Data 505

Jacob Plax

2025-01-28

Table of contents

Preface

This is a Quarto book of Jacob Plax's Work in Data 505: Applied Machine Learning

1 HW1:Wines of the PNW

Abstract:

This is a technical blog post of **both** an HTML file *and* .qmd file hosted on GitHub pages.

2 Setup

Step Up Code:

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```
wine <- readRDS(gzcon(url("https://github.com/cd-public/DSLM-505/raw/master/dat/wine.rds")))
  filter(province=="Oregon" | province=="California" | province=="New York") %>%
  mutate(cherry=as.integer(str_detect(description,"[Cc]herry"))) %>%
  mutate(lprice=log(price)) %>%
  select(lprice, points, cherry, province)
```

Explanataion:

1. Loads the tidyverse Library
2. Read in the data and save it in the wine dataframe
3. Filter the data to only include points where the province is Oregon, California, or New York.
4. Check if the desription contains the word cherry
5. Add a new column to the dataset called lprice that is the log of the existing price column.
6. Select only the columns lprice, points, cherry, and province to include in the dataset

3 Multiple Regression

3.1 Linear Models

First run a linear regression model with log of price as the dependent variable and 'points' and 'cherry' as features (variables).

```
m1 <- lm(lprice ~ points + cherry, data = wine)
predictions <- predict(m1, wine)
residuals <- wine$lprice - predictions
rmse <- sqrt(mean(residuals^2))
rmse
```

```
[1] 0.4687657
```

Explanataion:

1. Creates a linear regression model where lprice is the dependent variable and points and cherry are the independent variables. Essentially we are predicting the lprice using points and cherry.
2. We use the predict function to generate predictions from the linear model m1 using the wine dataset.
3. Calculate the residuals by subtracting the predicted values from the actual log prices.
4. Compute the Root Mean Squared Error (RMSE) to evaluate the model's performance.

RMSE: The RMSE measures how effective our model is. A lower RMSE indicates a better fit of the model to the data, meaning the predictions are closer to the actual values. In this case, the RMSE value is 0.4687657, which suggests how well our model is performing.

3.2 Interaction Models

Add an interaction between 'points' and 'cherry'.

```
m2 <- lm(lprice ~ points * cherry, data = wine)
predictions2 <- predict(m2, wine)
residuals2 <- wine$lprice - predictions2
rmse2 <- sqrt(mean(residuals2^2))
rmse2
```

```
[1] 0.4685223
```

1. Creates a linear regression model where lprice is the dependent variable and points, cherry, and their interaction (points * cherry) are the independent variables. Essentially we are predicting the lprice using points, cherry, and their interaction (points * cherry).
2. We use the predict function to generate predictions from the linear model m2 using the wine dataset.
3. Calculate the residuals by subtracting the predicted values from the actual log prices.
4. Compute the Root Mean Squared Error (RMSE) to evaluate the model's performance.

RMSE: The RMSE measures how effective our model is. A lower RMSE indicates a better fit of the model to the data, meaning the predictions are closer to the actual values. In this case, the RMSE value is 0.4685223, which suggests how well our model is performing.

3.2.1 The Interaction Variable

```
summary(m2)
```

```
Call:
lm(formula = lprice ~ points * cherry, data = wine)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.6432 -0.3332 -0.0151  0.2924  3.9645 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)   -5.659620   0.102252 -55.350  < 2e-16 ***
points         0.102225   0.001149  88.981  < 2e-16 ***
cherry        -1.014896   0.215812  -4.703 2.58e-06 ***
points:cherry  0.012663   0.002409   5.256 1.48e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.4686 on 26580 degrees of freedom
Multiple R-squared:  0.3062,    Adjusted R-squared:  0.3061
F-statistic:  3910 on 3 and 26580 DF,  p-value: < 2.2e-16
```

The coefficient of the interaction term (points:cherry) is positive, indicating that the effect of points on the log of price increases when the wine description contains the word "cherry". This suggests that wines described with "cherry" tend to have a higher price for the same number of points compared to those without the "cherry" description.

3.3 Applications

Determine which province (Oregon, California, or New York), does the 'cherry' feature in the data affect price most?

```r
wine_oregon <- wine %>% filter(province == "Oregon")
wine_california <- wine %>% filter(province == "California")
wine_newyork <- wine %>% filter(province == "New York")

model_oregon <- lm(lprice ~ points + cherry, data = wine_oregon)
model_california <- lm(lprice ~ points + cherry, data = wine_california)
model_newyork <- lm(lprice ~ points + cherry, data = wine_newyork)

summary_oregon <- summary(model_oregon)
summary_california <- summary(model_california)
summary_newyork <- summary(model_newyork)

coef_oregon <- summary_oregon$coefficients["cherry", "Estimate"]
coef_california <- summary_california$coefficients["cherry", "Estimate"]
coef_newyork <- summary_newyork$coefficients["cherry", "Estimate"]

coef_oregon
```

```
[1] 0.2203241
```

```r
coef_california
```

```
[1] 0.09566567
```

```r
coef_newyork
```

```
[1] 0.1517924
```

1. We create 3 subsets of the data for the 3 provinces.
2. We perform a linear regression model on all 3 sets of data.
3. We get the coefficents from the summary function.
4. The higher the coefficient the more important the cherry feature is.

The coefficients for the 'cherry' feature in each province are as follows:

- Oregon: 0.2203241
- California: 0.0956657
- New York: 0.1517924

The 'cherry' feature has the highest positive effect on the log of price in Oregon. This suggests that in this province, wines described with "cherry" tend to have a higher price compared to those without the "cherry" description.

4 Scenarios

4.1 On Accuracy

Imagine a model to distinguish New York wines from those in California and Oregon. After a few days of work, you take some measurements and note: "I've achieved 91% accuracy on my model!"

Should you be impressed? Why or why not?

```
province_counts <- wine %>%
  count(province)

total_count <- sum(province_counts$n)
baseline_accuracy <- max(province_counts$n) / total_count

baseline_accuracy
```

```
[1] 0.7174616
```

```
model_accuracy <- 0.91
model_accuracy > baseline_accuracy
```

```
[1] TRUE
```

The baseline accuracy is a little under 72% which is significantly lower than the 91% we are getting with our model. Therefore it is okay to be impressed by model since it is doing better than always predicating the most likely outcome.

4.2 On Ethics

Why is understanding this vignette important to use machine learning in an ethical manner?

It's important to undertsnad the context around our models. High accuracy doesn't mean much if it isn't actually that much better than the baseline accuracy. It's important for us to be able to evaluate the quality of our models rather than just looking at things like RMSE and P Values and assuming our models are doing a great job without underestanding the context of the data we are using. In theory this will also lead to us creating more intersting and more effective models.

4.3 Ignorance is no excuse

Imagine you are working on a model to predict the likelihood that an individual loses their job as the result of the changing federal policy under new presidential administrations. You have a very large dataset with many hundreds of features, but you are worried that including indicators like age, income or gender might pose some ethical problems. When you discuss these concerns with your boss, she tells you to simply drop those features from the model. Does this solve the ethical issue? Why or why not?

It doesn't solve the the ethical issues for a couple of reasons. First, there could be proxy variables in the dataset so even if we remove thing likes age, income, or gender we still might end up discriminating based on those things with other variables like zipcode for example. Second, even if those variables don't have proxies we still might be descriminatory because in the past people have been discriminatory and our model learns from those human biases. Instead we might actually want to include those variables so that we can then so how much they are affecting the model and adjust it accordingly to be more fair than if we had just removed them.

5 HW2:Wine Features

Abstract:

This is a technical blog post of **both** an HTML file *and* .qmd file hosted on GitHub pages.

6 Setup

Step Up Code:

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```
library(caret)
```

```
Loading required package: lattice

Attaching package: 'caret'

The following object is masked from 'package:purrr':

    lift
```

```
library(fastDummies)
```

```
Warning: package 'fastDummies' was built under R version 4.3.3
```

```
wine <- readRDS(gzcon(url("https://github.com/cd-public/D505/raw/master/dat/wine.rds")))
```

7 Feature Engineering

We begin by engineering an number of features.

1. Create a total of 10 features (including points).
2. Remove all rows with a missing value.
3. Ensure only log(price) and engineering features are the only columns that remain in the `wino` dataframe.

```
wino <- wine %>%
  mutate(lprice=log(price)) %>%
  mutate(country = fct_lump(country, 4)) %>%
  mutate(variety = fct_lump(variety, 4)) %>%
  select(lprice, points, country, variety) %>%
  drop_na()
head(wino)
```

```
# A tibble: 6 x 4
  lprice points country variety
   <dbl>  <dbl> <fct>   <fct>
1   2.71     87 Other   Other
2   2.64     87 US      Other
3   2.56     87 US      Other
4   4.17     87 US      Pinot Noir
5   2.71     87 Spain   Other
6   2.77     87 Italy   Other
```

Explanataion:

1. We create a new column `lprice` which is the logarithm of the `price` column.
2. We lump the `country` column into the top 4 most common countries and group the rest into "Other".
3. We lump the `variety` column into the top 4 most common varieties and group the rest into "Other".
4. We select only the `lprice`, `points`, `country`, and `variety` columns.
5. We remove any rows that contain missing values.
6. Finally, we display the first few rows of the resulting `wino` dataframe using the `head` function.

8 Caret

We now use a train/test split to evaluate the features.

1. Use the Caret library to partition the wino dataframe into an 80/20 split.
2. Run a linear regression with bootstrap resampling.
3. Report RMSE on the test partition of the data.

```r
set.seed(123)

trainIndex <- createDataPartition(wino$lprice, p = 0.8, list = FALSE)
wino_train <- wino[trainIndex, ]
wino_test <- wino[-trainIndex, ]

train_control <- trainControl(method = "boot", number = 100)
model <- train(lprice ~ ., data = wino_train, method = "lm", trControl = train_control)

predictions <- predict(model, wino_test)
rmse <- sqrt(mean((wino_test$lprice - predictions)^2))
rmse
```

```
[1] 0.4902949
```

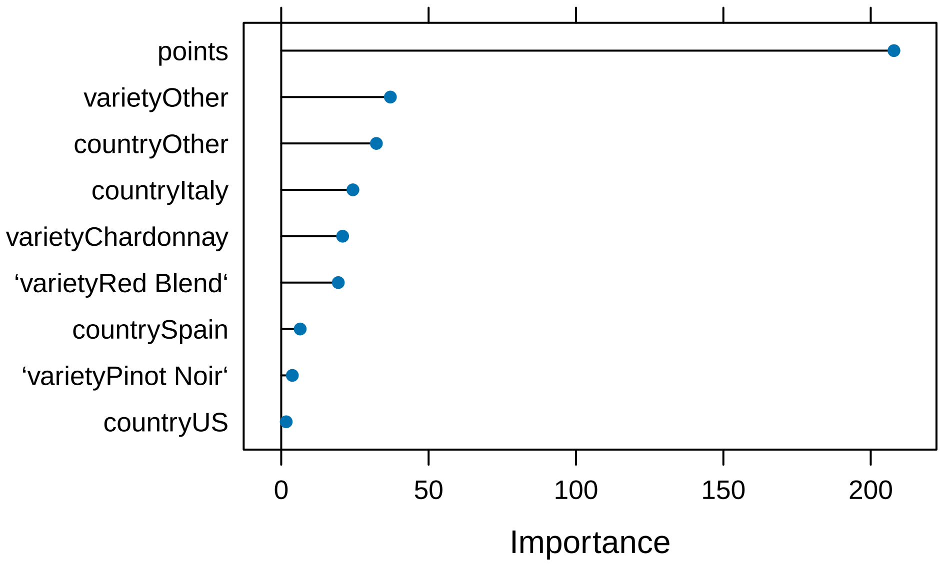
Explanation

1. We set a seed for reproducibility.
2. We create a training index that partitions the `wino` dataframe into an 80/20 split.
3. We create training and testing datasets using the partition index.
4. We define the training control using bootstrap resampling with 100 iterations.
5. We train a linear regression model using the training data and the defined training control.
6. We make predictions on the test data using the trained model.
7. We calculate the Root Mean Squared Error (RMSE) to evaluate the model's performance on the test data.

9 Variable selection

We now graph the importance of your 10 features.

```
plot(varImp(model, scale = FALSE))
```



Explanation

1. We use the `varImp` function from the `caret` package to calculate the importance of each feature in the model.
2. We plot the variable importance using the `plot` function, which helps us visualize the significance of each feature in predicting the target variable `lprice`.