Crop Quality Prediction Using Crop Price

Angelica Kim

```r
#load the data using your read-in command
data <- read_csv("./data.csv")
```

```
## Rows: 54057 Columns: 34
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (34): date, crop, daily_avg_price, daily_total_volume, avg_price_low, av...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

Introduction

Most agricultural crops in Korea are traded in the public wholesale produce auctions governed by local governments. When farmers supply their produce to the wholesale market, they report the quality based on their own evaluation. However, this voluntary quality grading system in Korea thus raises doubts about its objectivity because the standards are subjective and arbitrary, which may lead to the overvaluation or undervaluation of the crops. It may also cause high price volatility because the same crops can be evaluated very differently according to subjective standards. Therefore, establishing an objective quality grading system would be important for transforming the agricultural supply chain and market to be fairer, more competitive, and more resilient. In order to provide data-driven insights into how to identify crop quality in a reliable manner, this project examines what factors can potentially represent the quality. In the absence of data on quality grades, price is used as an indicator of quality because the auction price is determined by the experts at the produce auctions who inspect the crops based on the freshness, color, shape, and ripeness, not just the arbitrary grades self-reported by the producers. An objective quality rating system would not only improve the livelihood of smallholder farmers who had been unable to sell their produce at full prices, but also upscale the overall quality of the crops distributed in the market because farmers would have to grow high quality products in order to get a higher price value for them according to the new objective standard.

Choi et al. redefines the quality grades based on the price and transaction volume (Choi et al, 2021). However, they apply ANOVA, not classification methods, to differentiate the crops in four levels. In this project, I analyze a time series data on agricultural crops that contains information on wholesale and retail market transactions, weather, and trades (import and export). I use the price range as a proxy for crop quality and apply classification techniques to see if the data is capable of classifying the quality. Under the assumption that some information in the data, such as meteorological variables, has an underlying relationship with crop quality, I also conduct factor analysis to discover any latent factors in the hopes that they may be reflective of the quality.

Preliminary Analysis

In this project, I analyze aggregate time series data that combines daily transaction information from wholesale and retail markets, daily weather in the primary growing areas, and monthly trades data for 37 crops from 2013 to 2016. The data was provided by an online artificial intelligence (AI) competition platform hosted by Korea Agro-Fisheries & Food Trade Corporation (Link to data source). Note that I translated the variable names into English since the original data was from a source based in Korea.

Data Description:

1. `date`: Date
2. `crop`: Type of crops encoded in numbers (0-36)

Price information (All price units are in Korean Won):

3. `daily_avg_price`: Daily average price of each crop across 32 wholesale markets in Korea
4. `daily_total_volume`: Total volume of each crop traded in 32 wholesale markets each day
5. `avg_price_low`: Daily average of prices that fall below the value of `daily_avg_price` on that day
6. avg_price_high": Daily average of prices that are higher than the value of `daily_avg_price` on that day
   7."`low_price_volume`: Total volume of each crop whose price is lower than the `daily_avg_price` for that crop traded on that day

7. `high_price_volume`: Total volume of each crop whose price is higher than the `daily_avg_price` for that crop traded on that day

Wholesale and retail transaction-related variables:

9. `daily_max_wholesale_price`: Daily maximum price of each crop traded in 12 wholesale markets
10. `daily_avg_wholesale_price`: Daily average price of each crop traded in 12 wholesale markets
11. `daily_min_wholesale_price`: Daily minimum price of each crop traded in 12 wholesale markets
12. `daily_max_retail_price`: Daily maximum price of each crop traded in 45 retail markets
13. `daily_avg_retail_price`: Daily average price of each crop traded in 45 retail markets
14. `daily_min_retail_price"`: Daily minimum price of each crop traded in 45 retail markets

Trade-related variables:

15. `export_weight`: Weight of the crop exported, by month (in kilograms)

16. `export_amount_usd`: Value of export, by month (in US Dollars)
17. `import_weight`: Weight of the crop imported, by month (in kilograms)

18. `import_amount_usd`: Value of import, by month (in US Dollars)
19. `trade_balance_usd`: The balance of trade, which is the difference between the value of exports and imports (in US Dollars)

Meteorological variables:

20. `area0_base_temp`: The base temperature of each crop at the first primary growing area, below which the crop no longer develops (in Celsius degrees)
21. `area0_max_temp`: Daily maximum temperature at the first primary growing area of each crop (in Celsius degrees)

22. `area0_min_temp`: Daily minimum temperature at the first primary growing area of each crop (in Celsius degrees)

23. `area0_avg_temp`: Daily average temperature at the first primary growing area of each crop (in Celsius degrees)

24. `area0_precip`: Daily precipitation amount at the first primary growing area of each crop (in milliliters) There are two more sets of the same variables for two other primary growing areas.

```
glimpse(data)
```

```
## Rows: 54,057
## Columns: 34
## $ date                        <dbl> 20130101, 20130102, 20130103, 20130104, 2013~
## $ crop                        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ daily_avg_price             <dbl> NA, 20712.24, 3401.86, 6553.90, 4299.08, 466~
## $ daily_total_volume          <dbl> NA, 22, 541, 279, 1233, 396, 324, NA, 649, 3~
## $ avg_price_low               <dbl> NA, 4605.775, 2003.085, 2266.215, 2882.420, ~
## $ avg_price_high              <dbl> NA, 40040.00, 7526.73, 8247.54, 9705.63, 115~
## $ low_price_volume            <dbl> NA, 12, 404, 79, 977, 291, 164, NA, 477, 184~
## $ high_price_volume           <dbl> NA, 10, 137, 200, 256, 105, 160, NA, 172, 17~
## $ daily_max_wholesale_price <dbl> NA, 180000, 180000, 180000, 180000, 180000, ~
## $ daily_avg_wholesale_price <dbl> NA, 178800, 178800, 178800, 178800, 177800, ~
## $ daily_min_wholesale_price <dbl> NA, 174000, 174000, 174000, 174000, 174000, ~
## $ daily_max_retail_price    <dbl> NA, 5980, 5980, 5980, 5980, 5980, NA, NA, 59~
## $ daily_avg_retail_price    <dbl> NA, 4298.52, 4298.52, 4298.52, 4298.52, 4286~
## $ daily_min_retail_price    <dbl> NA, 3210, 3210, 3210, 3210, 3210, NA, NA, 32~
## $ export_weight               <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ export_amount_usd           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ import_weight               <dbl> 1864728, 1864728, 1864728, 1864728, 1864728,~
## $ import_amount_usd           <dbl> 1195211, 1195211, 1195211, 1195211, 1195211,~
## $ trade_balance_usd           <dbl> -1195211, -1195211, -1195211, -1195211, -119~
## $ area0_base_temp             <dbl> -3.9, 2.8, -1.2, -1.7, -4.6, 0.6, -3.5, -0.5~
## $ area0_max_temp              <dbl> 4.4, 6.7, 5.0, 6.8, 6.2, 8.1, 8.9, 9.1, 4.1,~
## $ area0_min_temp              <dbl> -7.7, -1.1, -6.3, -4.5, -6.1, -3.3, -5.8, -1~
## $ area0_avg_temp              <dbl> -1.4, 3.5, -1.2, 0.9, -0.7, 2.2, -0.4, 4.2, ~
## $ area0_precip                <dbl> 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,~
## $ area1_base_temp             <dbl> -2.3, -1.8, 4.4, -3.3, -4.2, 3.2, -3.7, -0.6~
## $ area1_max_temp              <dbl> 6.0, 7.7, 6.8, 8.4, 8.4, 9.9, 9.4, 13.0, 7.0~
## $ area1_min_temp              <dbl> -4.9, -2.3, -3.6, -4.3, -4.9, -3.1, -5.7, -1~
## $ area1_avg_temp              <dbl> -0.3, 3.7, 1.5, 1.9, 1.3, 3.5, 0.0, 8.0, 3.5~
## $ area1_precip                <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,~
## $ area2_base_temp             <dbl> 0.9, 3.5, 3.3, -0.6, -0.6, 6.2, 0.9, 2.4, 5.~
## $ area2_max_temp              <dbl> 9.0, 10.0, 8.3, 10.2, 8.4, 11.3, 9.7, 12.3, ~
## $ area2_min_temp              <dbl> -1.3, 2.4, -3.4, -1.0, -2.0, 1.9, -1.8, 2.1,~
## $ area2_avg_temp              <dbl> 4.4, 5.4, 2.7, 5.8, 3.7, 6.4, 4.0, 6.6, 4.3,~
## $ area2_precip                <dbl> 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,~
```

```
head(data)
```

```
## # A tibble: 6 x 34
##       date  crop daily~1 daily~2 avg_p~3 avg_p~4 low_p~5 high_~6 daily~7 daily~8
##      <dbl> <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 20130101     0     NA       NA     NA      NA       NA       NA       NA       NA
## 2 20130102     0  20712.       22   4606.  40040       12       10   180000   178800
## 3 20130103     0   3402.      541   2003.   7527.     404      137   180000   178800
## 4 20130104     0   6554.      279   2266.   8248.      79      200   180000   178800
## 5 20130105     0   4299.     1233   2882.   9706.     977      256   180000   178800
## 6 20130106     0   4669.      396   2192.  11533.     291      105   180000   177800
## # ... with 24 more variables: daily_min_wholesale_price <dbl>,
## #   daily_max_retail_price <dbl>, daily_avg_retail_price <dbl>,
## #   daily_min_retail_price <dbl>, export_weight <dbl>, export_amount_usd <dbl>,
## #   import_weight <dbl>, import_amount_usd <dbl>, trade_balance_usd <dbl>,
## #   area0_base_temp <dbl>, area0_max_temp <dbl>, area0_min_temp <dbl>,
## #   area0_avg_temp <dbl>, area0_precip <dbl>, area1_base_temp <dbl>,
## #   area1_max_temp <dbl>, area1_min_temp <dbl>, area1_avg_temp <dbl>, ...
```

The data contains 54057 observations and 34 variables. Each row corresponds to daily information for each crop, and the observations from the same month for each crop have the same values for the trade-related variables because they are monthly data. The missing values indicate that there were no market transactions on that day, such as holidays. While all variables seem numerical in the data, I convert `date` to Date object and `crop` to factors because `crop` denotes type of crops, though encoded numerically as ranging between 0 and 36.

```
data$date <- as.Date(as.character(data$date), "%Y%m%d")
data$crop <- as.factor(data$crop)
class(data$date)
```

```
## [1] "Date"
```

```
class(data$crop)
```

```
## [1] "factor"
```

```
length(unique(data$crop))
```

```
## [1] 37
```

There 37 crops in the data, which are sampled from the population of all crops traded at the wholesale produce auctions in Korea. However, I focus on two crops with the highest and lowest price volatility, respectively. I examine the standard deviations of `daily_avg_price` by crops to narrow them down. I also take into consideration the proportion of missing values when choosing the crops.

```
price_sd <- data %>%
  group_by(crop) %>%
  summarize(sd = sd(daily_avg_price, na.rm=TRUE)) %>%
  arrange(desc(sd)) #%>%
  # slice(c(1, n()))
```

```
#calculate the average percentage of missing values across all numerical variables by crops
na_prop <- data %>%
  group_by(crop) %>%
  summarise(across("daily_avg_price":"area2_precip",  ~mean(is.na(.)))) %>%
  transmute(crop, mean_na_prop = rowMeans(.[-1])) %>%
  arrange(desc(mean_na_prop))

price_sd <- price_sd %>%
  left_join(na_prop, by = "crop")
```

```
# extract top 5 crops with highest daily average price volatility
price_sd %>%
  slice_max(sd, n = 5)
```

```
## # A tibble: 5 x 3
##   crop      sd mean_na_prop
##   <fct> <dbl>        <dbl>
## 1 18    6457.        0.204
## 2 6     6125.        0.269
## 3 33    2536.        0.147
## 4 14    2491.        0.305
## 5 23    2379.        0.265
```

```
# extract 5 crops with lowest daily average price volatility
price_sd %>%
  slice_min(sd, n = 5)
```

```
## # A tibble: 5 x 3
##   crop      sd mean_na_prop
##   <fct> <dbl>        <dbl>
## 1 9      237.        0.125
## 2 30     263.        0.100
## 3 29     280.        0.125
## 4 10     285.        0.126
## 5 28     289.        0.124
```

Among the top 5 crops with highest volatility, crop 33 has the lowest proportion of missing values in the numerical columns. For the crops with lowest volatility, there is not much difference in the missing value proportions, so I choose the one with lowest standard deviation of price. Hence, I focus on crops 33 and 9 onward. Since the missing values indicate that there were no market transactions on that day, I remove those rows from the data.

```
data2 <- data %>%
  filter(crop %in% c(33,9)) %>%
  drop_na()

data2$crop <- drop.levels(data2$crop) #reset the factors after dropping them
```

```
nrow(data2)
```

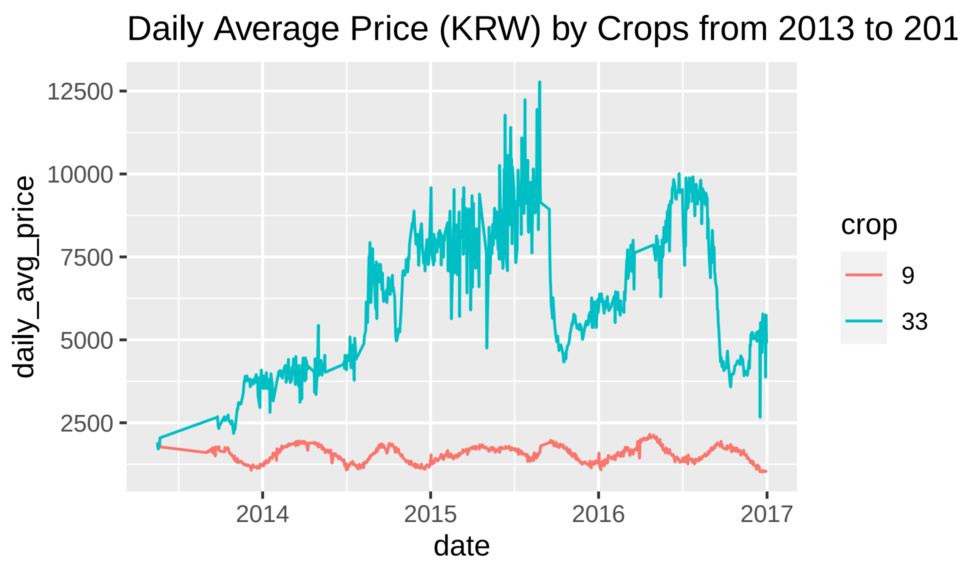
```
## [1] 1488
```

```
tally(~crop, data = data2)
```

```
## crop
##   9  33
## 790 698
```

After dropping the missing values, the data is reduced to 1488 rows. Crop 9 has 790 observations, and crop 33 has 698.

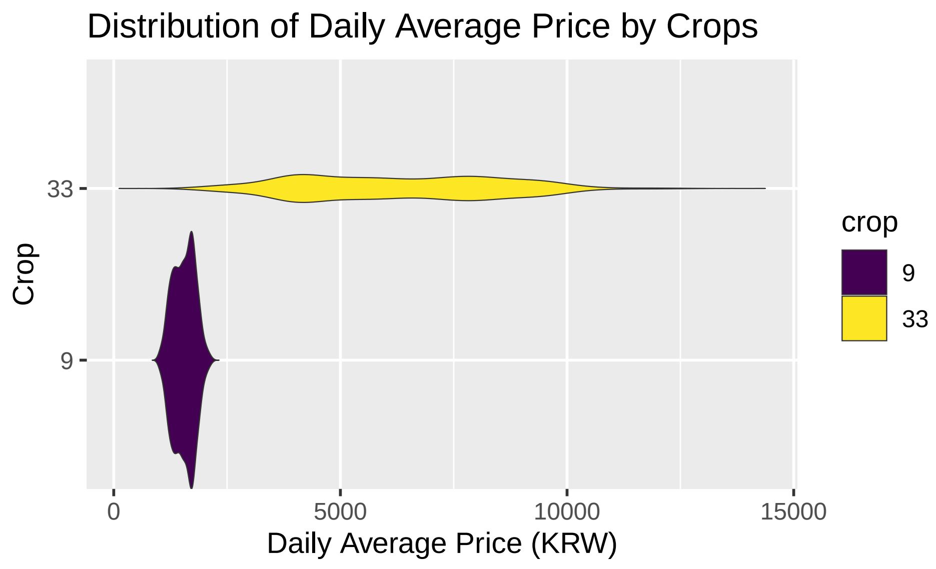
```
ggplot(data=data2, aes(x=date, y=daily_avg_price, color = crop)) +
  geom_line() +
  labs(title = "Daily Average Price (KRW) by Crops from 2013 to 2016 ")
```



The time series plot shows that crop 9 has both lower price range and lower price volatility than crop 33.

```
ggplot(data2, aes(x=crop, y=daily_avg_price, fill=crop)) +
    geom_violin(width=1.5, size=0.2, trim=FALSE) +
    scale_fill_viridis(discrete=TRUE) +
    scale_color_viridis(discrete=TRUE) +
    coord_flip() + # This switch X and Y axis and allows to get the horizontal version
    labs(title = "Distribution of Daily Average Price by Crops") +
    xlab("Crop") +
    ylab("Daily Average Price (KRW)")
```

```
## Warning: position_dodge requires non-overlapping x intervals
```



```r
favstats(daily_avg_price ~ crop, data = data2)
```

```
##   crop     min      Q1  median      Q3      max    mean       sd   n missing
## 1    9 1018.57 1355.67 1569.31 1736.97  2152.83 1553.14  239.631 790       0
## 2   33 1710.10 4347.29 6142.54 8017.47 12778.95 6284.94 2184.041 698       0
```

The violin plot shows that the distribution of daily average price for both crops is bimodal. The mean price is slightly lower than the median for crop 9, which is confirmed by its slightly left-skewed distribution. Crop 33 has a longer tail on the right, and the mean price is indeed greater than the median.

In order to find the variables that potentially represent crop quality, I use `daily_avg_price` as an indicator of the quality by creating 3 price ranges–high, mid, low. Before dividing the crops into price-based categories, I examine the relationship between `daily_avg_price` and other numerical variables.

```r
corr <- data2 %>%
  group_by(crop) %>%
  summarise(across("daily_avg_price":"area2_precip",
                   ~ cor(., daily_avg_price, use = "pairwise.complete.obs")))
corr2 <- corr %>%
  pivot_longer(!crop, names_to = "variables", values_to = "correlation")

corr2 %>%
  filter(variables != "daily_avg_price") %>%
  group_by(crop) %>%
  arrange(desc(abs(correlation)), .by_group = TRUE) %>%
  slice(1:5)
```

```
## # A tibble: 10 x 3
## # Groups:   crop [2]
##    crop  variables                 correlation
##    <fct> <chr>                           <dbl>
##  1 9     avg_price_high                  0.977
##  2 9     avg_price_low                   0.946
##  3 9     daily_avg_wholesale_price       0.789
##  4 9     daily_max_wholesale_price       0.780
##  5 9     daily_min_wholesale_price       0.762
##  6 33    avg_price_high                  0.942
##  7 33    daily_avg_wholesale_price       0.906
##  8 33    daily_min_wholesale_price       0.905
##  9 33    avg_price_low                   0.903
## 10 33    daily_max_wholesale_price       0.901
```

Because there are 34 variables, I print only the top 5 variables that show highest correlations with daily average price for each crop. In fact, `avg_price_high` and `avg_price_low` are somewhat redundant to `daily_avg_price` because they are simply the average of the prices that are less than or greater than `daily_avg_price` on that day. It also makes sense that all wholesale price-related variables show similar correlation with the target variable because they are essentially the same wholesale prices.

```
corr2 %>%
  filter(!grepl("price",variables)) %>%
  group_by(crop) %>%
  arrange(desc(abs(correlation)), .by_group = TRUE) %>%
  slice(1:6)
```

```
## # A tibble: 12 x 3
## # Groups:   crop [2]
##    crop  variables          correlation
##    <fct> <chr>                    <dbl>
##  1 9     trade_balance_usd       -0.330
##  2 9     import_amount_usd        0.330
##  3 9     import_weight            0.260
##  4 9     area0_max_temp           0.258
##  5 9     area2_max_temp           0.247
##  6 9     area2_avg_temp           0.236
##  7 33    daily_total_volume      -0.459
##  8 33    area2_min_temp           0.372
##  9 33    area0_min_temp           0.360
## 10 33    area2_avg_temp           0.358
## 11 33    area0_avg_temp           0.351
## 12 33    area2_base_temp          0.349
```

With price-related variables excluded, trade-related variables show the strongest correlation of about 0.3 with daily average price for crop 9, whereas `daily_total_volume` shows up as the most highly (negatively) correlated with the price for crop 33. This suggests that the daily price of crop 9 in the domestic market may be sensitive to the imports; that crop 33 may be price-elastic since the price and volume move in the opposite direction. Among the meteorological variables, temperature seems to be most correlated with the daily price. Given that plants typically favor lower temperatures, higher temperatures would negatively impact plant productivity, leading to the increase in prices due to supply issues.

```
data3 <- data2 %>%
  # filter(date != "2015-02-03") %>% #remove outliers to get balanced quantiles
  group_by(crop) %>%
  mutate(daily_avg_price_bin = cut(daily_avg_price,
                                   breaks = 3,
                                   labels = c("low","mid","high")))
```

```
tally(daily_avg_price_bin ~ crop, data = data3)
```

```
##                    crop
## daily_avg_price_bin   9  33
##                low  235 275
##                mid  411 336
##                high 144  87
```

```
favstats(daily_avg_price~daily_avg_price_bin+crop, data = data3)
```

```
##   daily_avg_price_bin.crop     min      Q1  median      Q3      max    mean
## 1                    low.9 1018.57 1195.15 1268.71 1330.26  1396.29 1258.63
## 2                    mid.9 1396.68 1511.11 1616.32 1705.01  1774.62 1605.91
## 3                   high.9 1776.14 1817.91 1864.29 1919.92  2152.83 1883.14
## 4                   low.33 1710.10 3712.70 4093.58 4544.20  5378.79 4040.71
## 5                   mid.33 5403.33 6266.41 7357.19 8065.82  9069.43 7222.47
## 6                  high.33 9091.24 9287.80 9554.33 9856.46 12778.95 9757.96
##          sd   n missing
## 1   93.1533 235       0
## 2  110.8017 411       0
## 3   84.9915 144       0
## 4  822.9886 275       0
## 5 1041.4096 336       0
## 6  731.6107  87       0
```

I divide each crop into three different price ranges–low, mid, high–which I believe is reflective of crop quality. For crop 9, there are 235 observations in "low" price range, 411 in "mid", and 144 in "high". For crop 33, there are 275 rows in low price range, 336 in mid, and 87 in high. For both crops, there are more observations tagged as "mid" than the other two categories. Since crop 9 is the crop with lowest price volatility, the difference between mean prices for each category is fairly small.

In this analysis, I do not remove any outliers because they might represent unusual market activity. Agricultural markets are volatile and can fluctuate significantly in response to political, regulatory, market, or macroeconomic conditions. I would like to investigate how the data helps identify crop quality even in the presence of such anomalies.

Methods

In order to classify price range, I apply three classification techniques–trees, random forest, and SVM–to two data sets split by crop type, and compare their performance in terms of error rates. For all three models, I use all numerical variables except `daily_avg_price` because it is redundant to the target variable. Trees perform binary splits based on the predictor variables, and the splits are made where one branch ends up with a majority set of class labels. Trees may not necessarily use all variables. The impurity measure determines whether or not a node at the end of a branch is pure. There are several parameters that can be specified for trees. `minsplit` specifies the number of observations required in a node in order for a split to occur; `minbucket` dictates the minimum number of observations required in a final node. We can also perform k-fold cross-validation by setting k via `xval`. I set `minsplit` to be 7 and `minbucket` 10 for both crops 9 and 33. For `xval`, I set it as the number of observations in the data for each crop to perform Jackknife validation to estimate true error rate, which means each observation takes the role of heldout sample once. I set `cp` = 0 to prevent pruning.

The second method is random forests, which are developed to combat the greediness and instability of trees. The `mtry` parameter–number of variables allowed at each split–addresses the first issue. Like trees, random forests also have an internal variable selection mechanism. Random forests use bootstrapping to build many trees, the number of which can be specified through the `ntree` parameter. For crop 9, I set `mtry` as 10 and use 1000 trees based on trial and error; for crop 33, `mtry` of 6 and 1500 trees. Bootstrapping is drawing a sample with replacement from the original data with the same number of observations. This approach results in the estimated TERs since the observations that are left out from the bootstrap samples–out of bootstrap (OOB)–form a natural test set. Random forests also provide the feature importance results based on majority voting–the number of times each variable was picked for each split. I combine this information with accuracy and Gini impurity to examine which features are considered important for classifying price ranges.

Lastly, I use Support Vector Machine (SVM), which performs classification by finding a hyperplane that maximizes the margin between the decision boundary and the training examples that are closest to the boundary. SVM is useful for classifying non-linearly separable data because it uses kernel methods that create nonlinear combinations of the original features to project the data onto a higher-dimensional space. While the solution from SVM is hard to interpret, the idea of finding a separation in higher dimensions often generates a better classification solution. I use radial basis function(`radial`) as the `kernel` for SVM because it is known to work well with non-linearly separable data. SVM also takes in parameters called `gamma` and `cost`. gamma defines the influence of a single training observation, and cost, often referred as C, controls the penalty for misclassification. Large gamma values lead to tighter decision boundary, and small C helps lower the overfitting of the model by letting it less strict about misclassification. For crop 9, my final choice of gamma and C is 0.03 and 10, respectively; 0.08 and 50 for crop 33. For SVM, I split the data into train and test sets. Since it is a time-series data, I train the model on the data over the span of 2013-2015 and test it on the data from 2016, instead of randomly shuffling it to make the split.

The classification analysis is based on a premise that price is reflective of crop quality because I group the data into 3 bins based on `daily_avg_price` and use them as the target variable representing the quality. In addition to price, other variables in the data such as weather information could be an important indicator of crop quality because they impact the crop growth. While there is no observed information on crop quality in the data, I conduct exploratory factor analysis for each crop to find the latent factors that help identify quality, under the assumption that there may be some underlying relationship between the manifest variables and quality. I run maximum likelihood factor analysis, which requires the assumption of multivariate normality for the data. While we know that the condition may not be satisfied from the preliminary analysis where the distribution of `daily_avg_price` was not normal, we proceed with caution because the results will be used as a guide for exploratory factor analysis. For factor analysis, I remove redundant factors discovered from the preliminary analysis because the `factanal()` function wasn't able to find a solution when all the variables were used. Given 9 variables, I run sequential hypothesis tests to determine the number of factors to keep since the number at which p-value becomes insignificant suggests that no more factors are needed. For both crops, I use four factors. A factor solution is obtained by finding communality and uniqueness. Communality is the variance shared with the other variables via the common factors, and uniqueness measures the variability not shared with other variables. I use uniqueness to evaluate the fitness of the variables in the solution because high uniqueness indicates that the extracted factor is not contributing much to those variables. I examine the factor loadings to interpret the factors and see if they convey any latent characteristics about crop quality. A factor solution is unique up to rotation so that when rotated, each variable is highly loaded on at most one factor. The `factanal()` command finds a solution with varimax rotation by default, but I also fit another solution with promax rotation to see if I can obtain a more interpretable solution. In the final step of factor analysis, I compute factor scores using Bartlett's method and check if the factor scores recover the daily average price ranges used for classification.

Results

Classification - Trees

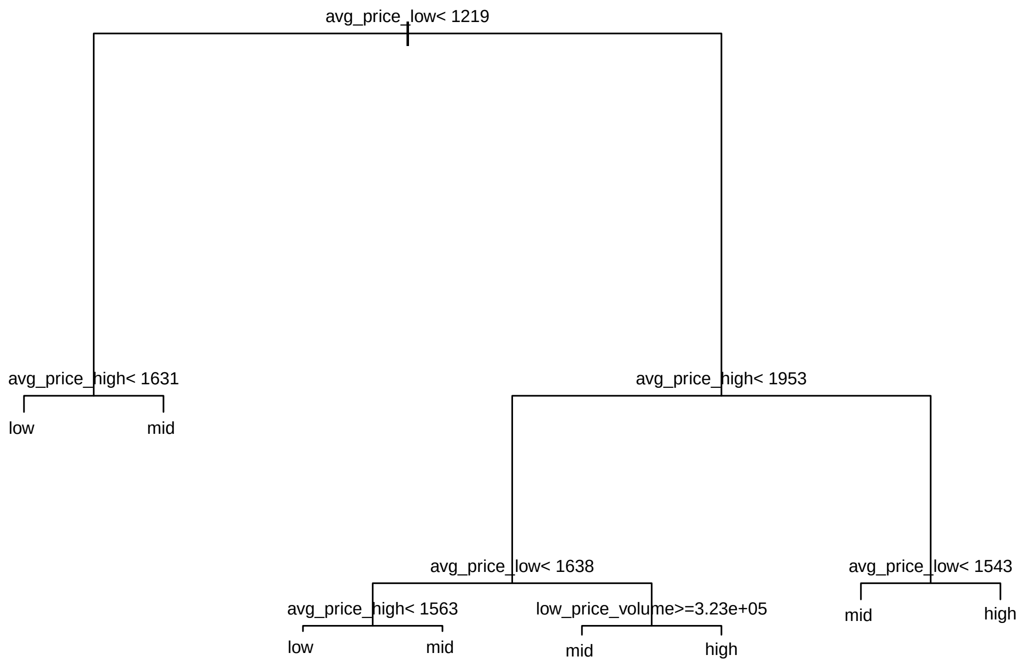
First, I compare three classification models for each crop in terms of the apparent error rates and estimated true error rates. For all three methods, all variables except `daily_avg_price` were used as predictors because its range is equivalent to the target variable `daily_avg_price_bin`.

```
data_9 <- data3 %>%
  filter(crop == "9") %>%
  ungroup() %>%
  select(-c(date, crop, daily_avg_price))

data_33 <- data3 %>%
  filter(crop == "33") %>%
  ungroup() %>%
  select(-c(date, crop, daily_avg_price))
```

```
set.seed(240)
bf.control <- rpart.control(minsplit = 7, minbucket = 10, xval = 790, cp = 0)
bf.treeorig <- rpart(daily_avg_price_bin ~ .,
                     data = data_9, method = "class", control = bf.control)
printcp(bf.treeorig)
```

```
##
## Classification tree:
## rpart(formula = daily_avg_price_bin ~ ., data = data_9, method = "class",
##     control = bf.control)
##
## Variables actually used in tree construction:
## [1] avg_price_high   avg_price_low    low_price_volume
##
## Root node error: 379/790 = 0.4797
##
## n= 790
##
##         CP nsplit rel error xerror    xstd
## 1 0.530343      0    1.0000 1.0000 0.03705
## 2 0.274406      1    0.4697 0.5673 0.03301
## 3 0.023747      2    0.1953 0.3061 0.02625
## 4 0.013193      4    0.1478 0.2480 0.02401
## 5 0.007916      6    0.1214 0.2084 0.02225
## 6 0.000000      7    0.1135 0.1847 0.02107
```

```
plot(bf.treeorig)
text(bf.treeorig, cex = 0.7)
```



```
#cp=0 -> better error rates
AER <- 0.4797 * 0.1135; AER
```

```
## [1] 0.054446
```

```
estTER <- 0.4797 * 0.1847; estTER
```

```
## [1] 0.0886006
```

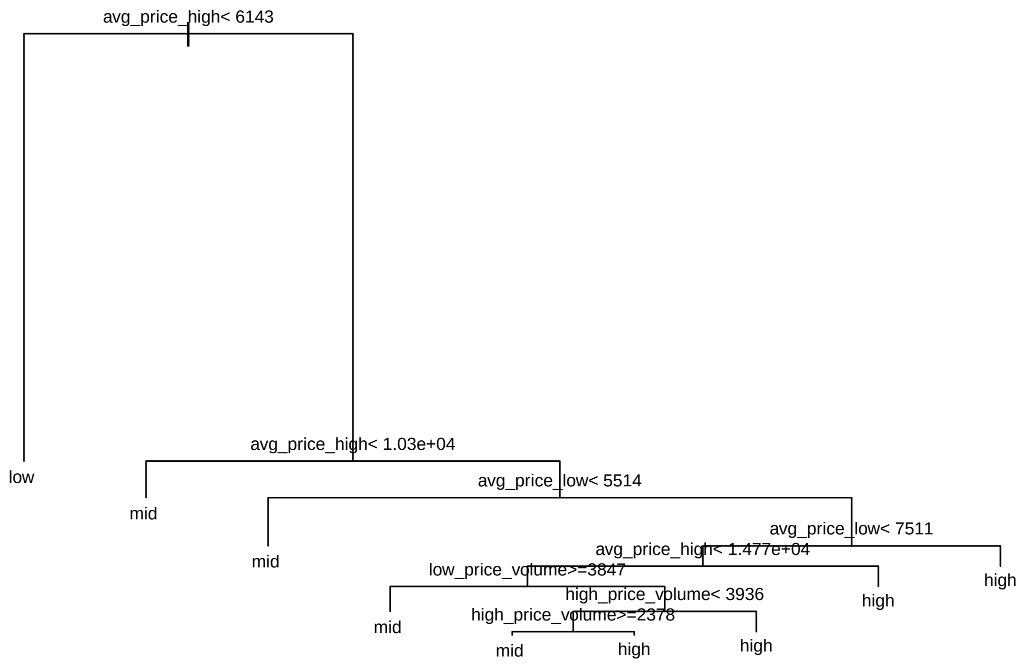
For crop 9, the tree model with minsplit of 7, minbucket of 10 yields an AER of 5% and an estimated TER of 8.8% from leave-one-out CV. The error rates are found relative to the root node error, which is the error rate if all observations were classified to the class with the majority observations–AER = root node error * rel error and estimated TER = root node error * xerror. While the estimated TER is slightly higher, the model seems to be performing fairly well.

```
set.seed(240)
bf.control <- rpart.control(minsplit = 7, minbucket = 10, xval = 698, cp = 0)
bf.treeorig <- rpart(daily_avg_price_bin ~ .,
                     data = data_33, method = "class", control = bf.control)
printcp(bf.treeorig)
```

```
##
```

```
## Classification tree:
## rpart(formula = daily_avg_price_bin ~ ., data = data_33, method = "class",
##     control = bf.control)
##
## Variables actually used in tree construction:
## [1] avg_price_high    avg_price_low     high_price_volume low_price_volume
##
## Root node error: 362/698 = 0.5186
##
## n= 698
##
##         CP nsplit rel error xerror    xstd
## 1 0.709945      0    1.0000 1.0000 0.03647
## 2 0.070442      1    0.2901 0.2956 0.02629
## 3 0.013812      3    0.1492 0.1575 0.01999
## 4 0.002762      6    0.1077 0.1326 0.01847
## 5 0.000000      8    0.1022 0.1409 0.01899
```

```
plot(bf.treeorig)
text(bf.treeorig, cex = 0.7)
```



```
AER <- 0.5186 * 0.1022; AER
```

```
## [1] 0.0530009
```

```
estTER <- 0.5186 * 0.1409; estTER
```

```
## [1] 0.0730707
```

The same model showed similar performance on crop 33–the AER of 5% and the estimated TER of 7%. For both crops, `avg_price_high` and `avg_price_low`, which are average prices falling into upper and lower ranges, dominate the trees. This makes sense because knowing the upper and lower price bands on that day would inform the average price range for that day.

Classification - Random Forests

Building many trees through random forest models tries to improve on the greediness and instability of trees. Each tree is generated using a bootstrap sample from the original data, where a random sample of variables are available to split on at each node. Observations left out of the bootstrap sample form a natural test data set and produce an out-of-bag (OOB) error rate.

```
set.seed(10)
bf.rf <- randomForest(daily_avg_price_bin ~ .,
                      data = data_9,
                      mtry = 10,
                      ntree = 1000,
                      importance = T,
                      proximity = T)
bf.rf
```

```
##
## Call:
##  randomForest(formula = daily_avg_price_bin ~ ., data = data_9,      mtry = 10, ntree = 1000, import
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 10
##
##         OOB estimate of  error rate: 6.84%
## Confusion matrix:
##      low mid high class.error
## low  222  13    0   0.0553191
## mid   13 386   12   0.0608273
## high   0  16  128   0.1111111
```

```
table(data_9$daily_avg_price_bin, predict(bf.rf, data_9))
```

```
##
##        low mid high
##   low  235   0    0
##   mid    0 411    0
##   high   0   0  144
```

For crop 9, I set mtry to be 10 and ntree 1000. While we obtained error rates relative to the root node error in trees, we use confusion matrix for random forests. OOB confusion matrix provides the estimated TER; AER is from the confusion matrix between the true class labels and the predicted classes from the `predict()` command. The AER is 0–all training observations are classified correctly, and the estimated TER is 6.84%, which is an improvement compared to the performance of trees.

```
set.seed(10)
bf.rf2 <- randomForest(daily_avg_price_bin ~ .,
                       data = data_33,
                       mtry = 6,
                       ntree = 1500,
                       importance = T,
                       proximity = T)
bf.rf2
```

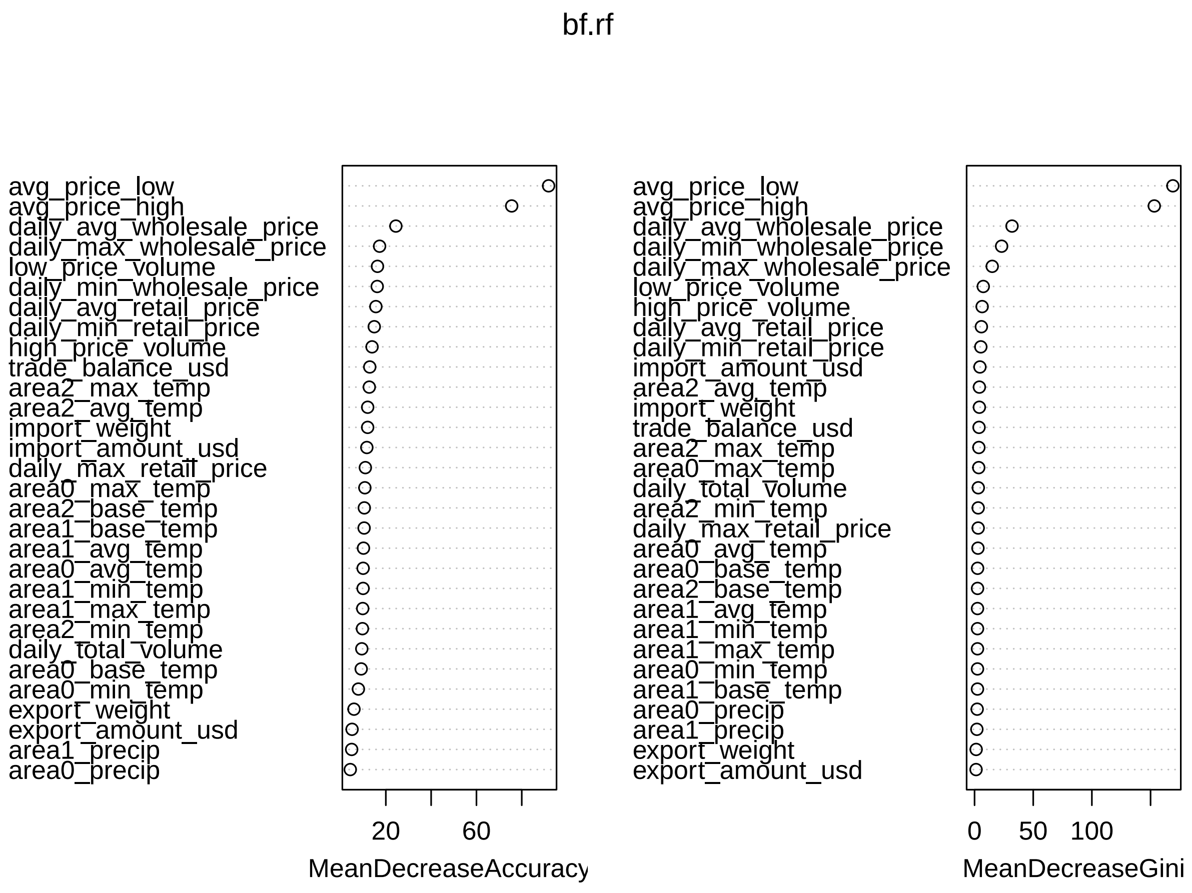
```
##
## Call:
##  randomForest(formula = daily_avg_price_bin ~ ., data = data_33,      mtry = 6, ntree = 1500, importa
##                Type of random forest: classification
##                      Number of trees: 1500
## No. of variables tried at each split: 6
##
##         OOB estimate of  error rate: 7.02%
## Confusion matrix:
##      low mid high class.error
## low  265  10    0   0.0363636
## mid    5 313   18   0.0684524
## high   0  16   71   0.1839080
```

```
table(data_33$daily_avg_price_bin, predict(bf.rf2, data_33))
```

```
##
##        low mid high
##   low  275   0    0
##   mid    0 336    0
##   high   0   0   87
```

For crop 33, mtry of 6 and ntree of 1500 for crop 33 yielded the lowest error rates. AER was also 0, and the estimated TER of 7.02%, which is similar from the trees.

```
varImpPlot(bf.rf)
```



```r
varImpPlot(bf.rf2)
```

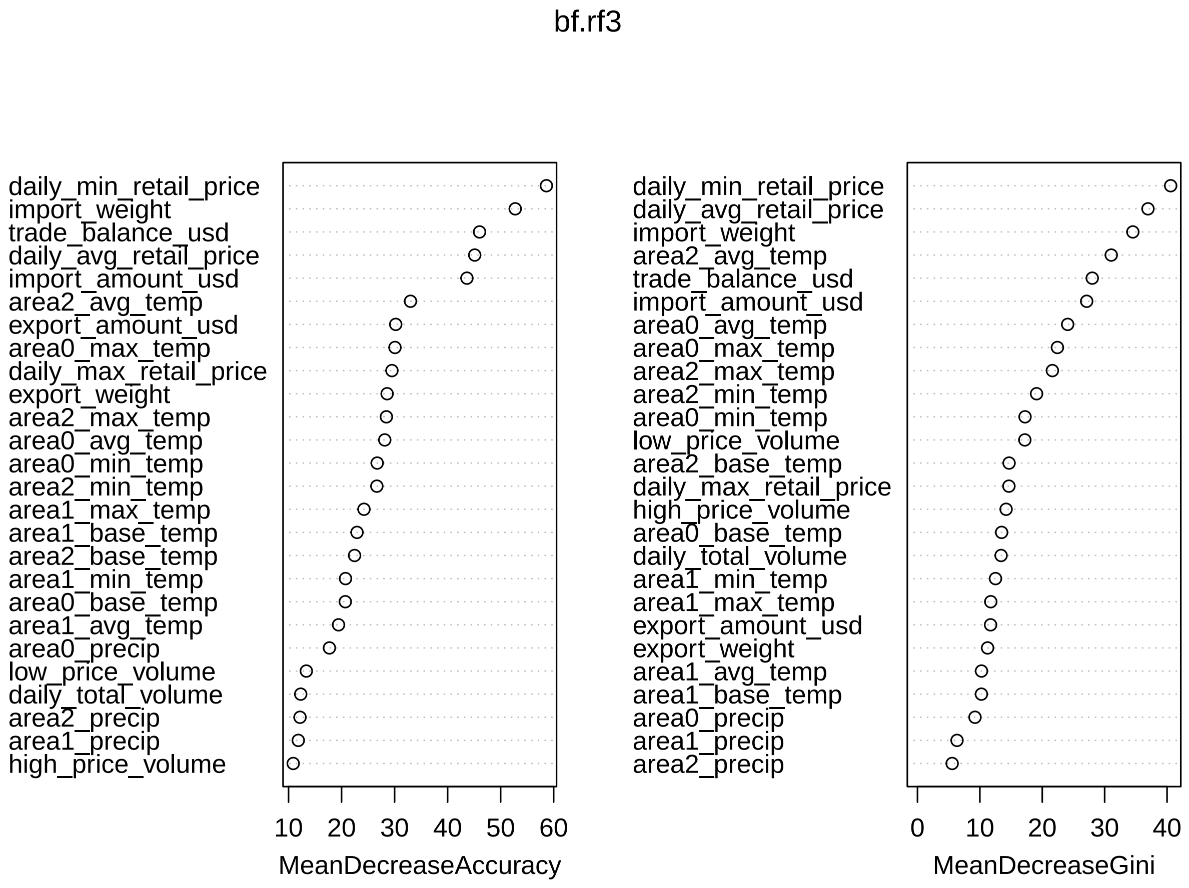
I also create variable importance plots where variables higher up are better because they result in larger mean decreases in accuracy or Gini coefficients that need to be minimized. For both crops, the upper and lower price ranges show up as the most important for classification, and there is agreement between the accuracy and Gini lists.

```r
set.seed(10)

data_9_rm <- data_9 %>%
  select(-c(avg_price_low, avg_price_high,
            daily_avg_wholesale_price, daily_min_wholesale_price,
            daily_max_wholesale_price))

bf.rf3 <- randomForest(daily_avg_price_bin ~ .,
                      data = data_9_rm,
                      mtry = 10,
                      ntree = 1000,
                      importance = T,
                      proximity = T)
bf.rf3
```

```
##
```

```
## Call:
##  randomForest(formula = daily_avg_price_bin ~ ., data = data_9_rm,      mtry = 10, ntree = 1000, impo
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 10
##
##         OOB estimate of  error rate: 18.35%
## Confusion matrix:
##      low mid high class.error
## low  201  33    1    0.144681
## mid   35 355   21    0.136253
## high   0  55   89    0.381944
```

```
varImpPlot(bf.rf3)
```



I fit another random forest model with the same parameters–`mtry` of 10 and `ntree` of 1000–after excluding the variables that appeared at the top of the feature importance plot because they capture redundant information as the target variable. The retail price and trade-related variables show up as the next most important variables. However, the error rate increased after excluding the redundant variables, which implies that the accuracy of the previous model was inflated.

Classification - Support Vector Machine (SVM)

For SVM, I manually split the data into train and test sets because they are time series data and price tends to follow temporal patterns. Train data is from 2013 to 2015 and test data spans 2016.

```r
data9_train <- data3 %>%
  filter(crop == "9" & date>="2013-01-02" & date <= "2015-12-31" ) %>%
  ungroup() %>%
  select(-c(date, crop, daily_avg_price))

data9_test <- data3 %>%
  filter(crop == "9" & date>="2016-01-02" & date <= "2016-12-31" ) %>%
  ungroup() %>%
  select(-c(date, crop, daily_avg_price))
```

```r
data33_train <- data3 %>%
  filter(crop == "33" & date>="2013-01-02" & date <= "2015-12-31" ) %>%
  ungroup() %>%
  select(-c(date, crop, daily_avg_price))

data33_test <- data3 %>%
  filter(crop == "33" & date>="2016-01-02" & date <= "2016-12-31" ) %>%
  ungroup() %>%
  select(-c(date, crop, daily_avg_price))
```

```r
svm1 <- svm(daily_avg_price_bin ~ ., data = data9_train, gamma = 0.03, cost = 3,
            kernel = "radial")
summary(svm1)
```

```r
svm1 <- svm(data9_train$daily_avg_price_bin ~ ., data = data9_train, gamma = 0.03,
            cost = 50, kernel = "radial")
summary(svm1)
```

```r
svm1 <- svm(daily_avg_price_bin ~ ., data = data9_train, gamma = 0.03, cost = 10,
            kernel = "radial")
summary(svm1)
```

```
##
## Call:
## svm(formula = daily_avg_price_bin ~ ., data = data9_train, gamma = 0.03,
##     cost = 10, kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  10
##
## Number of Support Vectors:  209
##
##  ( 53 102 54 )
##
```

```
##
## Number of Classes:  3
##
## Levels:
##  low mid high
```

```
svm1predtrain <- predict(svm1, data9_train)
svm1predtest <- predict(svm1, data9_test)
table(data9_train$daily_avg_price_bin, svm1predtrain)
```

```
##       svm1predtrain
##        low mid high
##   low  173   0    0
##   mid    0 280    1
##   high   0   4   90
```

```
table(data9_test$daily_avg_price_bin, svm1predtest)
```

```
##       svm1predtest
##        low mid high
##   low   46  10    6
##   mid    6  98   26
##   high   0   0   50
```

Through trial and error of gamma and cost, it appears that lower gamma and larger cost seem to yield lower error rates. Larger cost means that incorrect classifications are penalized more, but too much regularization would lead to overfitting. My final choice of gamma and cost for crop 9 is 0.03 and 10, respectively. The model has AER of (4+1)/548 = 0.9% and estimated TER of (10+6+6+26)/242 = 19.8%.

```
svm2 <- svm(daily_avg_price_bin ~ ., data = data33_train, gamma = 0.08, cost =25, kernel = "radial")
summary(svm2)
```

```
svm2 <- svm(daily_avg_price_bin ~ ., data = data33_train, gamma = 0.08, cost = 50, kernel = "radial")
summary(svm2)
```

```
##
## Call:
## svm(formula = daily_avg_price_bin ~ ., data = data33_train, gamma = 0.08,
##     cost = 50, kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  50
##
## Number of Support Vectors:  181
##
##  ( 67 81 33 )
##
##
```

```
## Number of Classes:  3
##
## Levels:
##  low mid high
```

```
svm2predtrain <- predict(svm2, data33_train)
svm2predtest <- predict(svm2, data33_test)
table(data33_train$daily_avg_price_bin, svm2predtrain)
```

```
##       svm2predtrain
##        low mid high
##   low  212   0    0
##   mid    0 226    0
##   high   0   0   44
```

```
table(data33_test$daily_avg_price_bin, svm2predtest)
```

```
##       svm2predtest
##        low mid high
##   low   58   5    0
##   mid   11  99    0
##   high   0  43    0
```

```
tally(~daily_avg_price_bin, data = data33_test)
```

```
## daily_avg_price_bin
##  low  mid high
##   63  110   43
```

```
tally(~daily_avg_price_bin, data = data33_train)
```

```
## daily_avg_price_bin
##  low  mid high
##  212  226   44
```

For crop 33, I set gamma as 0.08 and cost 50. Crop 33 has AER of 0, but estimated TER of (11+5+43)/216 = 27.3%. In fact, none of the high price ranges for crop 33 were classified correctly. Although there are twice more observations in the train test, the number of observations for high price range is very similar in both data. This would mean that price tended to be higher in 2016. Thus, it would be difficult for the model to detect unusually high prices.

Overall, random forest performed best for both crops: AER of 0 and estimated TER of 7.02% for crop 33 and 6.84% for crop 9. While I expected SVM to produce best classification results because it finds a hyperplane in higher dimensions, tree-based models were more effective for classifying the price ranges.

Factor Analysis

Next, I conduct exploratory factor analysis to find the underlying factors that may be reflective of crop quality. First, I tried using all the variables, but the `factanaly()` function wasn't able to find a solution. Thus, I removed the variables that are redundant to `daily_avg_price` and `daily_total_volume`, such as `avg_price_low` and `high_price_volume`. I also choose weather information from one of the primary growing areas only. For both crops, I use 9 variables for factor analysis.

```r
data9_2 <- data3 %>%
  filter(crop == "9") %>%
  ungroup()

data33_2 <- data3 %>%
  filter(crop == "33") %>%
  ungroup()
```

```r
cor(select(data9_2, -date, -crop, -daily_avg_price_bin), use = "pairwise.complete.obs")
cor(select(data33_2, -date, -crop, -daily_avg_price_bin), use = "pairwise.complete.obs")
```

While I do not include the correlation matrix output due to large number of variables, most variables seem to be moderately correlated. In particular, the variables that convey similar information, such as `trade_balance_usd` and `import_usd` or temperatures across three areas, show very strong correlations. The correlations between the observed variables suggest that there are some underlying relationships among the variables.

While we observed that `daily_avg_price` did not have a normal distribution, which means that multivariate normality assumption for factor analysis is violated, I proceed with caution because factor analysis will be used as a guide for exploratory purposes in this project.

```r
data9_3 <- data9_2 %>%
  select(daily_avg_price, daily_total_volume,daily_avg_wholesale_price,
         daily_avg_retail_price, export_weight, import_weight, trade_balance_usd,
        area0_avg_temp, area0_precip)

data33_3 <- data33_2 %>%
  select(daily_avg_price, daily_total_volume,daily_avg_wholesale_price,
         daily_avg_retail_price, export_weight, import_weight, trade_balance_usd,
        area0_avg_temp, area0_precip)
```

```r
q <- 9 #number of variables
k <- 5 #number of factors
df <- (q*(q+1)/2)-(q*(k+1)-(k*(k-1)/2))
df
```

```
## [1] 1
```

```r
sapply(1:5, function(f) factanal(data9_3, factors = f, start=rep(0, 9))$PVAL)
```

```
##    objective    objective    objective    objective    objective
## 5.09079e-263  2.08161e-51  5.04805e-12  2.52789e-03  4.50751e-04
```

Given 9 variables, we can fit at most 5 factors. I tested a number of factors from 1 to 5, but all of them had very low p-values. After trial and error, I choose a 3-factor solution because it represents a simple structure in which a few variables load highly on each factor.

```r
FAcar <- factanal(data9_3, factors = 3, start=rep(0,9))
print(FAcar)
```

```
##
## Call:
## factanal(x = data9_3, factors = 3, start = rep(0, 9))
##
## Uniquenesses:
##           daily_avg_price        daily_total_volume daily_avg_wholesale_price
##                     0.093                     0.745                     0.247
##    daily_avg_retail_price             export_weight             import_weight
##                     0.741                     0.682                     0.044
##         trade_balance_usd            area0_avg_temp              area0_precip
##                     0.028                     0.587                     0.853
##
## Loadings:
##                           Factor1 Factor2 Factor3
## daily_avg_price                    0.947
## daily_total_volume         0.504
## daily_avg_wholesale_price          0.844   0.200
## daily_avg_retail_price     0.248   0.445
## export_weight             -0.155           0.542
## import_weight              0.957   0.173  -0.101
## trade_balance_usd         -0.952  -0.255
## area0_avg_temp             0.153   0.241   0.576
## area0_precip                               0.383
##
##                Factor1 Factor2 Factor3
## SS loadings      2.195   1.961   0.824
## Proportion Var   0.244   0.218   0.092
## Cumulative Var   0.244   0.462   0.553
##
## Test of the hypothesis that 3 factors are sufficient.
## The chi square statistic is 79.54 on 12 degrees of freedom.
## The p-value is 5.05e-12
```

```
FAcar2 <- factanal(data9_3, factors = 3, start=rep(0,9), rotation = "promax")
print(FAcar2)
```

I first discuss the factor solution for crop 9. `area0_precip`, `daily_total_volume`, `daily_avg_retail_price`, and `area0_avg_temp` have high uniqueness, which indicates that the factors are not contributing much to these variables. The first factor has high loadings on `import_weight` and `trade_balance_usd` with opposite signs. The third factor has moderately high loadings on `export_weight` and `area0_avg_temp`. Both factors thus seem to be related to trades. Factor 2 loads highly on `daily_avg_price` and `daily_avg_wholesale_price`. This suggests that the second factor represents crop price. In other words, it appears that the factors simply explain the characteristics of the original variables in the data description–whether they are price-related or trade-related, not necessarily crop quality. I fit another 3-factor model with a promax rotation. There is a slight difference in factor loadings, but the overall interpretation of the factors did not change. Also, it is important to note that I had to set the starting values for the uniqueness to acquire the factor solution. This suggests that there may not be a strong or stable solution.

```
FAcar3 <- factanal(data33_3, factors = 2, start=rep(0,9))
print(FAcar3)
```

```
##
```

```
## Call:
## factanal(x = data33_3, factors = 2, start = rep(0, 9))
##
## Uniquenesses:
##           daily_avg_price        daily_total_volume daily_avg_wholesale_price
##                     0.176                     0.794                     0.005
##    daily_avg_retail_price             export_weight             import_weight
##                     0.100                     0.969                     0.089
##         trade_balance_usd             area0_avg_temp              area0_precip
##                     0.005                     0.851                     0.987
##
## Loadings:
##                           Factor1 Factor2
## daily_avg_price            0.886   0.198
## daily_total_volume        -0.454
## daily_avg_wholesale_price  0.976   0.207
## daily_avg_retail_price     0.911   0.265
## export_weight              0.176
## import_weight              0.101   0.949
## trade_balance_usd                 -0.997
## area0_avg_temp             0.362   0.133
## area0_precip                       0.105
##
##                Factor1 Factor2
## SS loadings      2.947   2.077
## Proportion Var   0.327   0.231
## Cumulative Var   0.327   0.558
##
## Test of the hypothesis that 2 factors are sufficient.
## The chi square statistic is 300.18 on 19 degrees of freedom.
## The p-value is 1.84e-52
```

```
FAcar4 <- factanal(data33_3, factors = 2, start=rep(0,9),  rotation = "promax")
print(FAcar4)
```

For crop 33, I fit a 2-factor solution that yields a simple structure.The same set of variables except `daily_avg_retail_price` showed highest uniqueness values as in crop 9. Another difference is high uniqueness for `export_weight`. Note that I also had to set limiting values for uniqueness to obtain a factor solution, which discounts the stability of the solution. Factor 1 had high loadings on all of the price-related variables, including the retail price. The second factor loaded highlly on trade-related variables. In other words, we can interpret the factors for the second crop in a similar manner as the other one. The factor loadings slightly changed when fitting with a promax rotation, but the factors did not even flip this time.

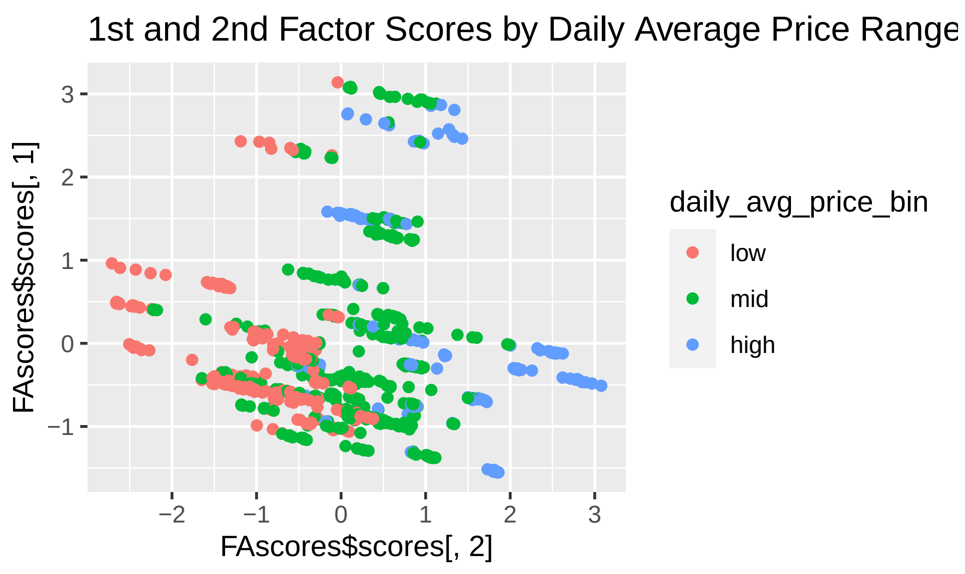
While factor analysis failed to uncover the latent factors that may be reflective of crop quality, I examine if the factor scores are capable of separating the daily average price ranges that were used for classification. I remove `daily_avg_price` from the factor model because it was used to create the price ranges we're trying to separate.

```
FAscores <- factanal(select(data9_3, -daily_avg_price), factors = 3, start=rep(0,8),
                     scores = "Bartlett")
FAscores
```

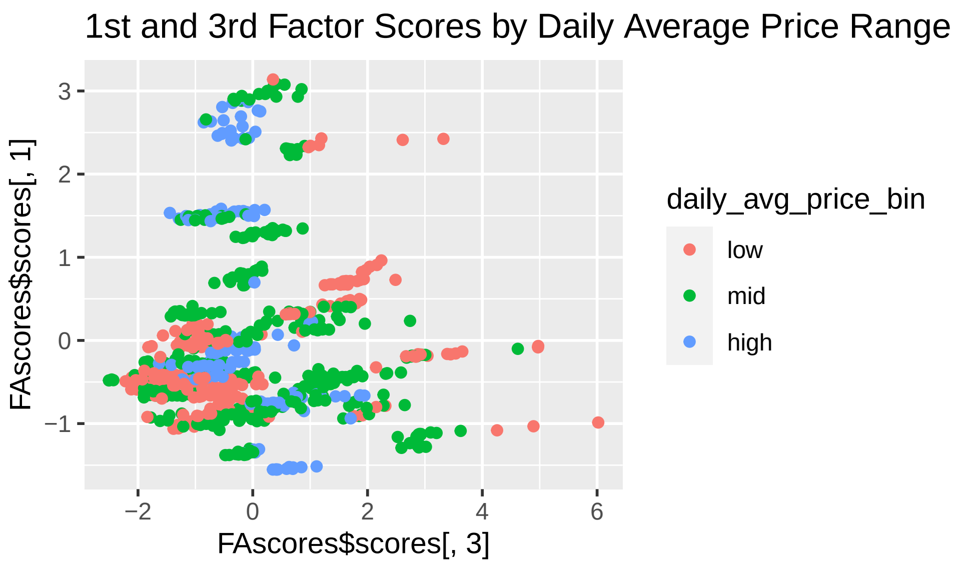
```
##
## Call:
```

```
## factanal(x = select(data9_3, -daily_avg_price), factors = 3,     start = rep(0, 8), scores = "Bartle
##
## Uniquenesses:
##        daily_total_volume daily_avg_wholesale_price    daily_avg_retail_price
##                     0.758                     0.005                     0.775
##             export_weight             import_weight         trade_balance_usd
##                     0.752                     0.050                     0.022
##            area0_avg_temp              area0_precip
##                     0.483                     0.865
##
## Loadings:
##                           Factor1 Factor2 Factor3
## daily_total_volume         0.492
## daily_avg_wholesale_price          0.984   0.165
## daily_avg_retail_price     0.276   0.386
## export_weight             -0.176           0.464
## import_weight              0.963   0.120
## trade_balance_usd         -0.967  -0.209
## area0_avg_temp             0.171   0.179   0.675
## area0_precip                               0.367
##
##                Factor1 Factor2 Factor3
## SS loadings       2.24   1.209   0.841
## Proportion Var    0.28   0.151   0.105
## Cumulative Var    0.28   0.431   0.536
##
## Test of the hypothesis that 3 factors are sufficient.
## The chi square statistic is 40.12 on 7 degrees of freedom.
## The p-value is 1.19e-06
```

```
gf_point(FAscores$scores[, 1] ~ FAscores$scores[, 2], color = ~ daily_avg_price_bin,
         data=data9_2) +
  labs(title = "1st and 2nd Factor Scores by Daily Average Price Range")
```



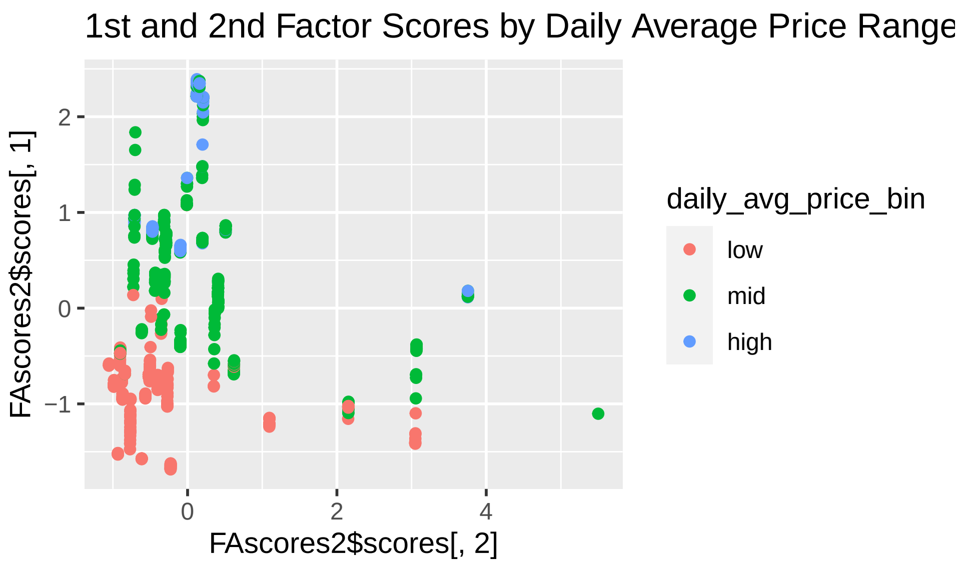
```
gf_point(FAscores$scores[, 1] ~ FAscores$scores[, 3], color = ~ daily_avg_price_bin,
         data=data9_2) +
  labs(title = "1st and 3rd Factor Scores by Daily Average Price Range")
```



The first and second factors represent trade information and wholesale price for crop 9. While there is some overlap, they are better at separating the daily average price ranges than the first and third factor scores. This suggests that wholesale transaction and trade information help explain daily average price of crops in the domestic market.

```
FAscores2 <- factanal(select(data33_3, -daily_avg_price), factors = 2, start=rep(0,8),
                      scores = "Bartlett")
```

```
gf_point(FAscores2$scores[, 1] ~ FAscores2$scores[, 2], color = ~ daily_avg_price_bin,
         data=data33_2) +
  labs(title = "1st and 2nd Factor Scores by Daily Average Price Range")
```



We obtain similar results for crop 33. Since the first and second factors also represent trade information and wholesale price, daily average price ranges are fairly well-separated in the factor space. However, the separation of the price ranges does not imply that the factor model has discovered the latent factors that explain crop quality.

Conclusion

In this analysis, classification techniques were applied to distinguish between crop price ranges under the assumption that price would be reflective of crop quality. Three models were considered–trees, random forests, and SVM–with all predictors. Based on their performance, random forests were chosen as the final model for both crops. Both had the AER of 0; the estimated TER was 6.84% for crop 9 and 7.02% for crop 33. The feature importance results from both models indicate that knowing about the average prices of the upper and lower price ranges help classify the which price range the crop belongs to on that day. While weather information serves as an important indicator of crop quality, none of the meteorological variables showed up as important variables. This may be attributable to two reasons. The weather-related variables in the data describe the weather conditions of three primary growing areas for each crop on the day when the transaction has occurred. What really impacts the crop quality is the meteorological factors along the growth cycle of the crops. Thus, I would like to examine further whether the meteorological data along the life cycle of cycles help better identify price-based crop quality. In addition, SVM had a higher estimated estimated TER than the tree-based models. I would also check if there is any improvement in the performance after scaling the data because the variables were on very different scales.

In order to find the underlying factors that may be reflective of crop quality, we also ran factor analysis. The factors that were found captured the inherent characteristics of the manifest variables but nothing about crop quality. However, since one of the factors represented crop price information, they were fairly good at separating price ranges. The fact that I had to use a subset of variables and set starting values for uniqueness to make the factor analysis function work suggests that there may not be a stable solution in the first place. One of the reasons behind this would be the violation of multivariate normality assumptions. I would try principal component factor analysis in the future since it doesn't require multivariate normality assumptions. Another reason why the factor analysis failed to discover latent information that recovers crop quality would be because price is not solely determined by quality. While crop quality is one of the determinants of crop price, it is also very volatile and easily impacted by macroeconomic, regulatory, and political conditions. In other words, it would be hard to pinpoint the underlying characteristics about crop quality given the current data because there are many complicated factors intertwined with price and there are not enough variables that explain crop quality.

Citations

Korea Agro-Fisheries & Food Trade Corporation. Agricultural Crop Price Prediction. AI Factory, 2022. https://aifactory.space/competition/detail/2091