Modeling Problem I

Karol Orozco & Charles Hanks

Predicting Province

```
knitr::opts_chunk$set(echo = TRUE, message = FALSE, warning = FALSE)

library(tidyverse)
library(formatR)
library(moderndive)
library(skimr)

wine_pinot <- readRDS(gzcon(url(
  "https://github.com/karolo89/machine_learning_assignment/raw/main/pinot.rds")))
```

```
#adding log price column
pinot <- wine_pinot %>%
  mutate(lprice = log(price))

pinot <- pinot %>%
  mutate(id = as.factor(id))%>%
  mutate(year = as.factor(year))

summary(pinot)
```

```
      id          province             price             points     
1      :   1   Length:8380        Min.   :   7.00   Min.   :80.00  
2      :   1   Class :character   1st Qu.:  31.00   1st Qu.:88.00  
3      :   1   Mode  :character   Median :  45.00   Median :90.00  
4      :   1                      Mean   :  52.52   Mean   :89.98  
5      :   1                      3rd Qu.:  60.00   3rd Qu.:92.00  
6      :   1                      Max.   :2500.00   Max.   :98.00  
(Other):8374                                                       
```

```
     year       description            lprice     
 2014   :2046   Length:8380        Min.   :1.946  
 2013   :1819   Class :character   1st Qu.:3.434  
 2012   :1505   Mode  :character   Median :3.807  
 2015   : 815                      Mean   :3.779  
 2011   : 582                      3rd Qu.:4.094  
 2010   : 502                      Max.   :7.824  
 (Other):1111                                     
```

Preliminary EDA, Feature Engineering Brainstorm, Initial Thoughts

```
pinot %>%
  group_by(province) %>%
  summarize(prov_freq = n(),
            percent_of_ds = round(prov_freq/8380,2))
```

```
# A tibble: 6 x 3
  province          prov_freq percent_of_ds
  <chr>                 <int>         <dbl>
1 Burgundy               1193          0.14
2 California             3959          0.47
3 Casablanca_Valley       131          0.02
4 Marlborough             229          0.03
5 New_York                131          0.02
6 Oregon                 2737          0.33
```

```
#nearly half of wines are californian, good to know...

pinot %>%
  filter(str_detect(description, "[Oo]ak")) %>%
  nrow()
```

```
[1] 1301
```

```
#1301/8380 have the work oak in description

pinot %>% filter(str_detect(description, "[Oo]ak")) %>%
  group_by(province) %>% summarize(prov_freq = n(),
                                    oak_perc = round(prov_freq/1301,2))
```

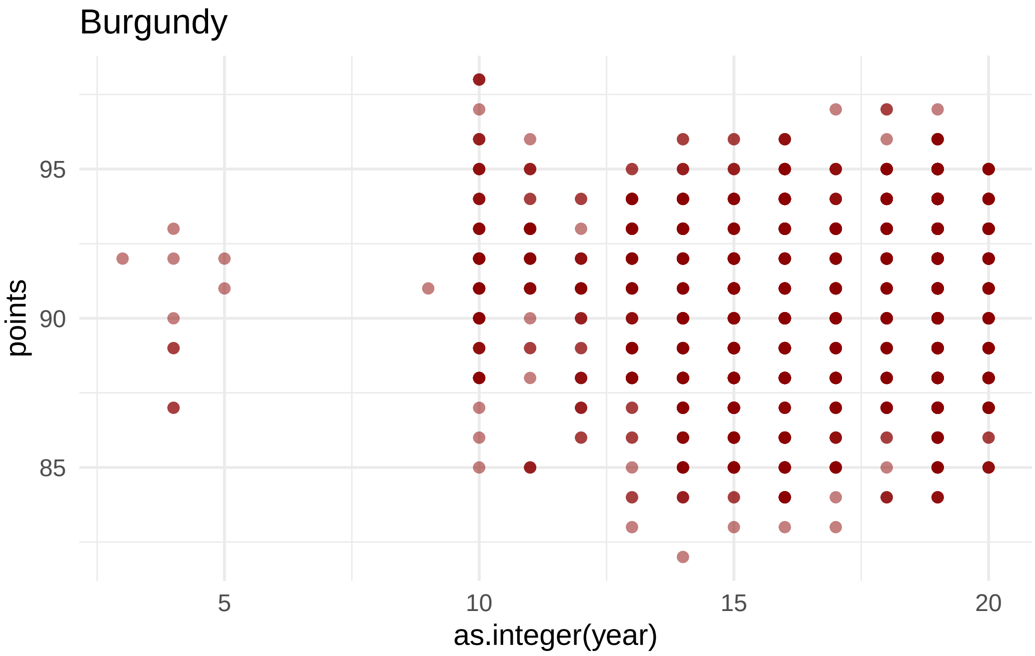
```
# A tibble: 6 x 3
  province          prov_freq oak_perc
  <chr>                 <int>    <dbl>
1 Burgundy                  8     0.01
2 California              739     0.57
3 Casablanca_Valley        64     0.05
4 Marlborough              32     0.02
5 New_York                  9     0.01
6 Oregon                  449     0.35
```

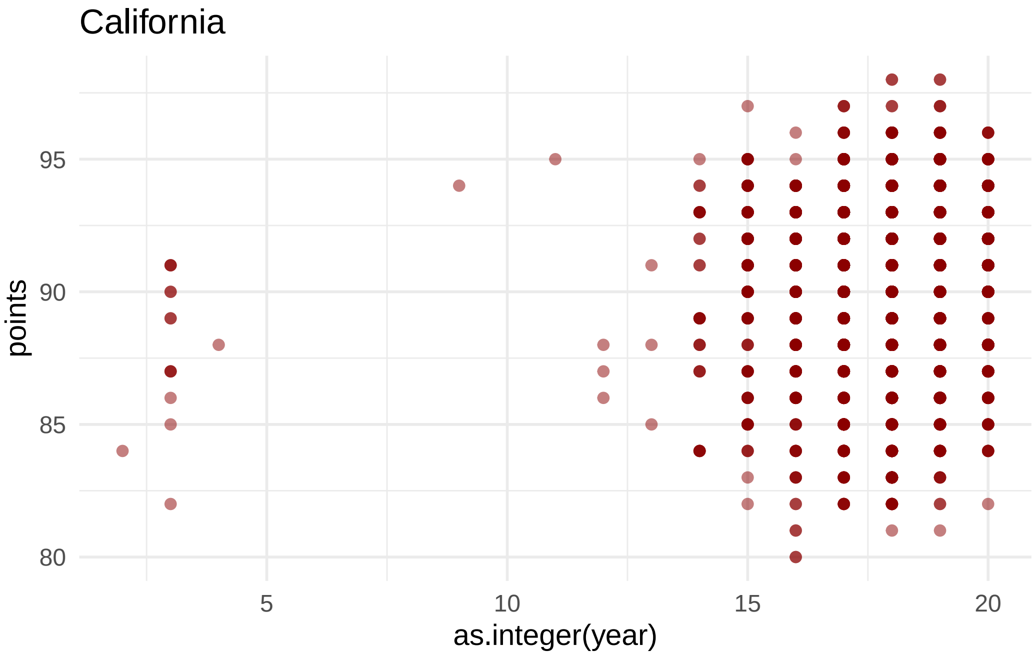
```
#it is likely California or Oregon if there is oak in the description

#some french language patterns to think about developing a regex from:
# "_de_" / "d'"
# "name-name"
# accented letters: "é","ô",
# "St."

pinot %>%
  group_by(province) %>%
  summarize(avgPrice = mean(price),
            avgPoints = mean(points))
```

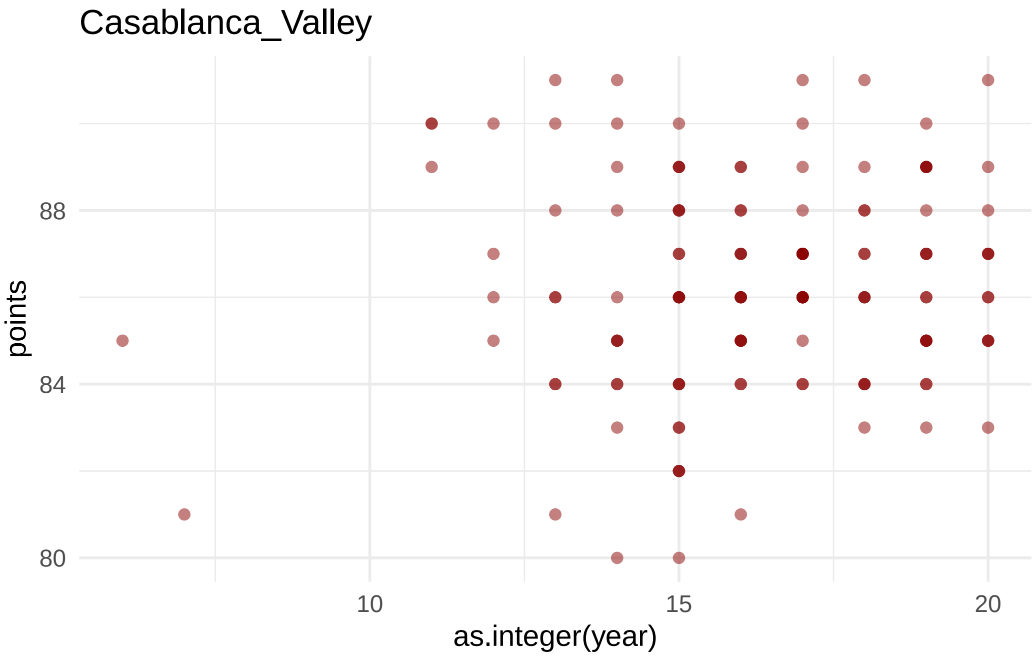
```
# A tibble: 6 x 3
  province          avgPrice avgPoints
  <chr>                <dbl>     <dbl>
1 Burgundy              98.0      90.4
2 California            47.5      90.5
3 Casablanca_Valley     21.1      86.3
4 Marlborough           27.7      87.6
5 New_York              25.7      87.7
6 Oregon                44.9      89.5
```

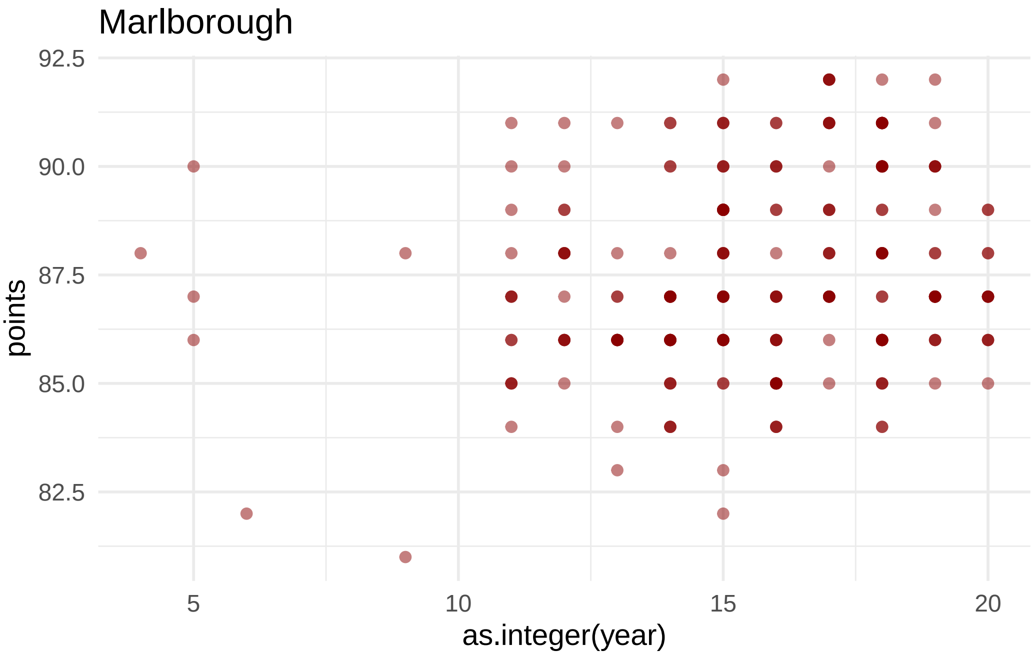
```
# Burgundy wines are on average significantly more expensive...
# and casablanca valley wines on average have the lowest price and score.

#which wines do people recommend waiting before drinking? i.e "drink from XXXX"

#some words to check out: "edge","tannins","dense","firm", oregon pinot is fruity.
```

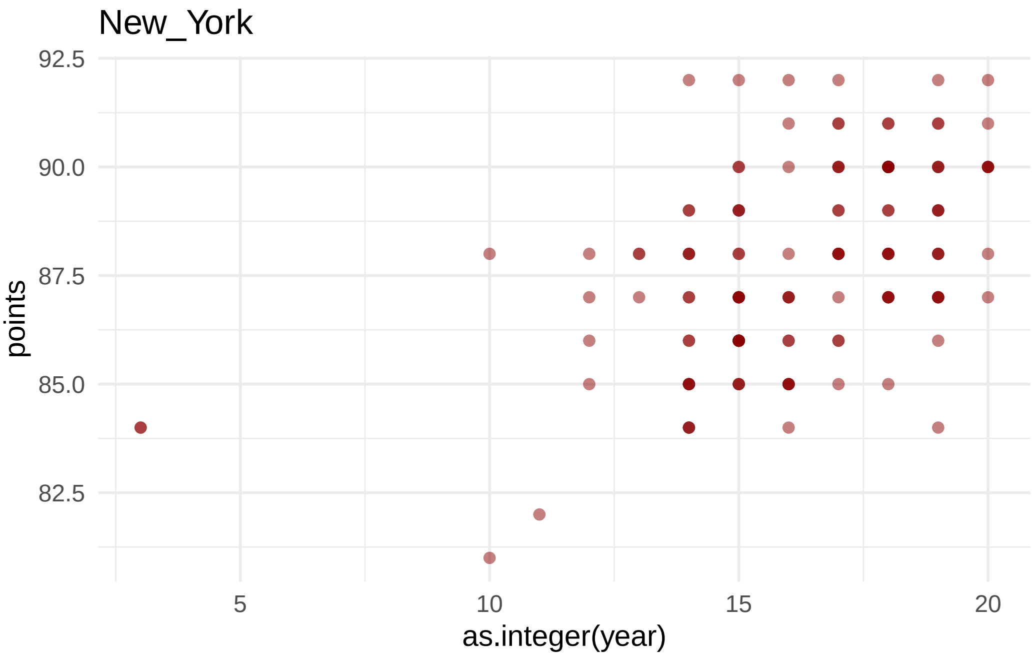
```
province_vec = c("Burgundy", "California", "Casablanca_Valley","Marlborough",
                 "New_York", "Oregon")

for(i in province_vec){
  plot = ggplot(pinot %>%
                  filter(province == i), aes(x = as.integer(year), y = points)) +
    geom_point(alpha =.5, color = "red4") +
    ggtitle(i)+
    theme_minimal()

  print(plot)
}
```

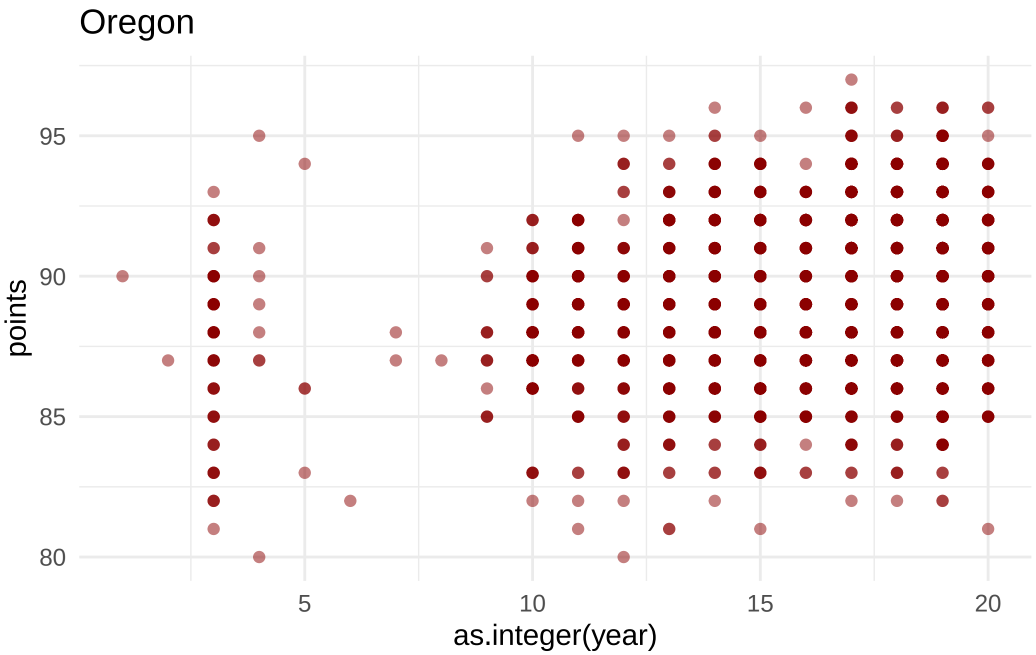



California
points
95
90
85
80
5
10
15
20
as.integer(year)

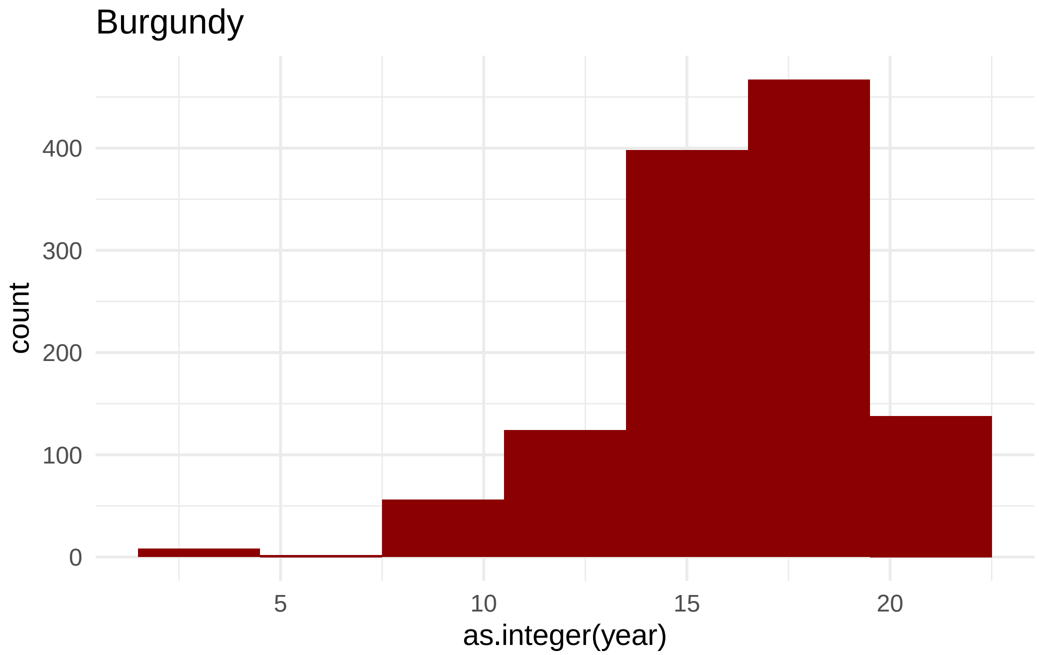

Casablanca_Valley
points
88
84
80
10
15
20
as.integer(year)

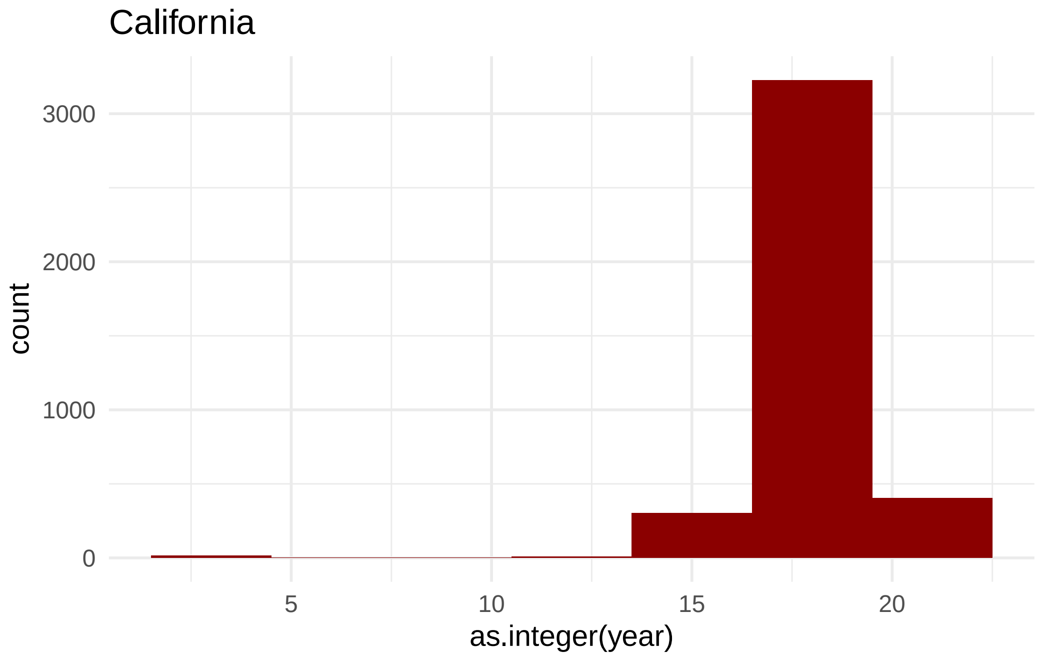

Marlborough
92.5
90.0
87.5
85.0
82.5
points
5
10
15
20
as.integer(year)

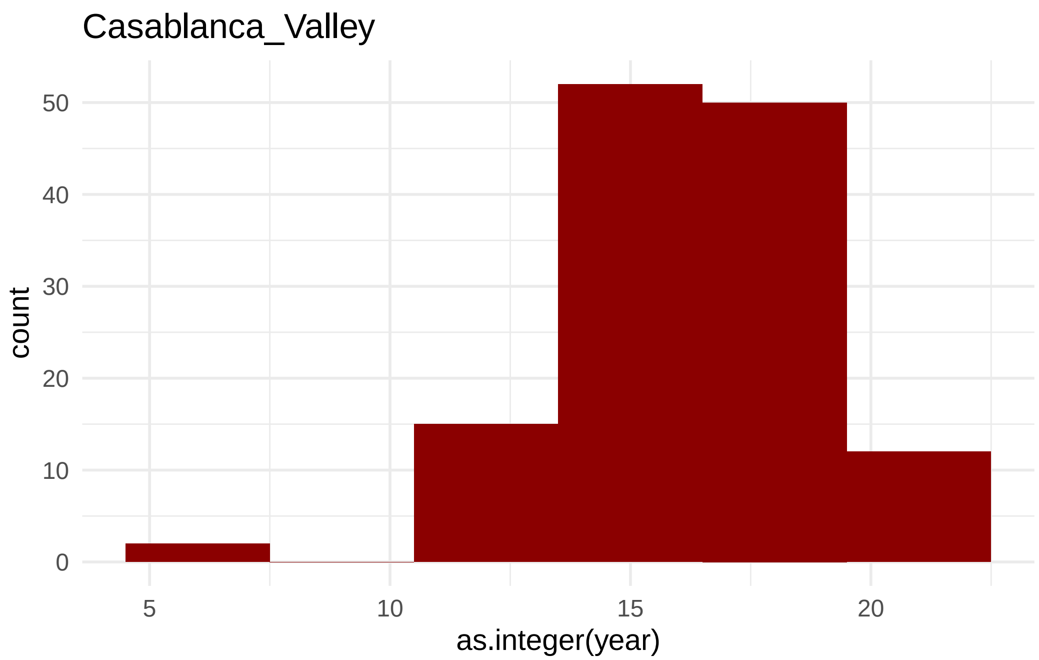

New_York
92.5
90.0
87.5
85.0
82.5
points
5
10
15
20
as.integer(year)

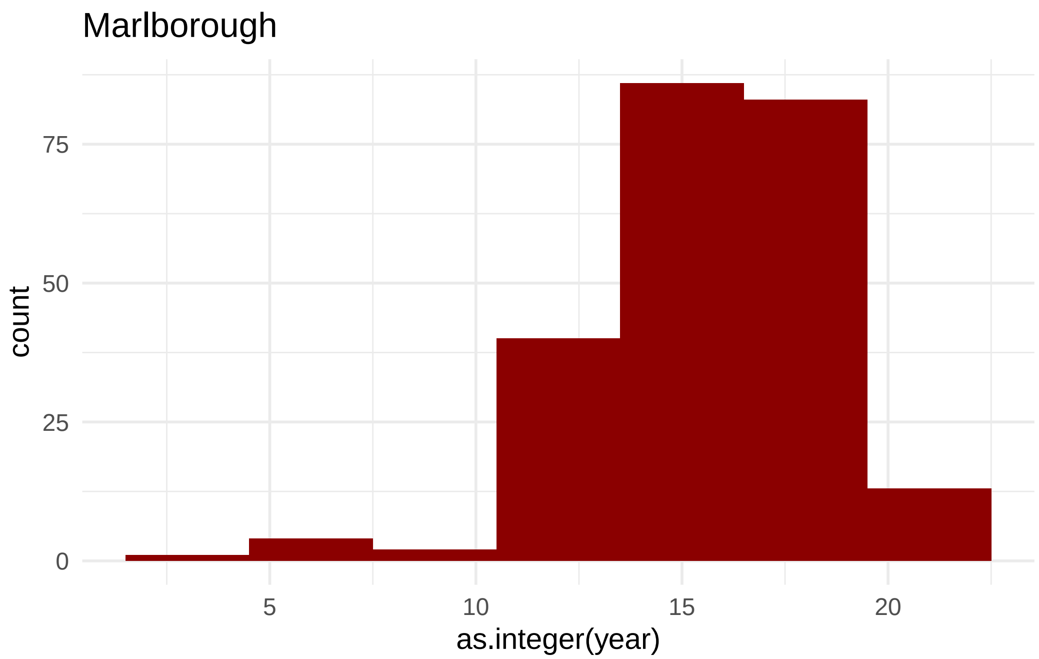


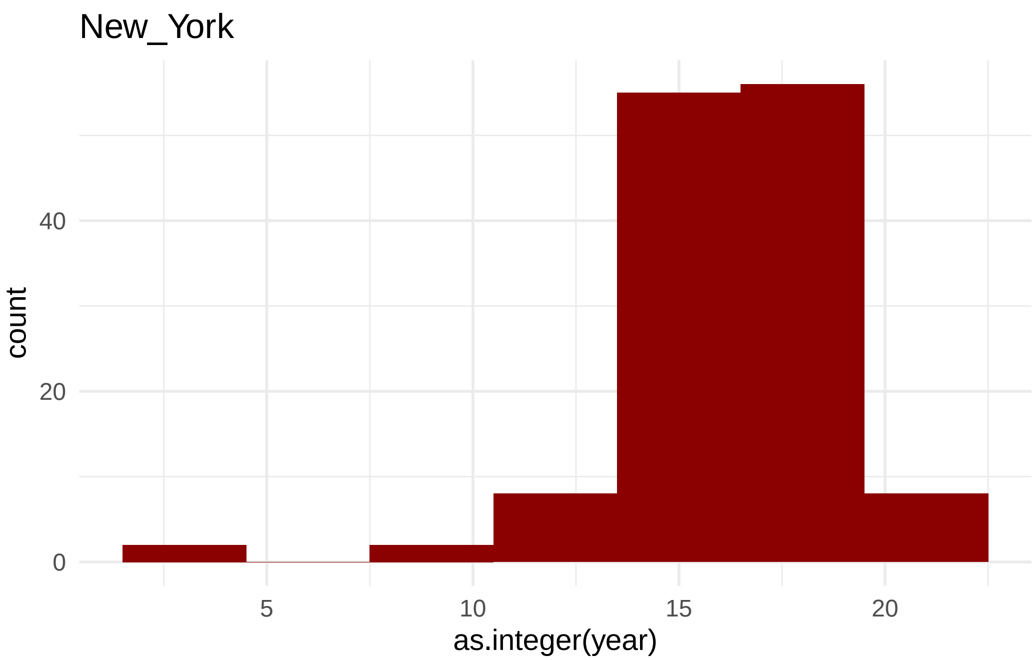
```
for(i in province_vec){
  plot2 = ggplot(pinot %>%
                   filter(province == i), aes(x = as.integer(year))) +
    geom_histogram(binwidth =3, fill = "red4") +
    ggtitle(i)+
    theme_minimal()
  print(plot2)
}
```

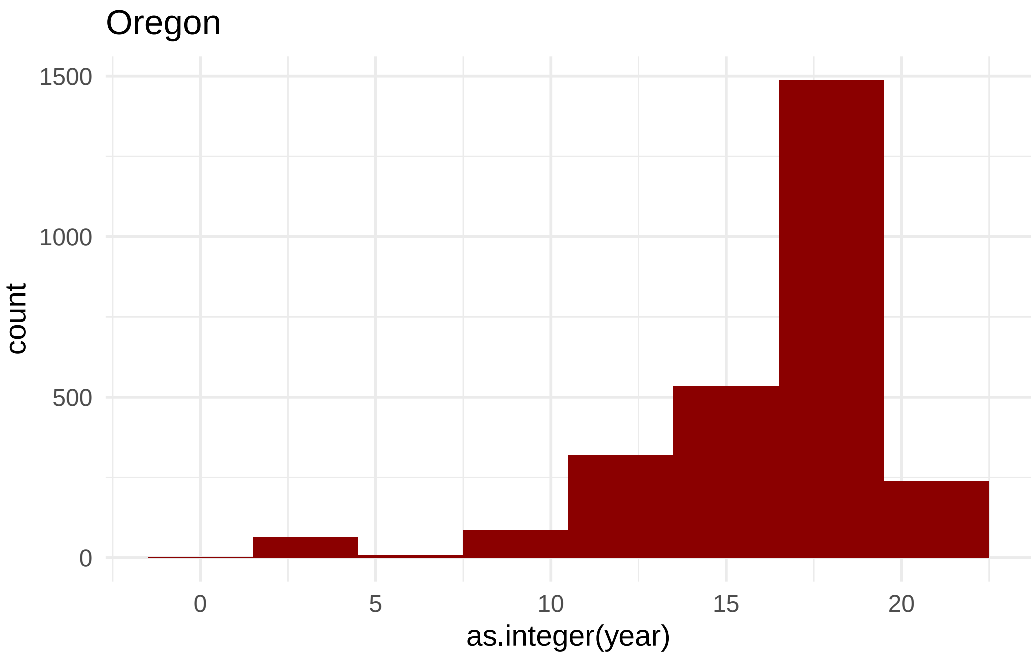

Burgundy
count
400
300
200
100
0
5
10
15
20
as.integer(year)


California
count
3000
2000
1000
0
5
10
15
20
as.integer(year)


Casablanca_Valley
count
0
10
20
30
40
50
5
10
15
20
as.integer(year)


Marlborough
count
0
25
50
75
5
10
15
20
as.integer(year)


New_York
count
0
20
40
5
10
15
20
as.integer(year)


Oregon
count
0
500
1000
1500
0
5
10
15
20
as.integer(year)

```
#Some findings from viz:
#california pinot noir production did not begin until ~2008, then exploded!
#before year 2000, likely to be oregon
#burgundy pinots score high around 2005,
#after almost no burgundy pinots between 2000 and 2005
#California pinot game WAY STRONG between 2010 and 2015
#New York pinot score high between 2008 and 2015
#What happened around 2014?? Counts drop across provinces....
```

Preprocessing (3pts)

1. Preprocess the dataframe that you created in the previous question using centering and scaling of the numeric features
2. Create dummy variables for the year factor column

Running KNN (5pts)

1. Split your data into an 80/20 training and test set
2. Use Caret to run a KNN model that uses your engineered features to predict province

- use 5-fold cross validated subsampling
- allow Caret to try 15 different values for K

3. Display the confusion matrix on the test data

Kappa (2pts)

Is this a good value of Kappa? Why or why not?

Answer: (write your answer here)

Improvement (2pts)

Looking at the confusion matrix, where do you see room for improvement in your predictions?

Answer: (write your answer here)

Group Activity: Naive Bayes Model

Use the top words by province to…

1. Engineer more features that capture the essence of Casablanca, Marlborough and New York

2. Look for difference between California and Oregon

3. Use what you find to run naive Bayes models that achieve a Kappa that approaches 0.5

```r
library(tidytext)
library(caret)
wine = wine_pinot
names(wine)[names(wine) == 'id'] = 'ID'
```

Document term matrix:

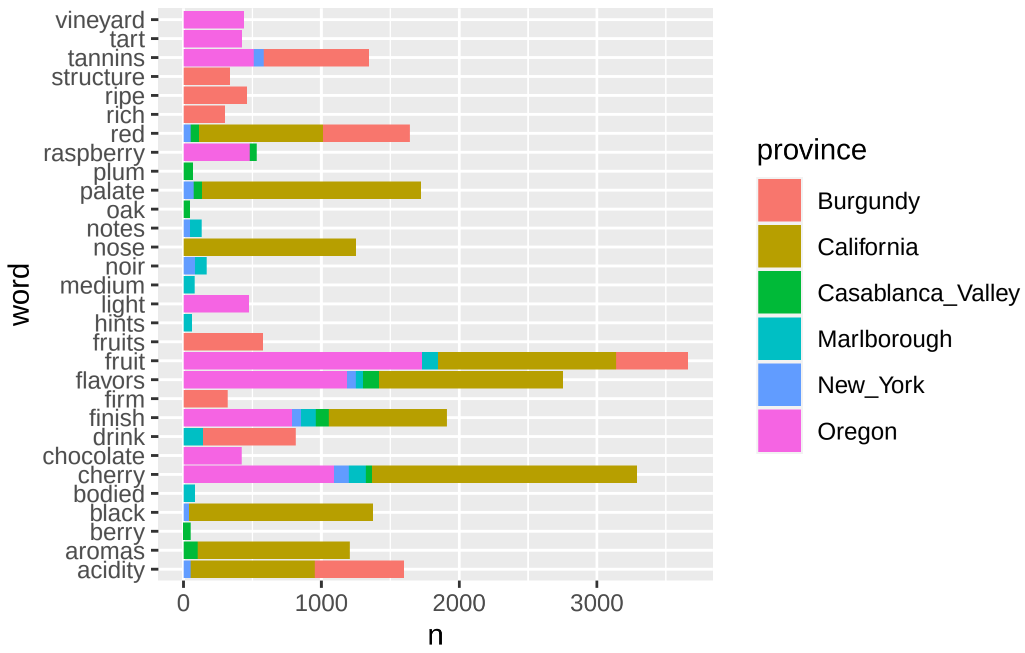
```r
df <- wine %>%
  unnest_tokens(word, description) %>%
  anti_join(stop_words) %>% # get rid of stop words
  filter(word != "wine") %>%
  filter(word != "pinot") %>%
  count(ID, word) %>%
  group_by(ID) %>%
  mutate(freq = n/sum(n)) %>%
  mutate(exists = (n>0)) %>%
  ungroup %>%
  group_by(word) %>%
  mutate(total = sum(n))
```

Pivot wide and rejoin with wine:

```r
wino <- df %>%
  filter(total > 900) %>%
  pivot_wider(id_cols = ID, names_from = word, values_from = exists, values_fill = list(ex
  merge(select(wine,ID, province), all.y=TRUE) #%>%
  #drop_na()

#wino <- merge(select(wine,ID, province), wino, by="ID", all.x=TRUE) %>%
#  arrange(ID)
#View(wino)
wino <- replace(wino, is.na(wino), FALSE)
```

Visualizing distribution to select distinct features for provinces:

```
df %>%
  left_join(select(wine, ID, province), by = "ID") %>%
  count(province, word) %>%
  group_by(province) %>%
  top_n(10,n) %>%
  arrange(province, desc(n)) %>%
  ggplot(aes(x = word, y = n, fill = province)) + geom_col() + coord_flip()
```



```
wino = wino %>% select(ID, province, tart, plum, oak, bodied,black,nose,palate,ripe,cherry
```

train & test model:

```
wine_index <- createDataPartition(wino$province, p = 0.80, list = FALSE)
train <- wino[ wine_index, ]
test <- wino[-wine_index, ]

fit <- train(province ~ .,
             data = train,
             method = "naive_bayes",
```

```r
             tuneGrid = expand.grid(usekernel = c(T,F), laplace = T, adjust = T),
             metric = "Kappa",
             trControl = trainControl(method = "cv"))


confusionMatrix(predict(fit, test),factor(test$province))
```

```
Confusion Matrix and Statistics

                   Reference
Prediction           Burgundy California Casablanca_Valley Marlborough New_York
  Burgundy                226        156                 5          17        8
  California                3        464                16           4       11
  Casablanca_Valley         1         21                 1           1        1
  Marlborough               3         17                 0          10        0
  New_York                  0         17                 0           3        4
  Oregon                    5        116                 4          10        2
                   Reference
Prediction           Oregon
  Burgundy              274
  California             80
  Casablanca_Valley      10
  Marlborough             5
  New_York                1
  Oregon                177

Overall Statistics

               Accuracy : 0.5272
                 95% CI : (0.5029, 0.5514)
    No Information Rate : 0.4728
    P-Value [Acc > NIR] : 4.765e-06

                  Kappa : 0.3395

 Mcnemar's Test P-Value : < 2.2e-16

Statistics by Class:

                     Class: Burgundy Class: California Class: Casablanca_Valley
Sensitivity                   0.9496            0.5866                0.0384615
Specificity                   0.6794            0.8707                0.9793564
```

```
Pos Pred Value                0.3294            0.8028                0.0285714
Neg Pred Value                0.9878            0.7014                0.9847375
Prevalence                    0.1423            0.4728                0.0155409
Detection Rate                0.1351            0.2773                0.0005977
Detection Prevalence          0.4100            0.3455                0.0209205
Balanced Accuracy             0.8145            0.7287                0.5089090
                     Class: Marlborough Class: New_York Class: Oregon
Sensitivity                    0.222222        0.153846        0.3236
Specificity                    0.984644        0.987250        0.8783
Pos Pred Value                 0.285714        0.160000        0.5637
Neg Pred Value                 0.978632        0.986650        0.7277
Prevalence                     0.026898        0.015541        0.3270
Detection Rate                 0.005977        0.002391        0.1058
Detection Prevalence           0.020921        0.014943        0.1877
Balanced Accuracy              0.603433        0.570548        0.6010
```

Creating more features

```
features = wine %>%
  mutate(aging = str_detect(description,"aging"),
         chocolate =  str_detect(description, "chocolate"),
         vineyard = str_detect(description, "vineyard")) %>%
            select(ID,aging,chocolate,vineyard)

wino2 = wino %>%
  left_join(features, by = "ID")
```

Test 2

```
wine_index <- createDataPartition(wino2$province, p = 0.80, list = FALSE)
train <- wino2[ wine_index, ]
test <- wino2[-wine_index, ]

fit <- train(province ~ .,
             data = train,
             method = "naive_bayes",
             tuneGrid = expand.grid(usekernel = c(T,F), laplace = T, adjust = T),
             metric = "Kappa",
             trControl = trainControl(method = "cv"))
fit
```

```
Naive Bayes
```

```
6707 samples
  15 predictor
   6 classes: 'Burgundy', 'California', 'Casablanca_Valley', 'Marlborough', 'New_York', 'Oreg

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 6037, 6036, 6036, 6036, 6038, 6037, ...
Resampling results across tuning parameters:

  usekernel  Accuracy   Kappa
  FALSE      0.2655545  0.1783915
   TRUE      0.5941670  0.3874900

Tuning parameter 'laplace' was held constant at a value of TRUE

Tuning parameter 'adjust' was held constant at a value of TRUE
Kappa was used to select the optimal model using the largest value.
The final values used for the model were laplace = TRUE, usekernel = TRUE
 and adjust = TRUE.
```

```
confusionMatrix(predict(fit, test),factor(test$province))
```

```
Confusion Matrix and Statistics

                   Reference
Prediction          Burgundy California Casablanca_Valley Marlborough New_York
  Burgundy               233        164                 4          24       10
  California               5        604                22          20       16
  Casablanca_Valley        0          0                 0           0        0
  Marlborough              0          0                 0           0        0
  New_York                 0          0                 0           0        0
  Oregon                   0         23                 0           1        0
                   Reference
Prediction          Oregon
  Burgundy             224
  California           180
  Casablanca_Valley      0
  Marlborough            0
  New_York               0
  Oregon               143
```

```
Overall Statistics

               Accuracy : 0.5858
                 95% CI : (0.5617, 0.6095)
    No Information Rate : 0.4728
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3836

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: Burgundy Class: California Class: Casablanca_Valley
Sensitivity                   0.9790            0.7636                  0.00000
Specificity                   0.7031            0.7245                  1.00000
Pos Pred Value                0.3536            0.7131                      NaN
Neg Pred Value                0.9951            0.7736                  0.98446
Prevalence                    0.1423            0.4728                  0.01554
Detection Rate                0.1393            0.3610                  0.00000
Detection Prevalence          0.3939            0.5063                  0.00000
Balanced Accuracy             0.8411            0.7440                  0.50000
                     Class: Marlborough Class: New_York Class: Oregon
Sensitivity                      0.0000         0.00000       0.26143
Specificity                      1.0000         1.00000       0.97869
Pos Pred Value                      NaN             NaN       0.85629
Neg Pred Value                   0.9731         0.98446       0.73174
Prevalence                       0.0269         0.01554       0.32696
Detection Rate                   0.0000         0.00000       0.08548
Detection Prevalence             0.0000         0.00000       0.09982
Balanced Accuracy                0.5000         0.50000       0.62006
```

```
#Higher kappa value, but now model is not predicting any of the sparse provinces
```