
# Analysis

Andrew Redd, PhD.

R Bootcamp 2021

# Data

# Bordeaux Wine Data

```
bordeaux <- readRDS("data/bordeaux.rds")
```

## Variables

- `Wine` - the name of the wine.
- `Price` - The price of the wine in pounds sterling(£)
- `ParkerPoints` - the rating out of 100 given by Robert Parker (https://www.robertparker.com/).
- `CoatesPoints` - the rating out of 20 given by Clive Coates (http://www.clive-coates.com/)
- `P95andAbove` - a dummy variable, 1 if `ParkerPoints>=95`
- `FirstGrowth` - inditcator of if the wine is a first growth (https://en.wikipedia.org/wiki/First_Growth)
- `CultWine` - indicator of if the wine is a cult wine (https://en.wikipedia.org/wiki/Cult_wine)
- `Pomerol` - indicator for if the wine is from Pomerol (https://en.wikipedia.org/wiki/Pomerol), France
- `VintageSuperstar` - indicator if the wine is a superstar

# Hypothesis Tests

# T-test

Test if there is a difference in price for first growth wines.

```
(fg.test <- t.test(Price~FirstGrowth, data=bordeaux))
```

```
##
##  Welch Two Sample t-test
##
## data:  Price by FirstGrowth
## t = -3.0865, df = 9.9044, p-value = 0.01164
## alternative hypothesis: true difference in means between group FALSE and group TRUE is not equa
## 95 percent confidence interval:
##  -4544.7200  -731.1188
## sample estimates:
## mean in group FALSE  mean in group TRUE
##           815.0806           3453.0000
```

**Terminology**

Formulas, created with a ~, represent relationships. They can

- be one sided `~x`
  - Lambda functions `~log(.+1)`
- or two sided `y ~ x + z`
  - specify relationships or models
- often include functions
  - `y ~ x + log(z)`
  - `y ~ x + poly(z,3)`, a polynomial fit of degree 3 on `z`
- have special syntax
  - interaction `y ~ x:z`
  - crossing `y ~ a*b` is equal to `y ~ a + b + a:b`
  - nesting `y ~ a + b %in% a` or equivalently `y~a/b`

# Getting usable results from a model

The `fg.test` object is a htest object, which prints nicely but what if we want to include this in our table 1?

Try these:

```
str(fg.test) # get the underlying structure of the object.
glimpse(fg.test) # alternative to str that handles some objects better.
fg.test$p.value
getElement(fg.test, 'p.value')
fg.test[['p.value']]
```

# Linear Models

# Wine Model

We will rely on the normal approximation for proportions.

```
model <- lm( Price ~ . - Wine - P95andAbove, data=bordeaux)
model
```

```
##
## Call:
## lm(formula = Price ~ . - Wine - P95andAbove, data = bordeaux)
##
## Coefficients:
##          (Intercept)          ParkerPoints
##              -7390.78                 61.94
##         CoatesPoints       FirstGrowthTRUE
##               116.27               2001.41
##         CultWineTRUE           PomerolTRUE
##              4583.54                739.16
## VintageSuperstarTRUE
##              1424.58
```

Not really useful.

# Formula Creation Model

**Concept**

**Formula Subtraction**

`Price ~ . - Wine - P95andAbove` should be read as

*"Model Price by all variables **except** Wine and P95andAbove."*

# Summarizing Models

```
(model.summary <- summary(model))
```

# Exercise: Try the following

## Extracting Parts of the model

```
coef(model)
coef(summary(model))
deviance(model)
formula(model)
residuals(model)
```

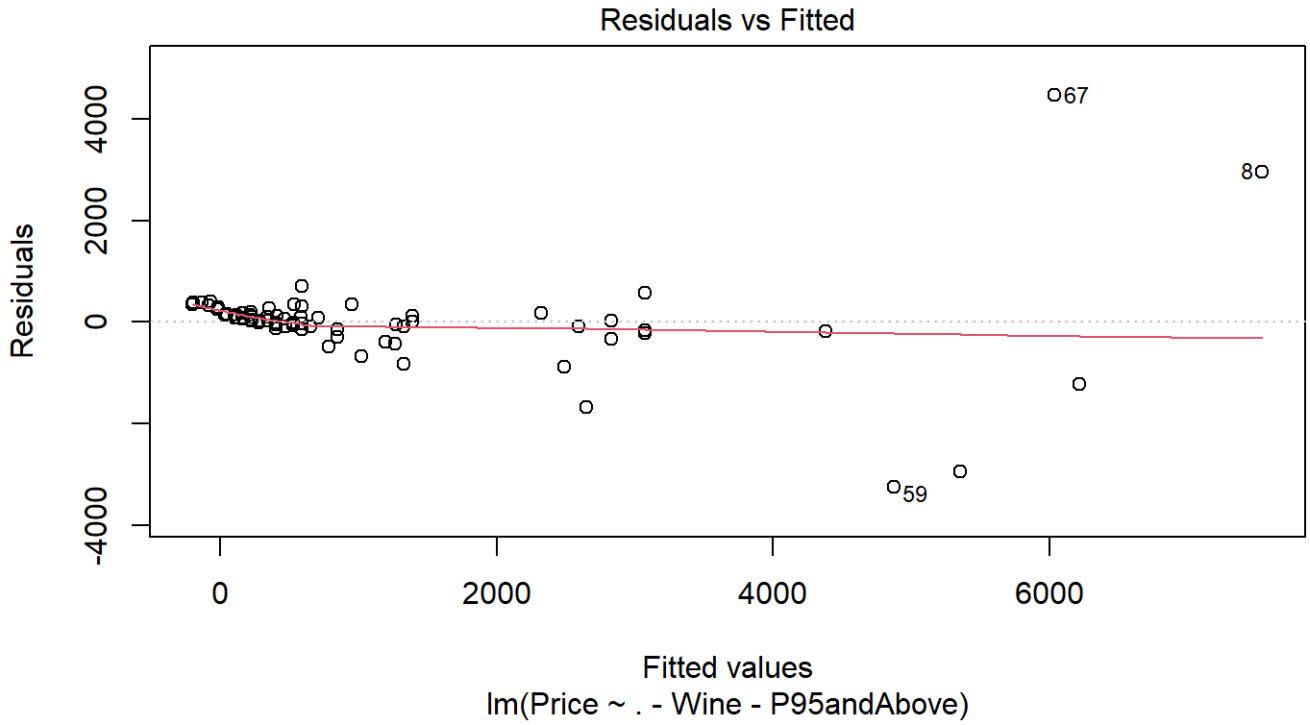
## Operations on models

```
summary(model)
plot(model)
predict(model)
vcov(model)
anova(model)
aov(model)
```

5:00

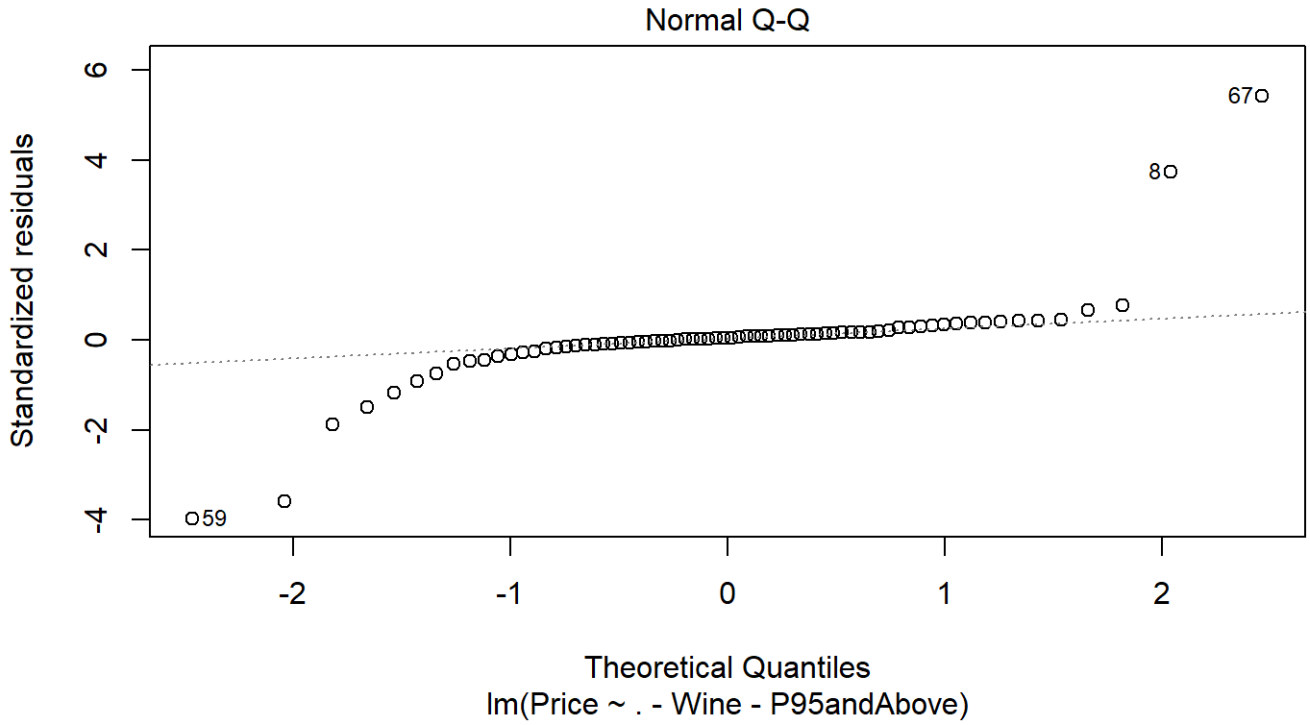
# Residual plots

```
plot(model, 1) #< Residuals vs. Fitted  | Goodness of fit
```



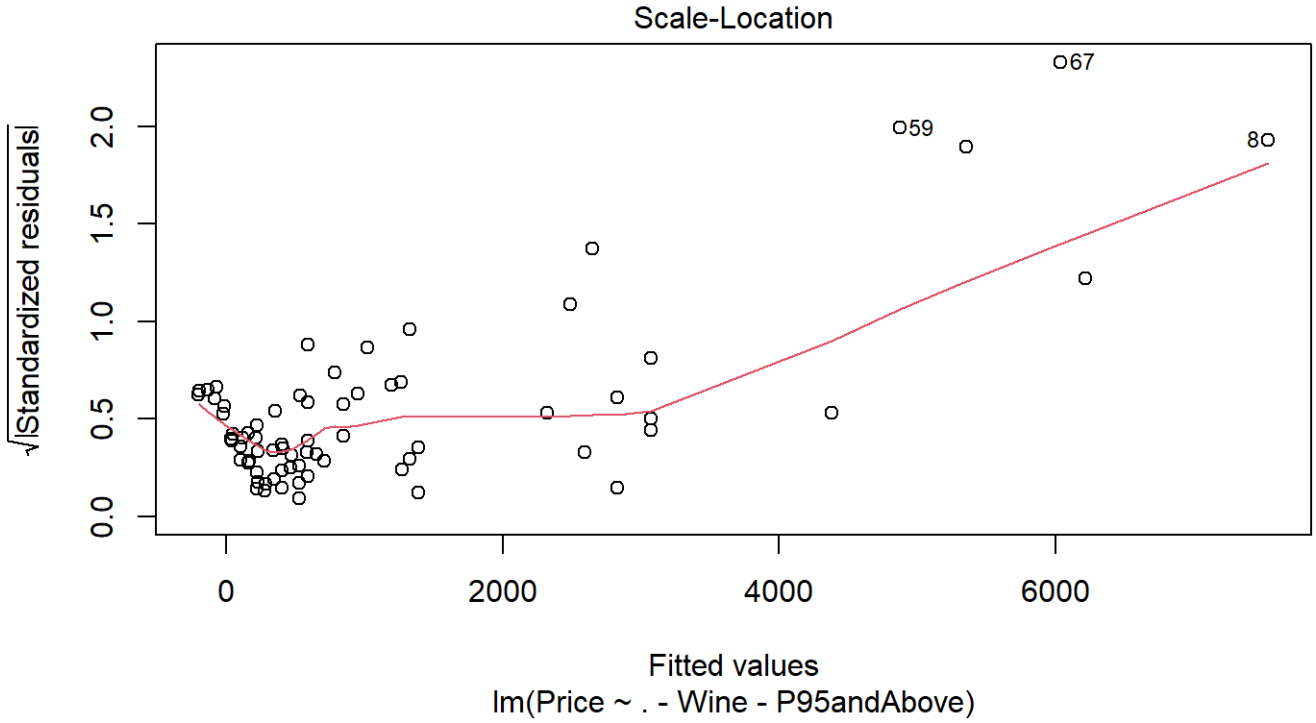
# Residual plots

```
plot(model, 2) #< Normal Q-Q                    | Appropriate error model
```



# Residual plots

```
plot(model, 3) #< Scale-Location          | Homoscedasticity
```



# Residual plots

```
plot(model, 5) #< Residual vs. Leverage | Influential points
```