are the runs where instead of apple samples, Blanks were run as part of calibration process. There are other unnecessary rows that are not interesting. I am going to remove it before proceeding. In other words, I am only going to be keeping the rows which have the `eunit` keyword in them because these rows are the ones that truly correspond to a nursery id and an apple id.

Hide

```
rows_to_keep <- which(grepl("eunit", date_data$FileName))
length(rows_to_keep)
```

```
[1] 123491
```

There are `123491` rows which we will be keeping, the rest are going to be thrown away.

Hide

```
date_data <- date_data[rows_to_keep,]
nrow(date_data)
```

```
[1] 123491
```

After cleanup, we end up with `123491` rows.

Final check to see what the `head` of the data looks like:

Hide

```
head(date_data)
```

Date <S3: POSIXct>	FileName <chr>
2018-04-18	AAFC Apple Quality Volatiles 2016 eunit 1002_1.csv
2018-04-19	AAFC Apple Quality Volatiles 2016 eunit 1009_1.csv
2018-04-18	AAFC Apple Quality Volatiles 2016 eunit 1010_1.csv
2018-04-19	AAFC Apple Quality Volatiles 2016 eunit 1014_1.csv
2018-04-18	AAFC Apple Quality Volatiles 2016 eunit 1018_1.csv
2018-04-19	AAFC Apple Quality Volatiles 2016 eunit 1023_1024_1.csv

6 rows

Investigating the correctness of date format

There are supposed to be `123491` rows in the `date_data` data frame. Therefore, if I match the format `YYYY-MM-DD` in the Date column, I should get exactly that number of records if all the dates have the same format.

Hide

```
correct_dates <- str_extract(date_data$Date, regex("[0-9]{4}-[0-9]{2}-[0-9]{2}"))
length(correct_dates)
```

```
[1] 123491
```

The number of rows in the data `123491` matches the number of correctly formatted dates (`123491`). Therefore, I conclude that the dates are corectly formatted.

Investigating the correctness of FileName format

In the FileName, I expect all the rows to have an `e-unit` ID (which is used to get to the apple id). Therefore, a good check would be to verify that all the rows have `e-unit` value.

To do this, I am going to match the `FileName` value to a regular expression.

Hide

```
correct_file_names <- str_extract(date_data$FileName, regex('eunit [0-9]{1,}_(1|[0-9]
{1,})'))
length(correct_file_names)
```

```
[1] 123491
```

The number of rows in the data `123491` matches the number of correctly formatted `FileName` values (`123491`). Therefore, I conclude that the `FileName` data are correctly formatted.

Joining apple ids to the data frame

In order to get to the apple ids, I first need to extract the nursery ids from `FileName` column. I do that as follows:

Hide

```
# parsing out the nursery ids (A.K.A eunit ids)
nurseryIds <- gsub('eunit ', '', str_extract(date_data$FileName, 'eunit [0-9]{1,}'))
length(nurseryIds)
```

```
[1] 123491
```

Hide

```
head(nurseryIds)
```

```
[1] "1002" "1009" "1010" "1014" "1018" "1023"
```

```
1002

1009

1010

1014

1018

1023
```

Now I have vector of 123491 rows which corresponds to the nursery ids for each of the row in the `date_data` data frame. I will add that to the data frame:

Hide

```
date_data$NurseryId <- nurseryIds
```

Now, the `FileName` column is no longer required. I will remove that so that we only have the `Date` and `NurseryId` column.

Hide

```
date_data <- date_data[, c("Date", "NurseryId")]
head(date_data)
```

Date <S3: POSIXct>	NurseryId <chr>
2018-04-18	1002
2018-04-19	1009
2018-04-18	1010
2018-04-19	1014
2018-04-18	1018
2018-04-19	1023

6 rows

There are going to be lots of duplicated rows because many compounds were analyzed for a single nursery id. And all the nursery ids were analyzed on the same day. Therefore, I need to remove the duplicated rows and only keep the unique ones.

Hide

```
nrow(date_data)
```

```
[1] 123491
```

Hide

```
nrow(unique(date_data))
```

```
[1] 670
```

There are 123491 rows, but most of them are duplicates. If we remove duplicated rows, we end up with 670 rows.

Removing duplicated rows:

Hide

```
date_data <- unique(date_data)
```

The next job is to attach apple ids to the data frame. I do this through a pivot table that I generated a while back which maps nursery ids to apple ids. I am going to join that pivot table to the `date_data` data frame.

Hide

```
pivot_tbl <- read.table('data/raw/nursery-id_apple-id_pivot.tsv')

head(pivot_tbl)
```

	apple_id <int>	nursery_id <int>
1	390	5010
2	390	5011
3	546	6160
4	546	6162
5	411	5072
6	411	5074

6 rows

Hide

```
# make sure that both columns are character so that it matches the class of date_data
pivot_tbl$nursery_id <- as.character(pivot_tbl$nursery_id)
pivot_tbl$apple_id <- as.character(pivot_tbl$apple_id)
```

Hide

```
nrow(pivot_tbl)
```

```
[1] 2356
```

There are 2356 rows. We also notice from above that there are duplicated apple ids. The reason is that there are multiple nursery ids that correspond to a single apple ids. This duplication should not be a problem, because we are matching the nursery id so regardless of which of the two nursery ids we end up picking, they all correspond to the same apple id.

Hide

```
final.df <- inner_join(date_data, pivot_tbl, by = c("NurseryId" = "nursery_id"))
head(final.df)
```

Date <S3: POSIXct>	NurseryId <chr>	apple_id <chr>
2018-04-18	1002	1
2018-04-19	1009	3
2018-04-18	1010	4
2018-04-19	1014	5
2018-04-18	1018	6
2018-04-19	1023	8

6 rows

Hide

```
nrow(final.df)
```

```
[1] 670
```

The final number (670) is exactly the same as the initial number (670) that we had for `date_data`.

Joining the harvest date data

Hide

```
abc_pheno_tbl <- read_excel('data/processed/sup_tbl_2-abc_phenotype_table_v2.xlsx')

#we only need to keep certain columns
abc_pheno_tbl <- abc_pheno_tbl[, c("apple_id", "date_jul_17_harv")]

# convert the apple_id to character so that it is comparable to the column in final.df
abc_pheno_tbl$apple_id <- as.character(abc_pheno_tbl$apple_id)

head(abc_pheno_tbl)
```

apple_id <chr>	date_jul_17_harv <dbl>
390	282.6511

apple_id <chr>	date_jul_17_harv <dbl>
411	276.2631
245	249.9314
505	260.0280
395	232.8948
571	272.1852

6 rows

Perfect, the above data looks good, the next step is to attach this data to the `final.df`

Hide

```
final.df <- inner_join(final.df, abc_pheno_tbl, by = "apple_id")
head(final.df)
```

Date <S3: POSIXct>	NurseryId <chr>	apple_id <chr>	date_jul_17_harv <dbl>
2018-04-19	1009	3	271.7881
2018-04-20	2023	108	246.5290
2018-04-20	2029	110	225.7553
2018-04-20	2041	1268	259.3719
2018-04-20	2045	1259	246.4892
2018-04-20	2049	1250	246.5121

6 rows

I will clean up the column names again:

Hide

```
colnames(final.df) <- c("AnalysisDate", "NurseryId", "AppleID", "HarvestDate")
```

Perfect! In order to make the `AnalysisDate` and `HarvestDate` to be compatible, I will need to convert the `AnalysisDate` to julian days. For that I've written up the following function:

Hide

```
#' FUNCTION: is_leap_year
#' DESCRIPTION: Returns TRUE if a given year is leap year, FALSE otherwise
#' INPUT: year - a YYYY formatted year
#' OUTPUT: TRUE if year is leap year, FALSE otherwise
is_leap_year <- function(year) {
  if (is.na(str_extract(year, regex('[0-9]{4}')))) {
    stop(paste0("ERROR: ", year, " is not a properly formatted year. The format should b
e YYYY"))
  } else {
    year_n <- as.numeric(year)
    if ((year_n %% 4 == 0) &
      (year_n %% 100 == 0) &
      (year_n %% 400 == 0)) {
      return(TRUE)
    } else return(FALSE)
  }
}

#' FUNCTION: convert_julian
#' DESCRIPTION: Converts a YYYY-MM-DD formatted date into julian days
#' INPUT: date - a YYYY-MM-DD formatted date
#' OUTPUT: a number representing the julian days of date
convert_julian <- function(date) {
  date_regex <- '[0-9]{4}-[0-9]{2}-[0-9]{2}'
  if (is.na(str_extract(date, regex(date_regex)))) { # check to see if date is valid
    stop(paste0("ERROR: ", date, " is not a valid date! The format should be YYYY-MM-DD"
))
  } else {
    date_split <- str_split(date, "-")[[1]]
    year <- as.numeric(date_split[1])
    month <- as.numeric(date_split[2])
    day <- as.numeric(date_split[3])

    # calculate days in a year
    if (is_leap_year(year)) {
      days_in_year <- c(31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31)
    } else {
      days_in_year <- c(31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31)
    }
  }
  julian_value <- sum(days_in_year[1:(month - 1)]) + day
  return(julian_value)
}
```

I am going to be using the above function to convert the `AnalysisDate` to julian days:

Hide

```
analysis_dates <- as.character(as.POSIXct(final.df$AnalysisDate, format = "%Y-%m-%d"))
analysis_julian_days <- NULL
for (i in seq_along(analysis_dates)) {
  date <- analysis_dates[i]
  analysis_julian_days[i] <- convert_julian(date)
}
final.df$AnalysisDateJulian <- analysis_julian_days

head(final.df)
```

AnalysisDate <S3: POSIXct>	NurseryId <chr>	AppleID <chr>	HarvestDate <dbl>	AnalysisDateJulian <dbl>
2018-04-19	1009	3	271.7881	109
2018-04-20	2023	108	246.5290	110
2018-04-20	2029	110	225.7553	110
2018-04-20	2041	1268	259.3719	110
2018-04-20	2045	1259	246.4892	110
2018-04-20	2049	1250	246.5121	110

6 rows

Great, now we have the analysis date in julian format. We are ready to do correlation. But, before we do that, I want to explore the data a little.

Data Investigation

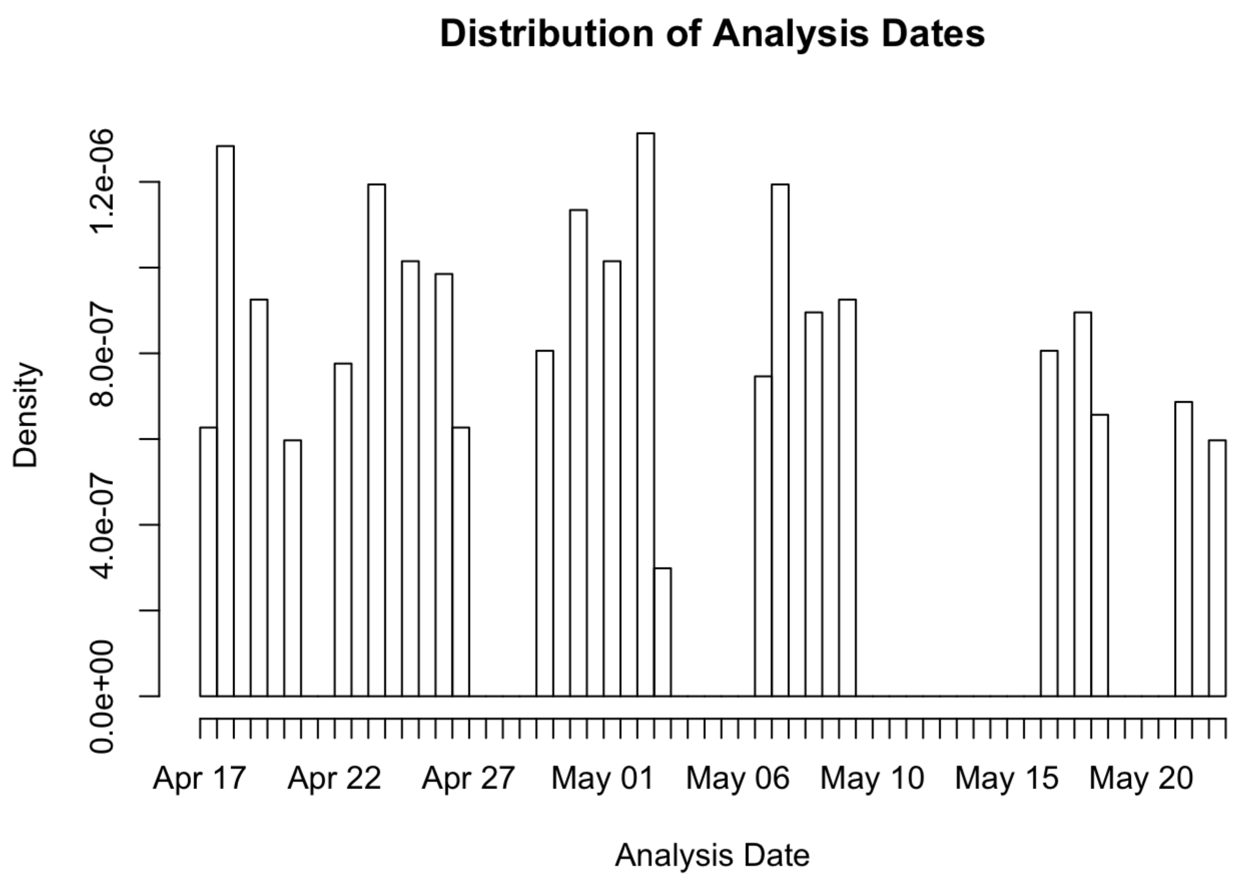
What is the distribution of the dates?

It looks like the analysis seems to have spanned from mid-April to end-May. Most of the analyses were done in the month of April.

Hide

```
hist(date_data$Date, breaks = 100, main = "Distribution of Analysis Dates", xlab = "Anal
ysis Date")
```

```
Warning in breaks[-1L] + breaks[-nB]: NAs produced by integer overflow
```



Were th analyses done on the same day?

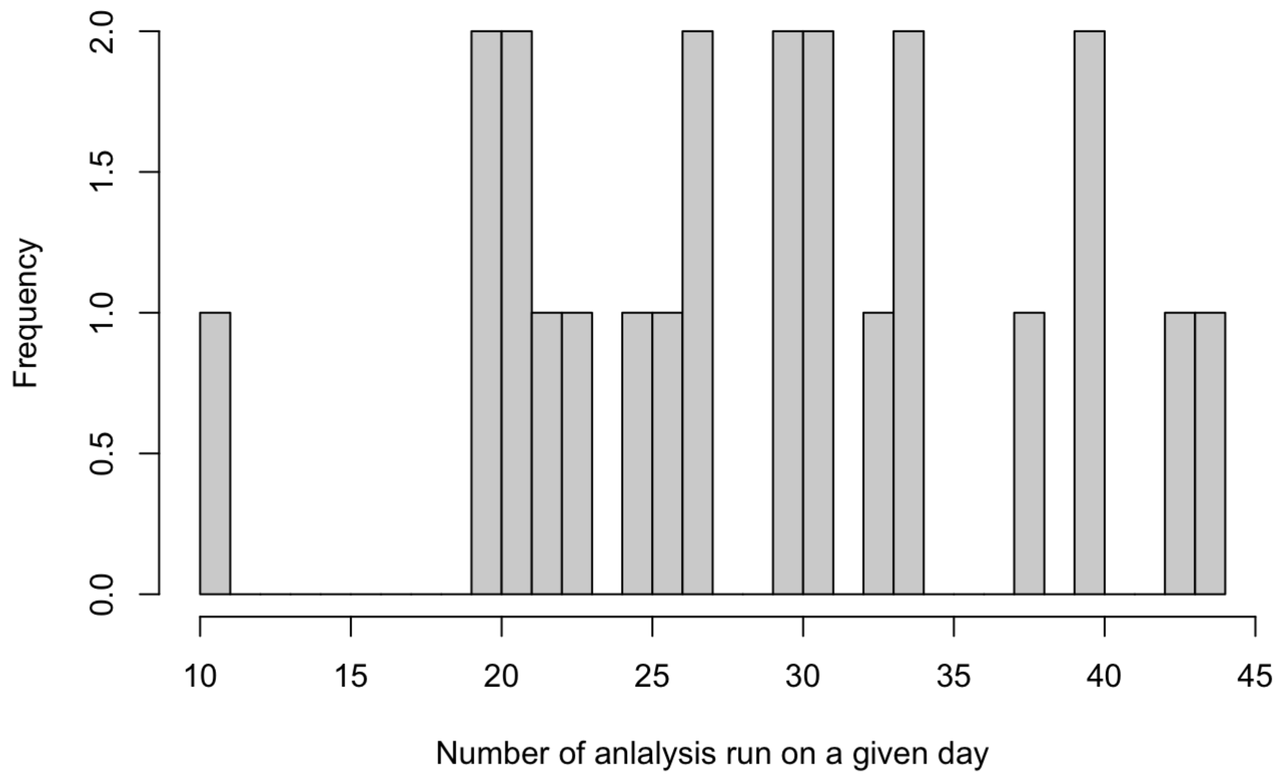
Hide

```
date_tbl <- sort(table(date_data$Date))
date_tbl
```

```


2018-05-04 2018-04-21 2018-05-23 2018-04-18 2018-04-27 2018-05-19 2018-05-22 
        10         20         20         21         21         22         23 
2018-05-07 2018-04-23 2018-04-30 2018-05-17 2018-05-09 2018-05-18 2018-04-20 
        25         26         27         27         30         30         31 
2018-05-10 2018-04-26 2018-04-25 2018-05-02 2018-05-01 2018-04-24 2018-05-08 
        31         33         34         34         38         40         40 
2018-04-19 2018-05-03 
        43         44 
```

Hide

```
hist(date_tbl, breaks = 25, main = "", xlab = "Number of anlalysis run on a given day")
```



There definitely were analysis that were run on the same day. 2018-05-03 was the day on which there were the highest number (44) of analyses run.

Harvest Date & Analysis Date Correlation

Uhm… there seems to be a positive correlation between the harvest dat and analysis date. This correlation also seems to be statistically significant. I am going to plot out the values on X-Y plot line of best fit

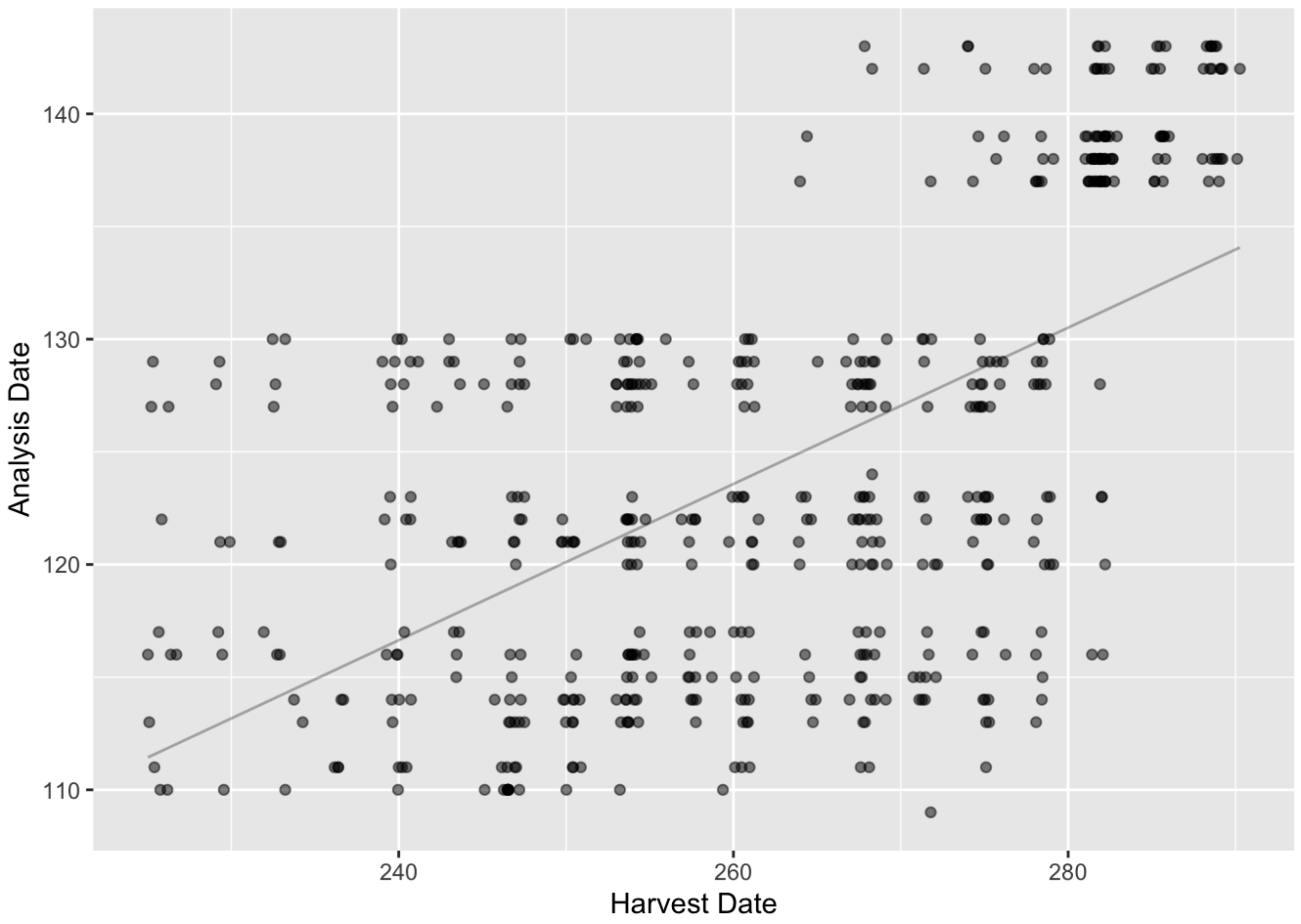
Hide

```
analysis_date <- final.df$AnalysisDateJulian
harvest_date <- final.df$HarvestDate
fit.hv_dt_analysis <- lm(analysis_date ~ harvest_date)
summary(fit.hv_dt_analysis)
```

```

Call:
lm(formula = analysis_date ~ harvest_date)

Residuals:
     Min       1Q   Median       3Q      Max 
-18.6620  -6.4298  -0.0265   6.5704  17.4568 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept)  33.39892    5.54942   6.018 3.33e-09 ***
harvest_date  0.34683    0.02109  16.445  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.851 on 519 degrees of freedom
Multiple R-squared:  0.3426,	Adjusted R-squared:  0.3413 
F-statistic: 270.4 on 1 and 519 DF,  p-value: < 2.2e-16
```

Hide

```
ggplot(data.frame(analysis_date, harvest_date), aes(x = harvest_date, y = analysis_dat
e)) +
  geom_point(alpha = 0.5, color = "black") +
  geom_line(aes(x = harvest_date, y = fit.hv_dt_analysis$fitted.values),  # predicted da
ta
            color = 'black', alpha = 0.3) +
  xlab("Harvest Date") +
  ylab("Analysis Date")
```



Looking at this graph, it is a little re-assuring that the that it is not a perfect correlation between the harvest date and analysis date. There seems to be sort of an outlier for late harvested apples. It almost feels like some late harvested apples were run at the very last.

The next thing I am going to do is assess the correlation of compound abundance and harvest date with analysis date as co-factor in order to see if the correlation still exists.

Hide

```
gcms_pheno_tbl <- read_excel('data/processed/sup_tbl_1-final_gcms_phenotype_table_v2.xls
x')
dim(gcms_pheno_tbl)
```

```
[1] 515 107
```

Hide

```
gcms_pheno_tbl$STVC <- rowSums(gcms_pheno_tbl[, 2:ncol(gcms_pheno_tbl)])

# only keep relevant columns
gcms_pheno_tbl <- gcms_pheno_tbl[, c("appleid", "STVC")]

# convert appleid column to character to be compatible with final.df
gcms_pheno_tbl$appleid <- as.character(gcms_pheno_tbl$appleid)

# looking at the data
head(gcms_pheno_tbl)
```

appleid	**STVC**
<chr>	<dbl>
108	128.55707
110	167.08762
1268	66.42582
1259	209.51587
1250	207.99805
112	115.03068

6 rows

We will now join this GC-MS STVC data with the `final.df` in order to perform correlation between abundance and harvest date.

Hide

```
nrow(final.df)
```

```
[1] 521
```

Hide

```
colnames(gcms_pheno_tbl)
```

```
[1] "appleid" "STVC"
```

```
appleid

STVC
```

Hide

```
final.df <- inner_join(final.df, gcms_pheno_tbl, by = c("AppleID" = "appleid"))
nrow(final.df)
```

```
[1] 521
```

Hide

```
head(final.df)
```

AnalysisDate <S3: POSIXct>	NurseryId <chr>	AppleID <chr>	HarvestDate <dbl>	AnalysisDateJulian <dbl>	STVC <dbl>
2018-04-19	1009	3	271.7881	109	149.94119
2018-04-20	2023	108	246.5290	110	128.55707
2018-04-20	2029	110	225.7553	110	167.08762
2018-04-20	2041	1268	259.3719	110	66.42582
2018-04-20	2045	1259	246.4892	110	209.51587
2018-04-20	2049	1250	246.5121	110	207.99805

6 rows

Hide

```
stvc <- final.df$STVC
fit.hv_dt_w_cofactor <- lm(stvc ~ harvest_date + analysis_date)
summary(fit.hv_dt_w_cofactor)
```

```

Call:
lm(formula = stvc ~ harvest_date + analysis_date)

Residuals:
    Min      1Q  Median      3Q     Max 
-219.37 -104.87  -38.42   85.40  626.79 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)   722.3417    99.4338   7.265 1.38e-12 ***
harvest_date   -1.8377     0.4506  -4.078 5.25e-05 ***
analysis_date  -0.1908     0.7604  -0.251    0.802    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 136 on 518 degrees of freedom
Multiple R-squared:  0.04992,	Adjusted R-squared:  0.04625 
F-statistic: 13.61 on 2 and 518 DF,  p-value: 1.736e-06
```

The R^2 value seems to have significantly decreased, meaning that the correlation gets weaker as we account for the analysis date, which is reassuring. However, the p-value still seems significant.