Project 1 Combined

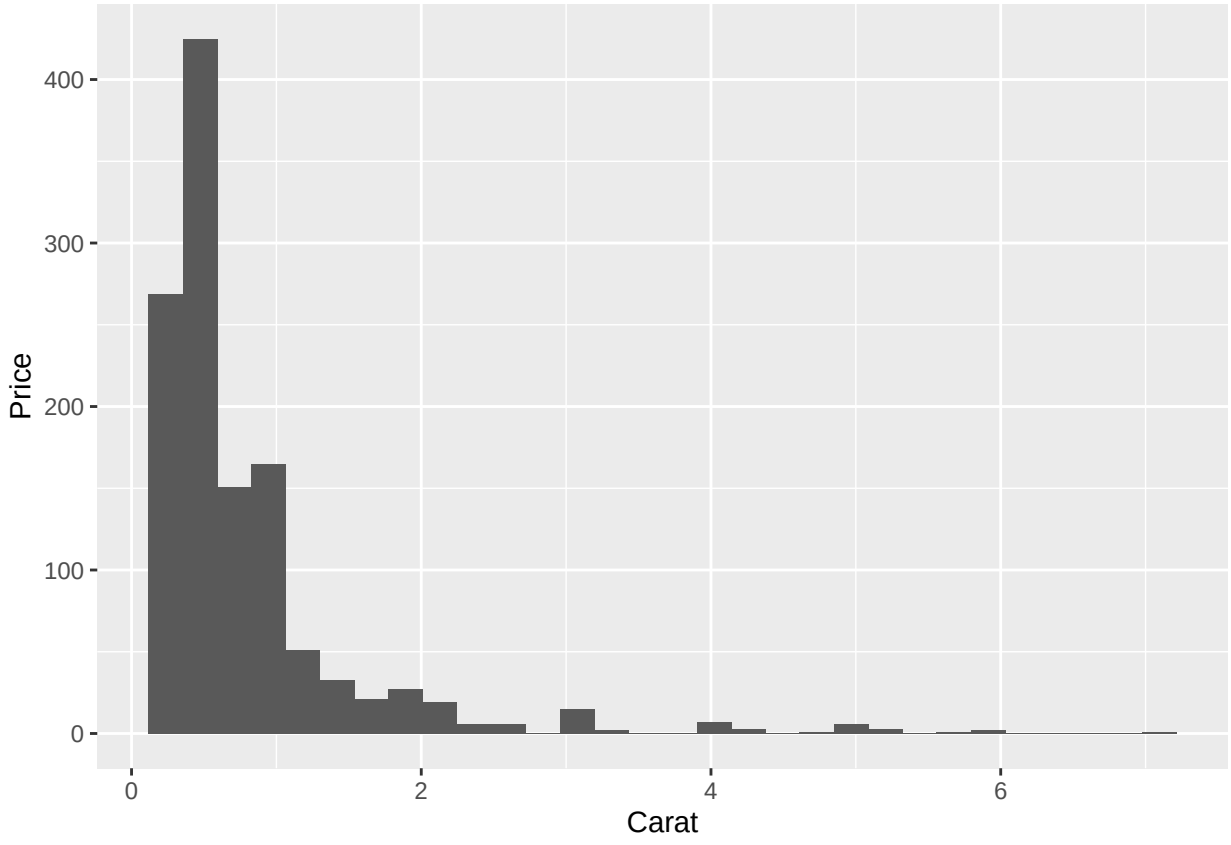
Alanna Hazlett

2024-03-20

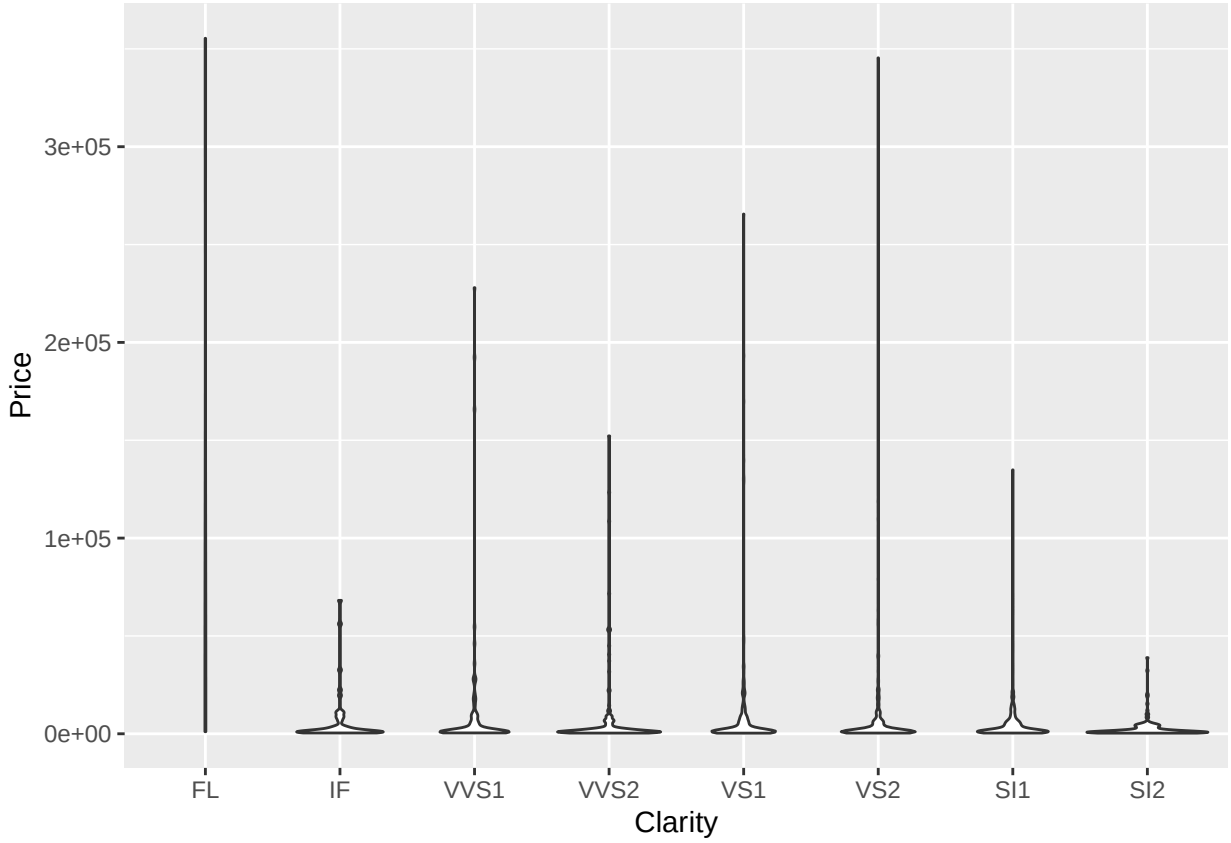
```
diamonds<-read.csv("diamonds4.csv", header=TRUE)
diamonds$color <- factor(diamonds$color, levels=c('D', 'E', 'F', 'G', 'H', 'I', 'J'))
diamonds$cut <- factor(diamonds$cut, levels=c('Astor Ideal', 'Ideal', 'Very Good', 'Good'))
diamonds$clarity <-factor(diamonds$clarity, levels=c('FL', 'IF', 'VVS1', 'VVS2', 'VS1', 'VS2', 'SI1', 'S
ggplot(diamonds, aes(x=carat, fill=price))+
  geom_histogram()+
  labs(x="Carat", y="Price")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

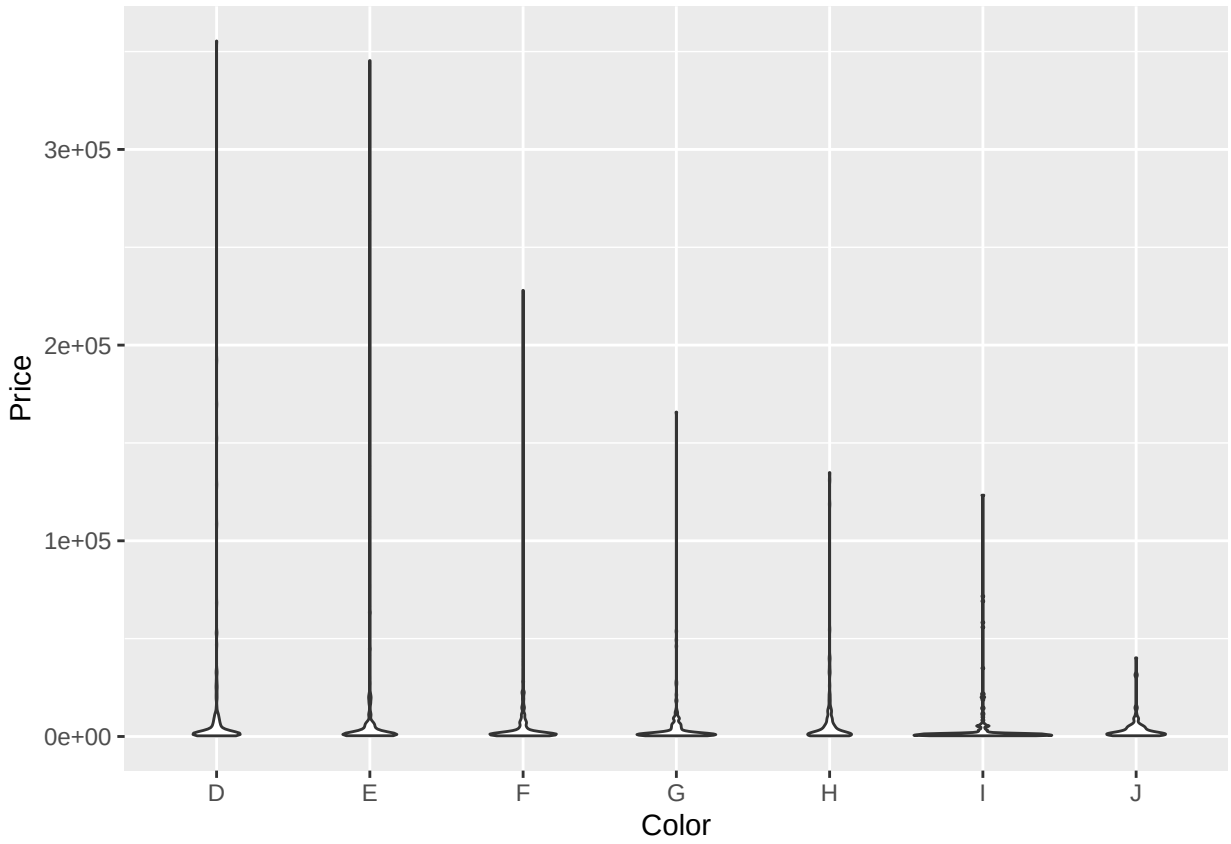
```
## Warning: The following aesthetics were dropped during statistical transformation: fill
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```



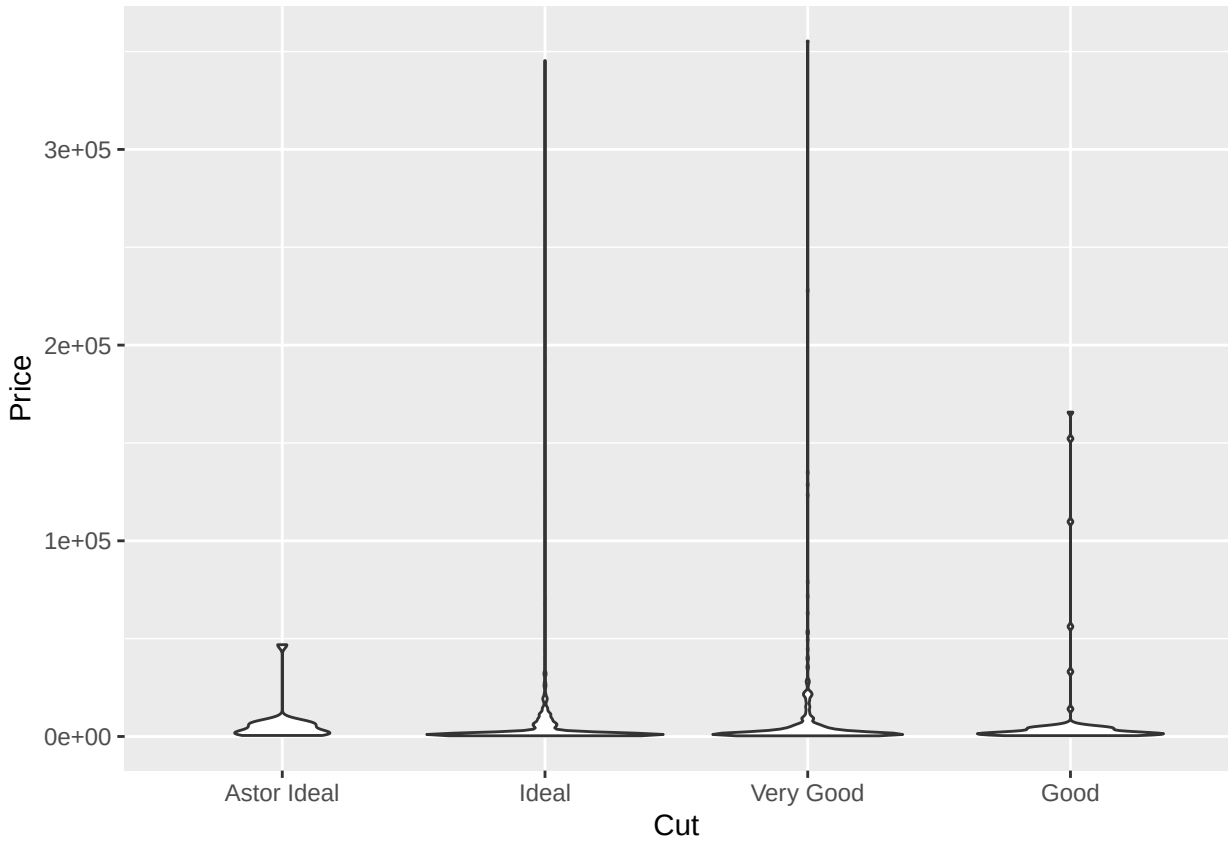
```
ggplot(diamonds, aes(x=clarity,y=price))+
  geom_violin()+
  labs(x="Clarity", y="Price")
```



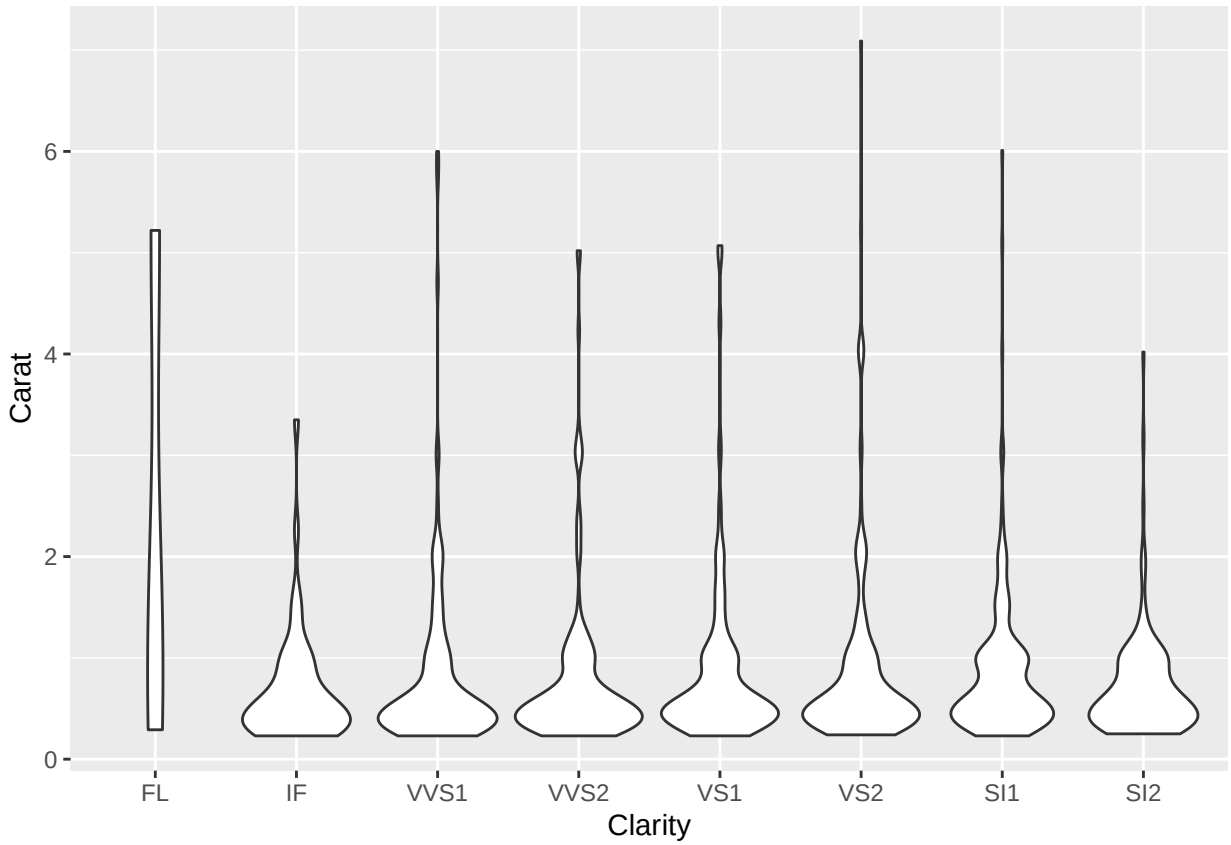

```
ggplot(diamonds, aes(x=color, y=price))+
  geom_violin()+
  labs(x="Color",y="Price")
```



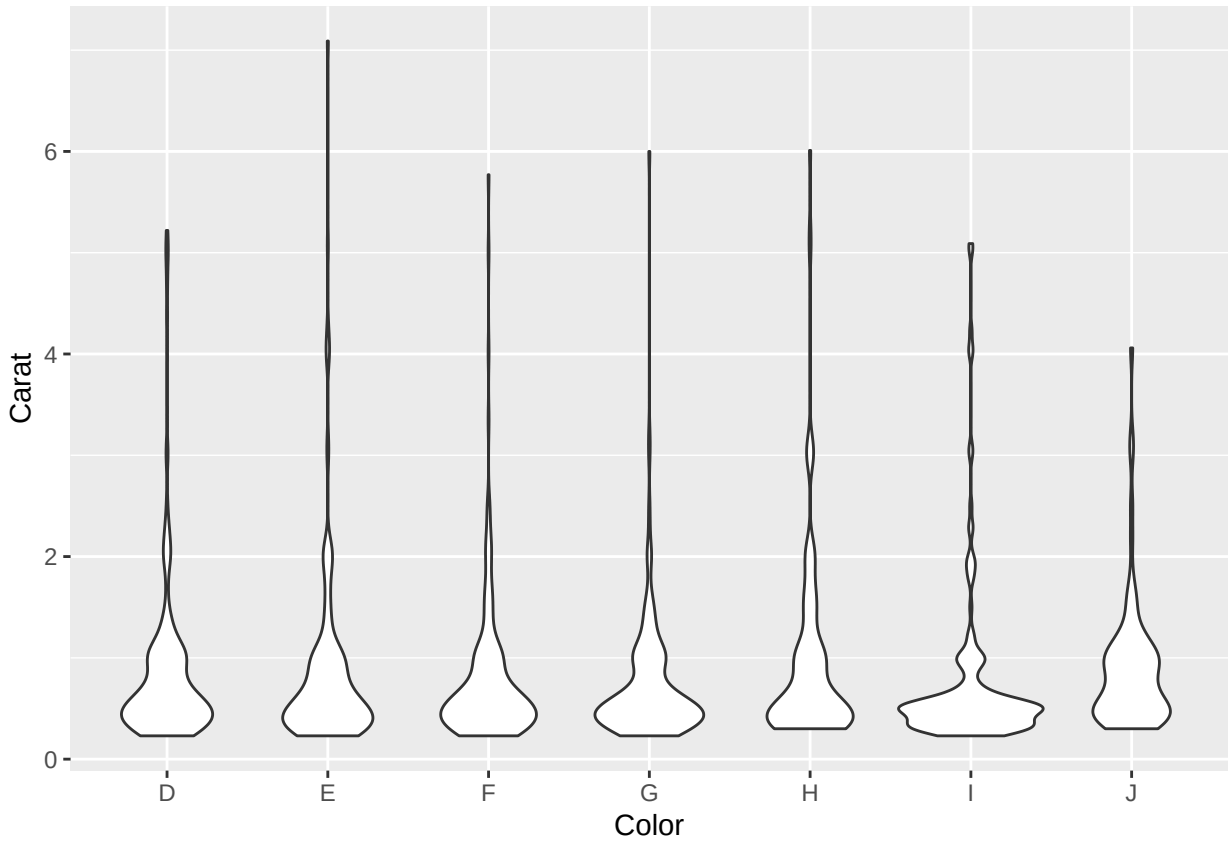

```
ggplot(diamonds, aes(x=cut, y=price))+
  geom_violin()+
  labs(x="Cut",y="Price")
```



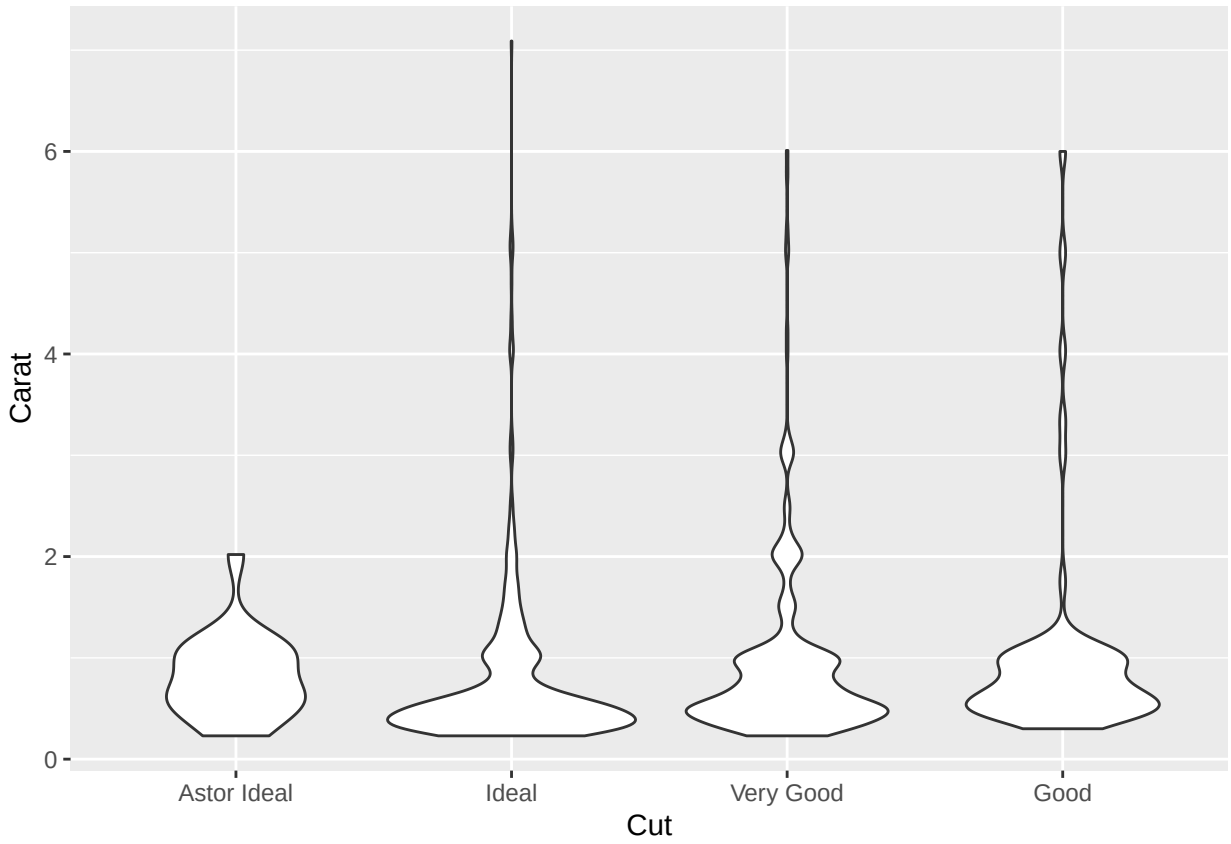

```
ggplot(diamonds, aes(x=clarity, y=carat))+
  geom_violin()+
  labs(x="Clarity",y="Carat")
```




```
ggplot(diamonds, aes(x=color, y=carat))+
  geom_violin()+
  labs(x="Color",y="Carat")
```

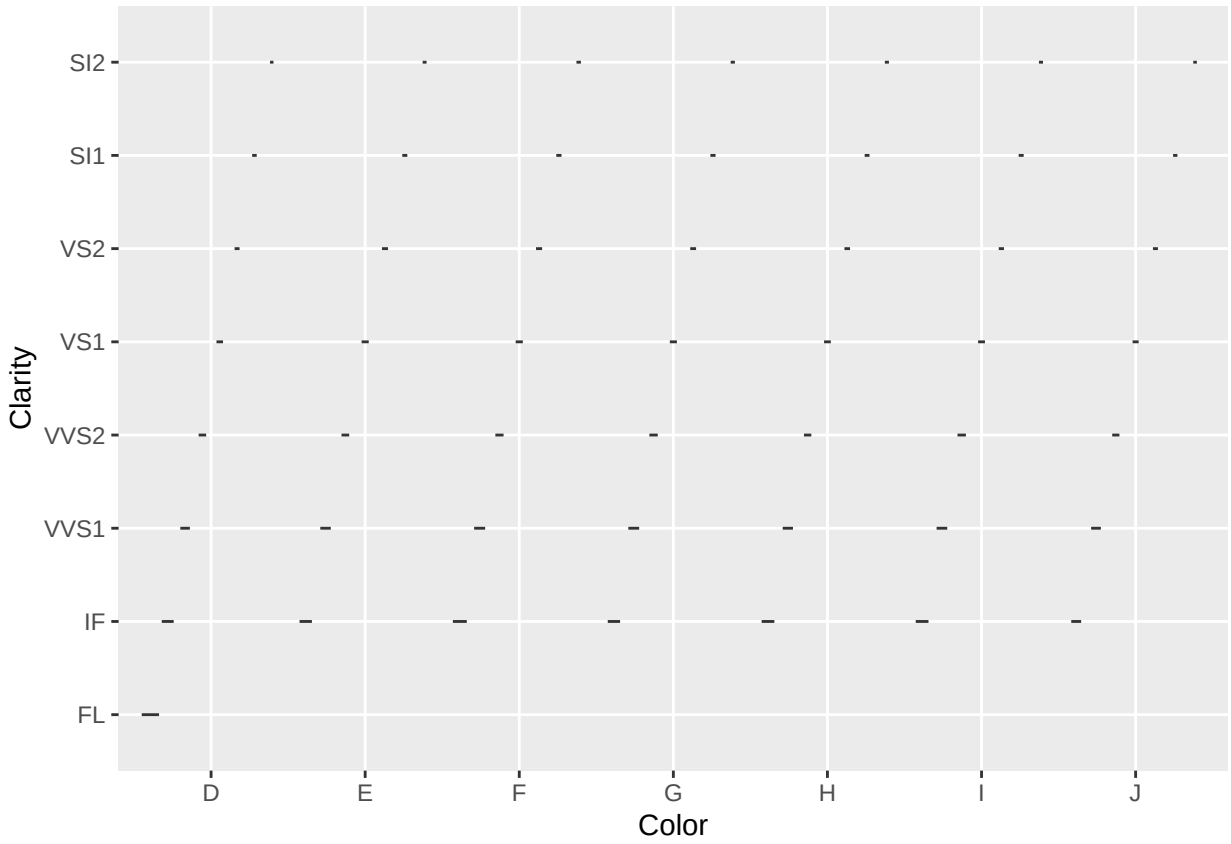



```
ggplot(diamonds, aes(x=cut, y=carat))+
  geom_violin()+
  labs(x="Cut",y="Carat")
```



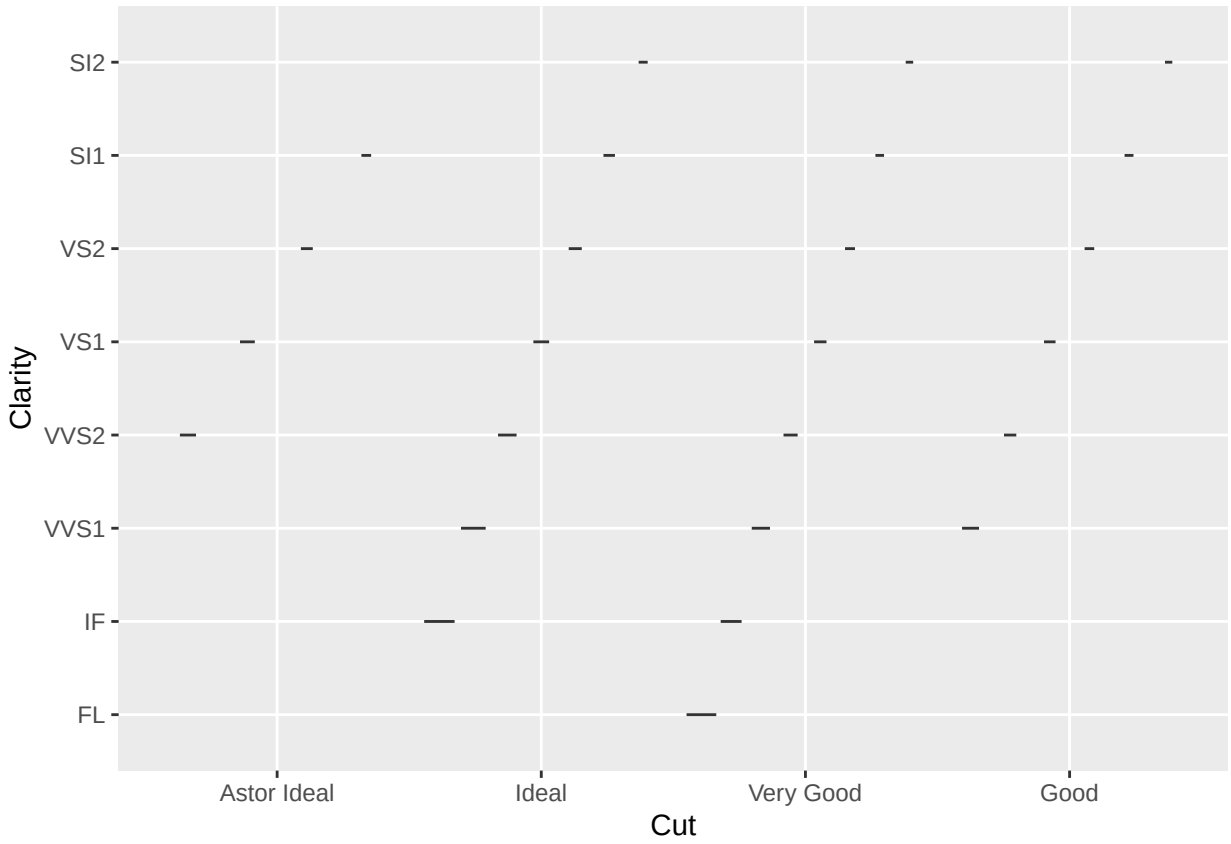

```
ggplot(diamonds, aes(x=color, y=clarity))+
  geom_violin()+
  labs(x="Color",y="Clarity")
```

```
## Warning: Groups with fewer than two data points have been dropped.
```

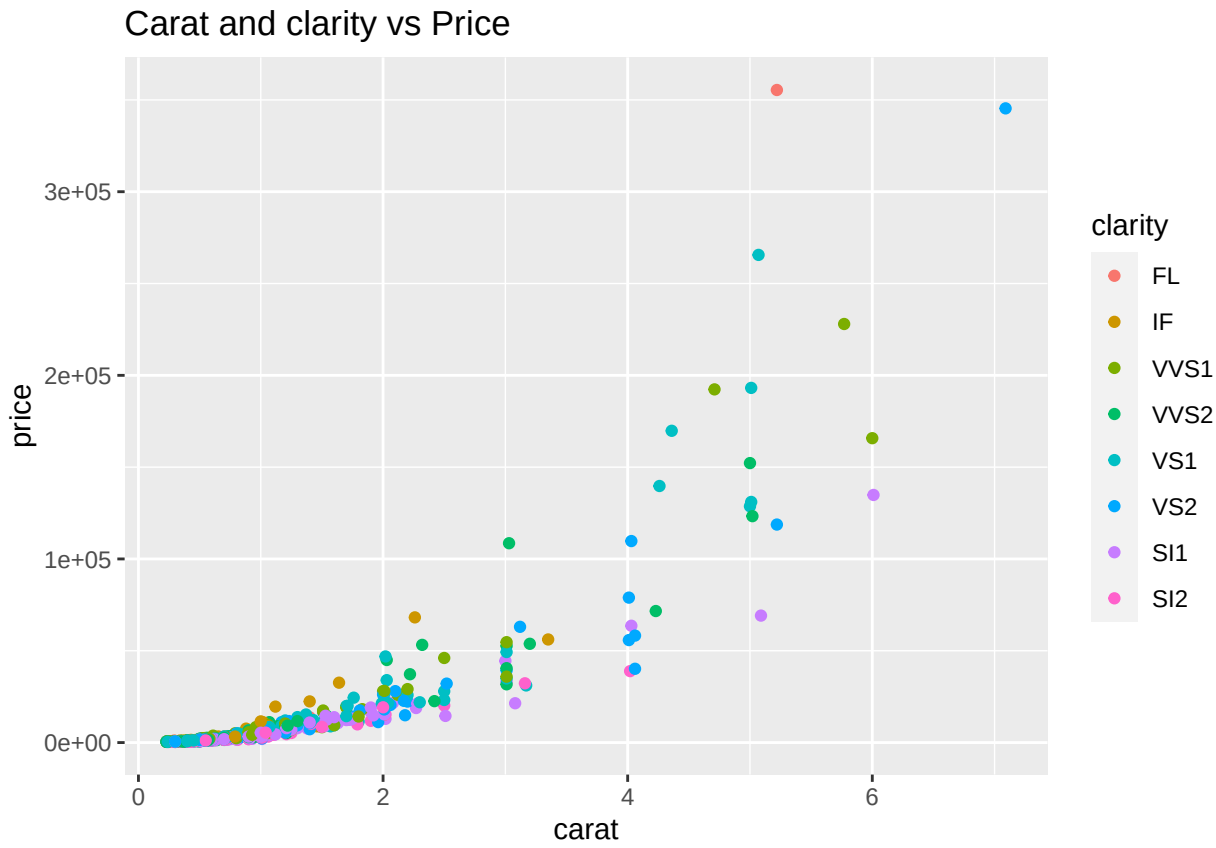



```
ggplot(diamonds, aes(x=cut, y=clarity))+
  geom_violin()+
  labs(x="Cut",y="Clarity")
```

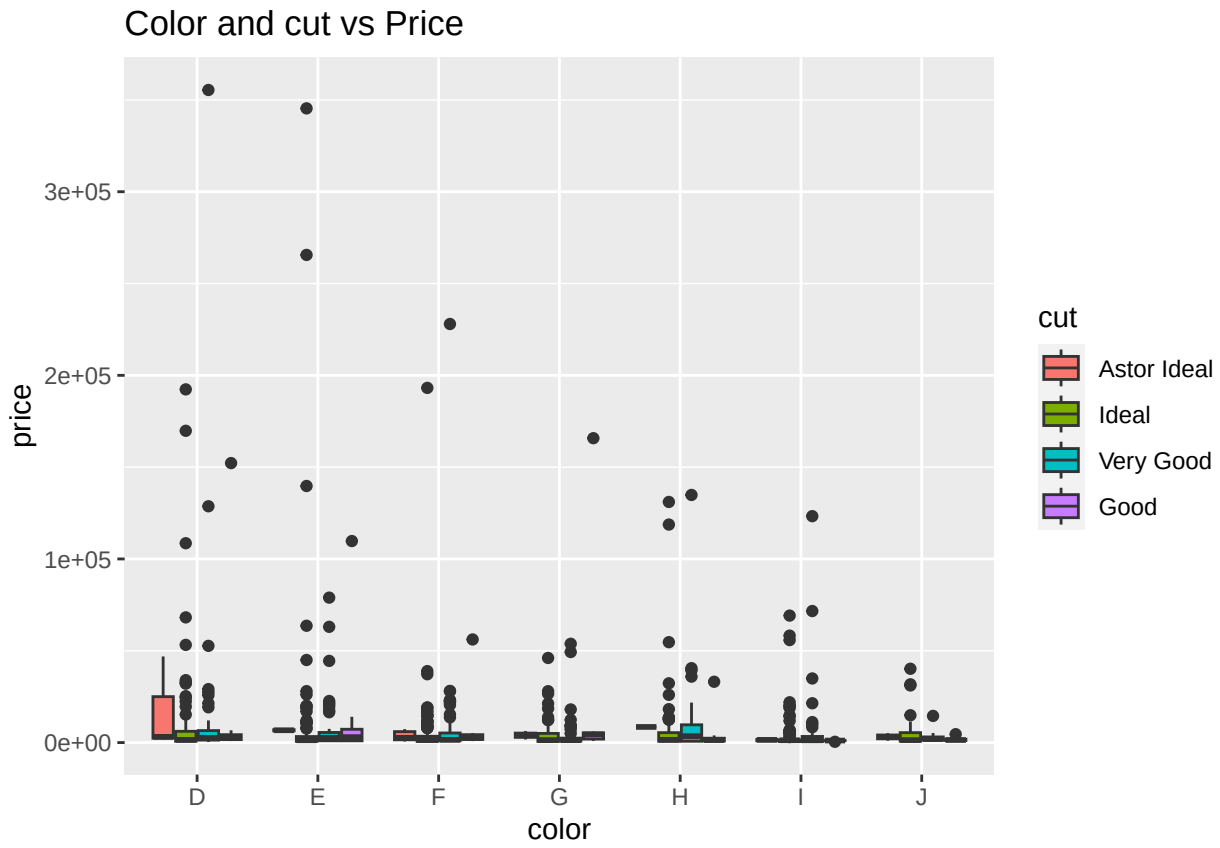
```
## Warning: Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
## Groups with fewer than two data points have been dropped.
```



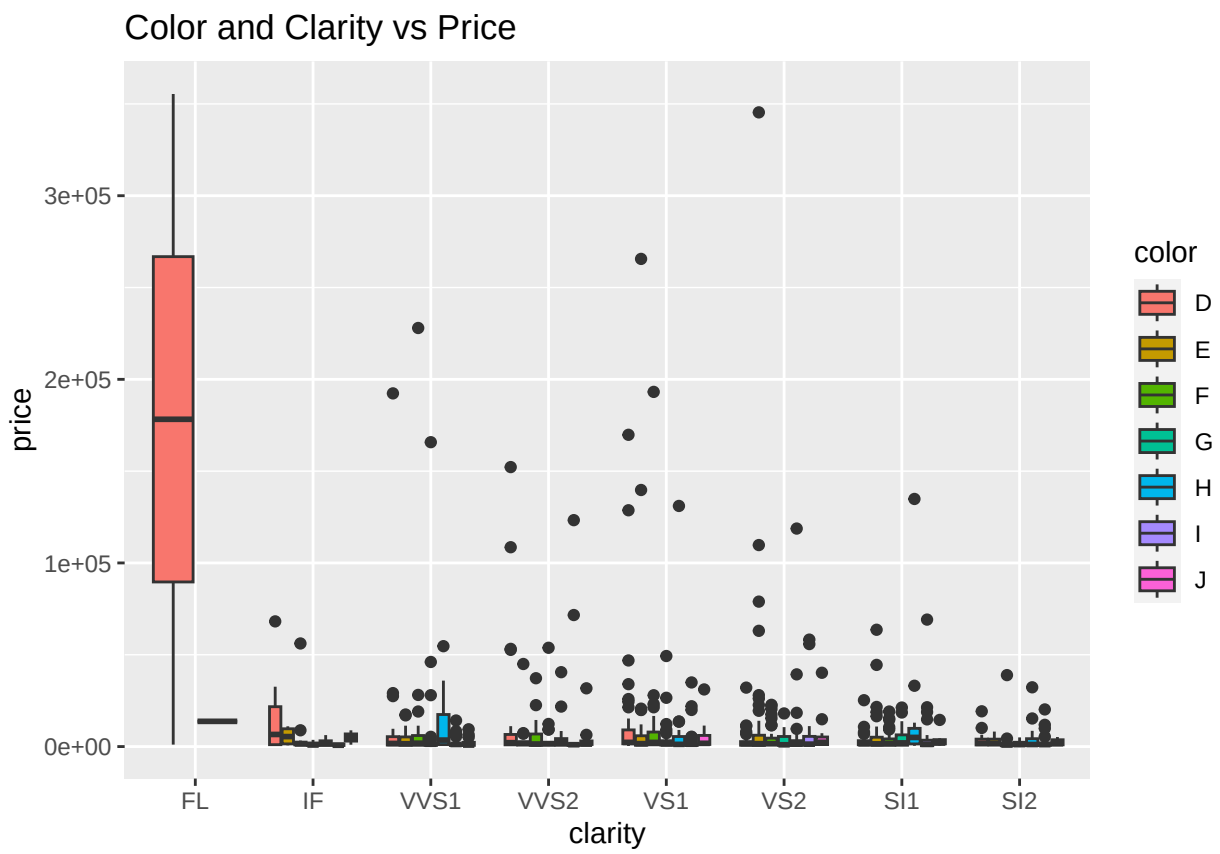

```
ggplot(diamonds, aes(x=carat, y=price, color=clarity))+
  geom_point()+
  labs(title="Carat and clarity vs Price")
```



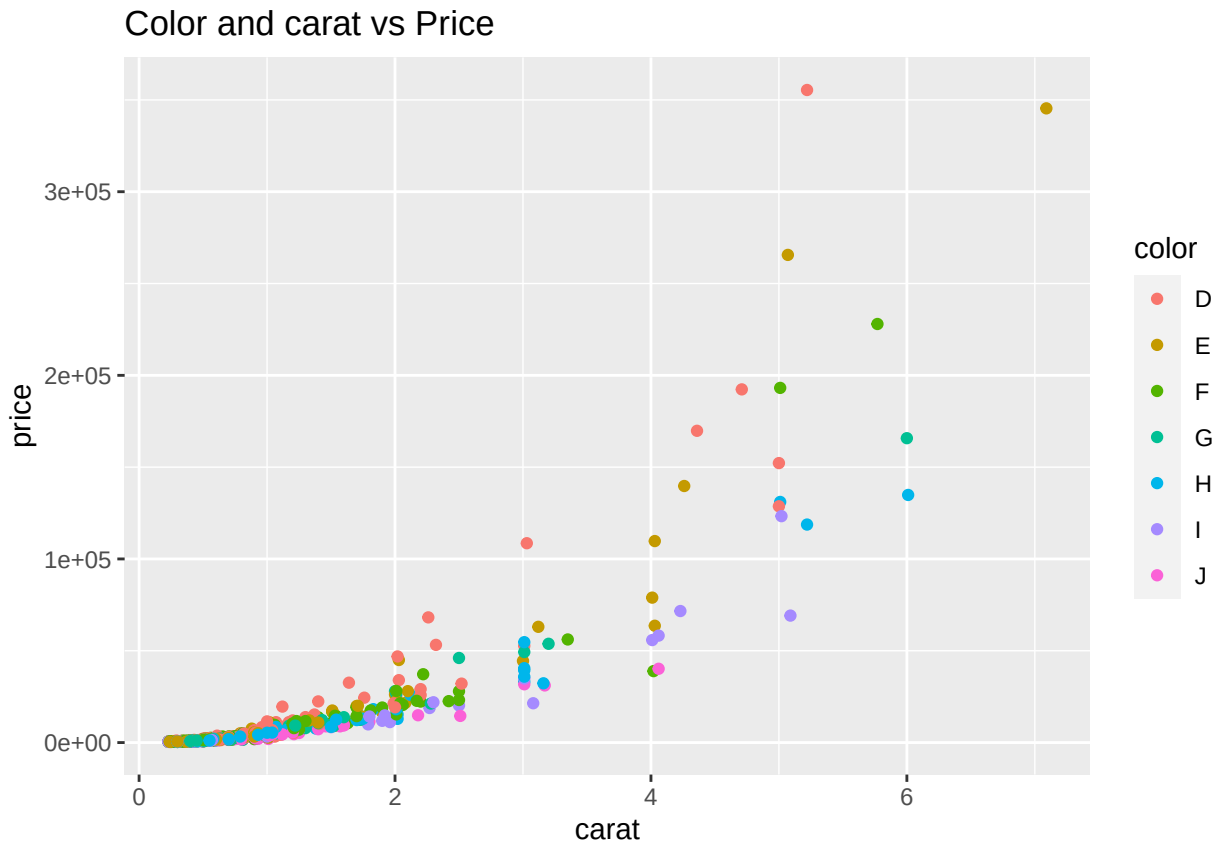

```
ggplot(diamonds, aes(x=color, y=price, fill=cut))+
  geom_boxplot() +
  labs(title="Color and cut vs Price")
```



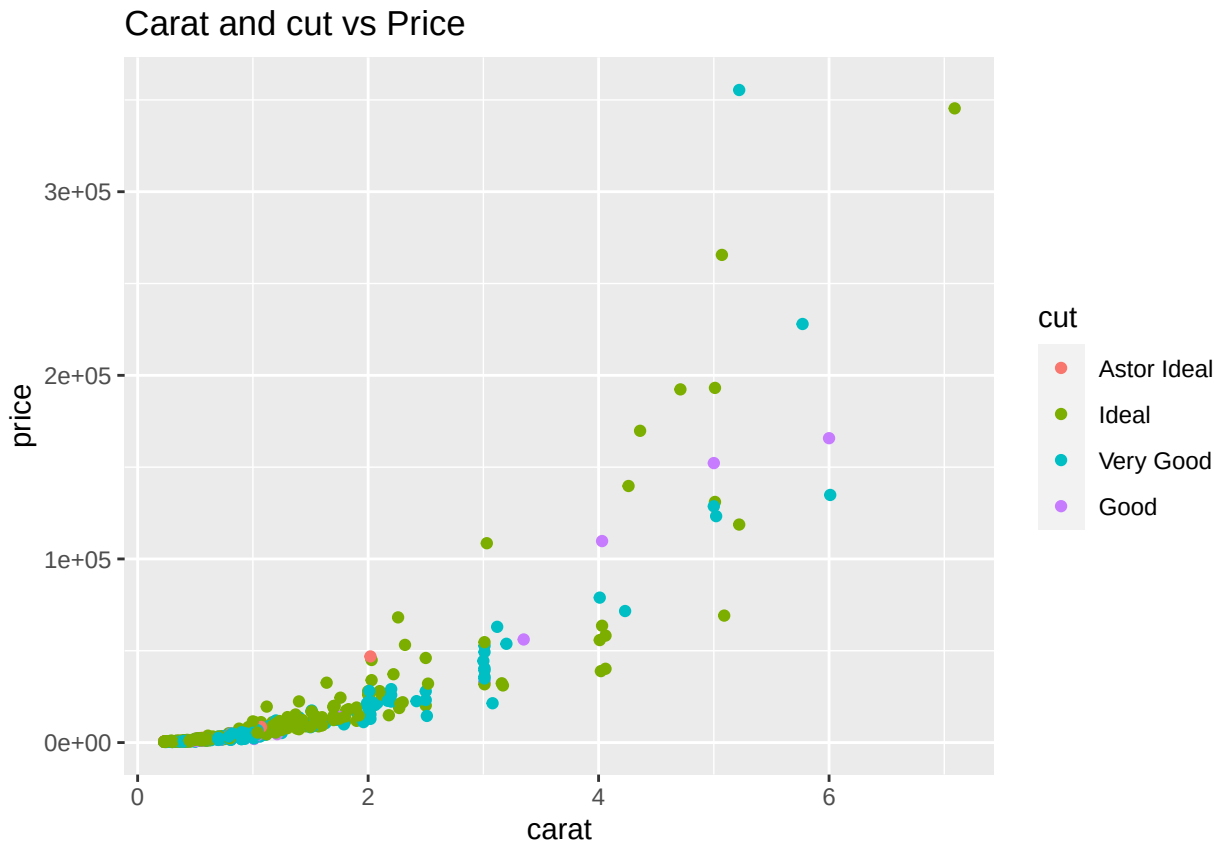

```
ggplot(diamonds, aes(x=clarity, y=price, fill=color))+
  geom_boxplot() +
  labs(title="Color and Clarity vs Price")
```



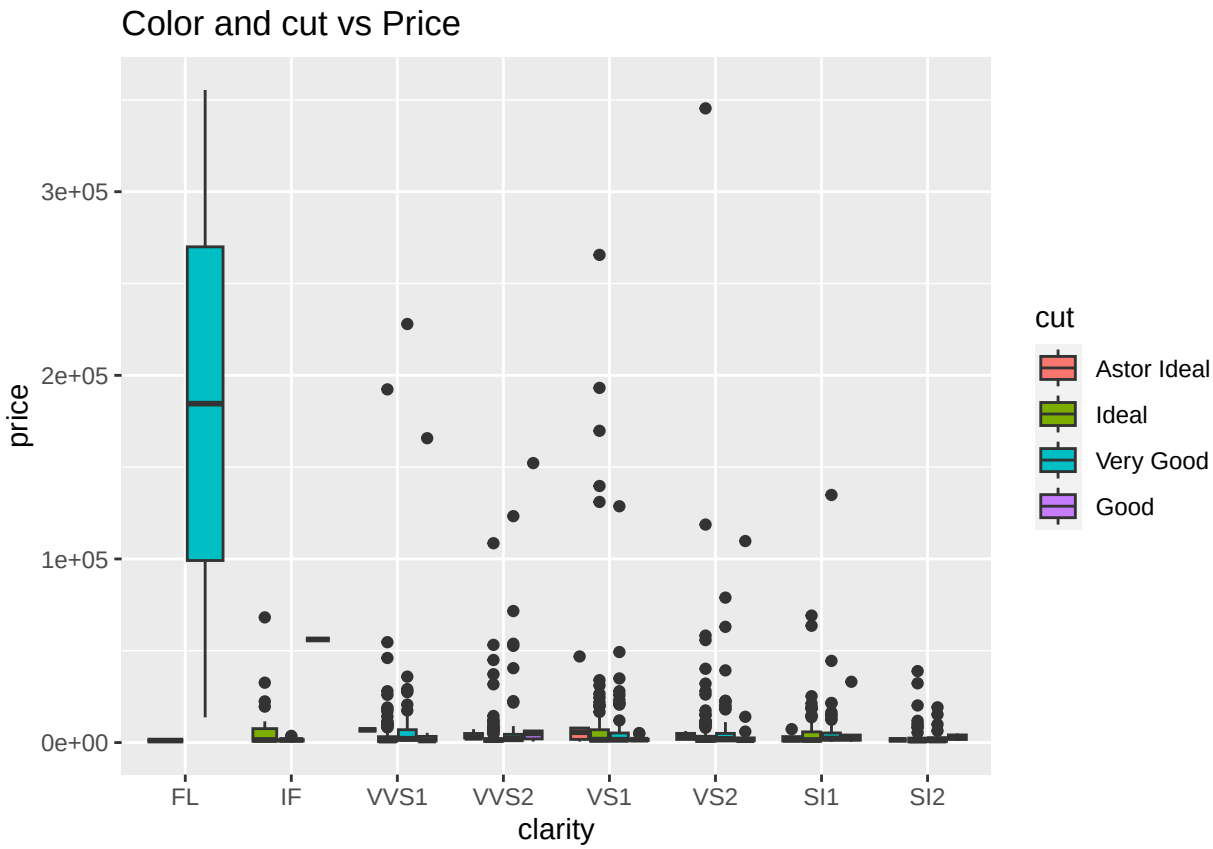

```
ggplot(diamonds, aes(x=carat, y=price, color=color))+
  geom_point() +
  labs(title="Color and carat vs Price")
```




```
ggplot(diamonds, aes(x=carat, y=price, color=cut))+
  geom_point() +
  labs(title="Carat and cut vs Price")
```

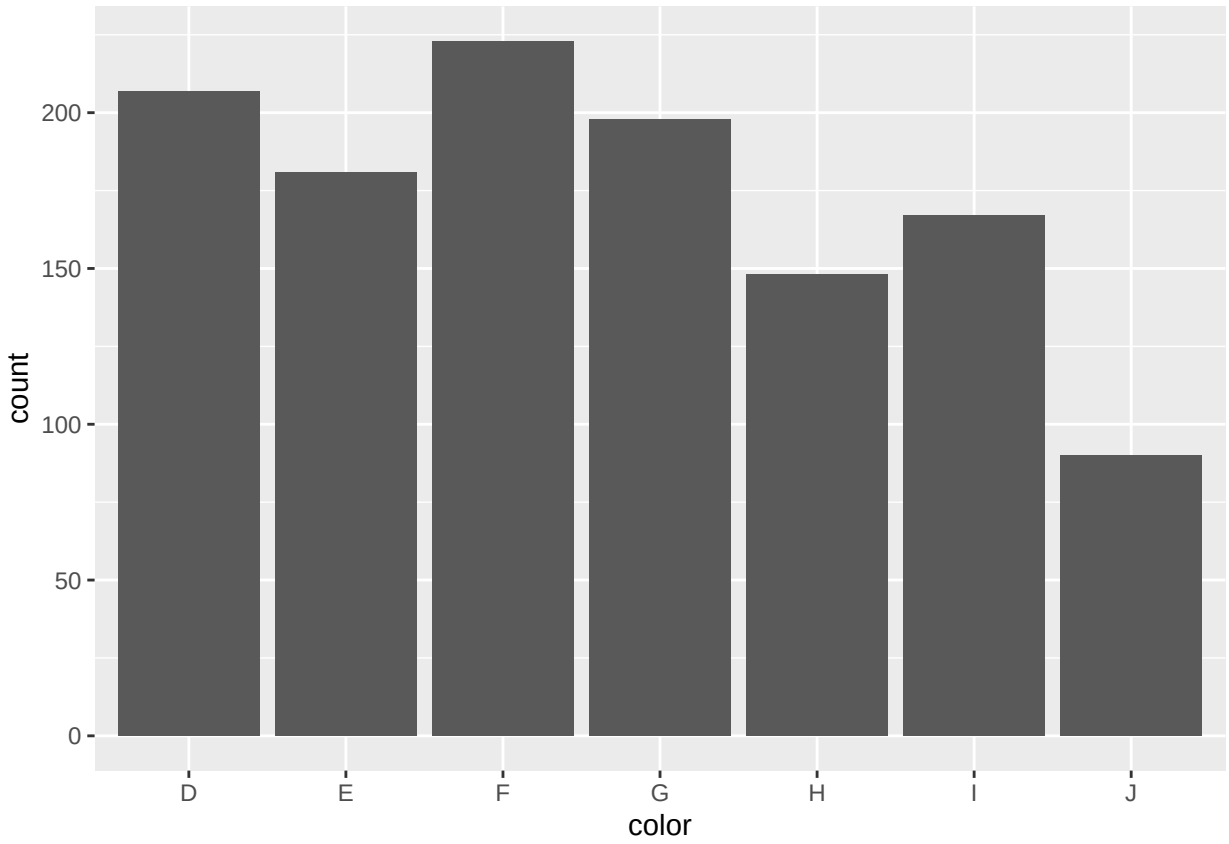



```
ggplot(diamonds, aes(x=clarity, y=price, fill=cut))+
  geom_boxplot() +
  labs(title="Color and cut vs Price")
```

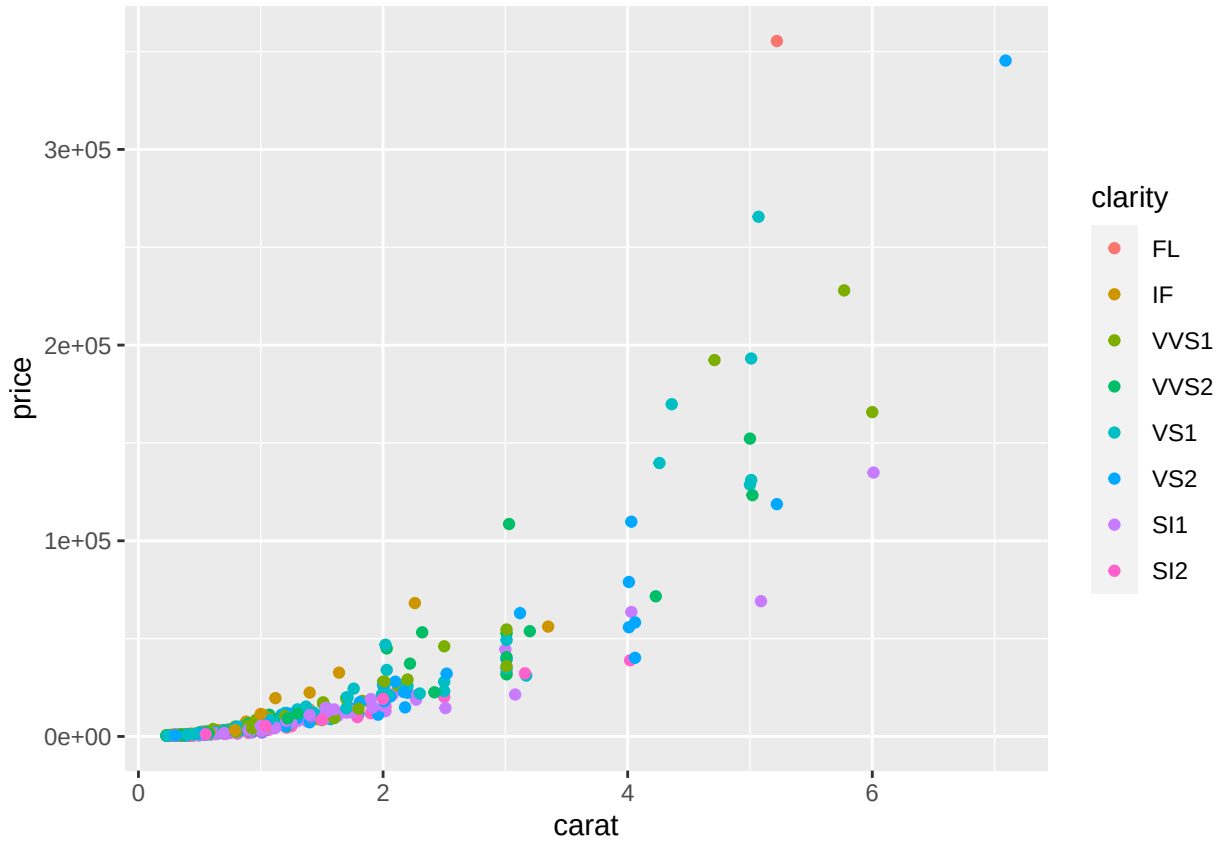



```
ggplot(diamonds,aes(x = color, fill=price))+
  geom_bar()
```

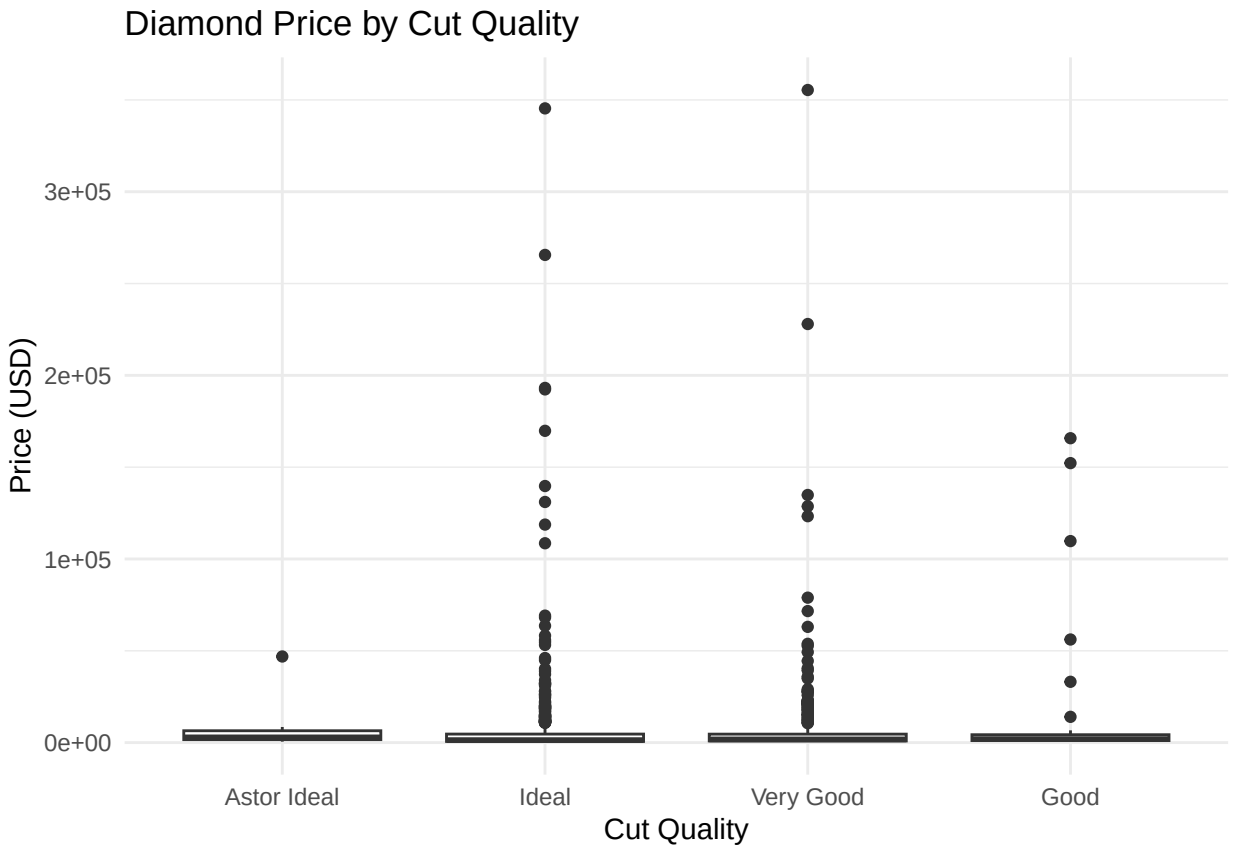
```
## Warning: The following aesthetics were dropped during statistical transformation: fill
## i This can happen when ggplot fails to infer the correct grouping structure in
##   the data.
## i Did you forget to specify a `group` aesthetic or to convert a numerical
##   variable into a factor?
```




```
ggplot(data = filter(diamonds, diamonds$clarity == "VS1")) +
  geom_point(data= diamonds, aes(x = carat, y = price, color = clarity))
```

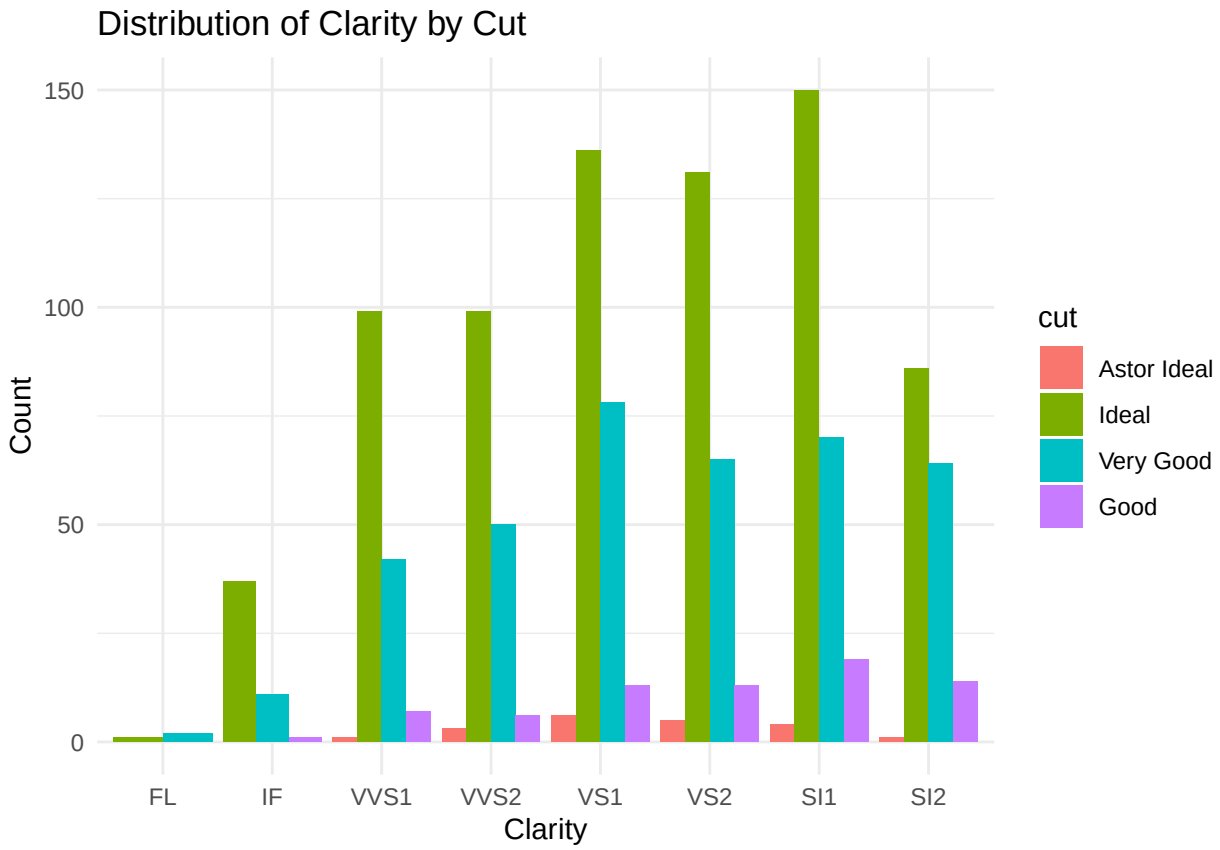



```
# Plotting the relationship between 'cut' and 'price'
ggplot(diamonds, aes(x = cut, y = price)) +
  geom_boxplot() +
  labs(title = "Diamond Price by Cut Quality",
       x = "Cut Quality",
       y = "Price (USD)") +
  theme_minimal()
```

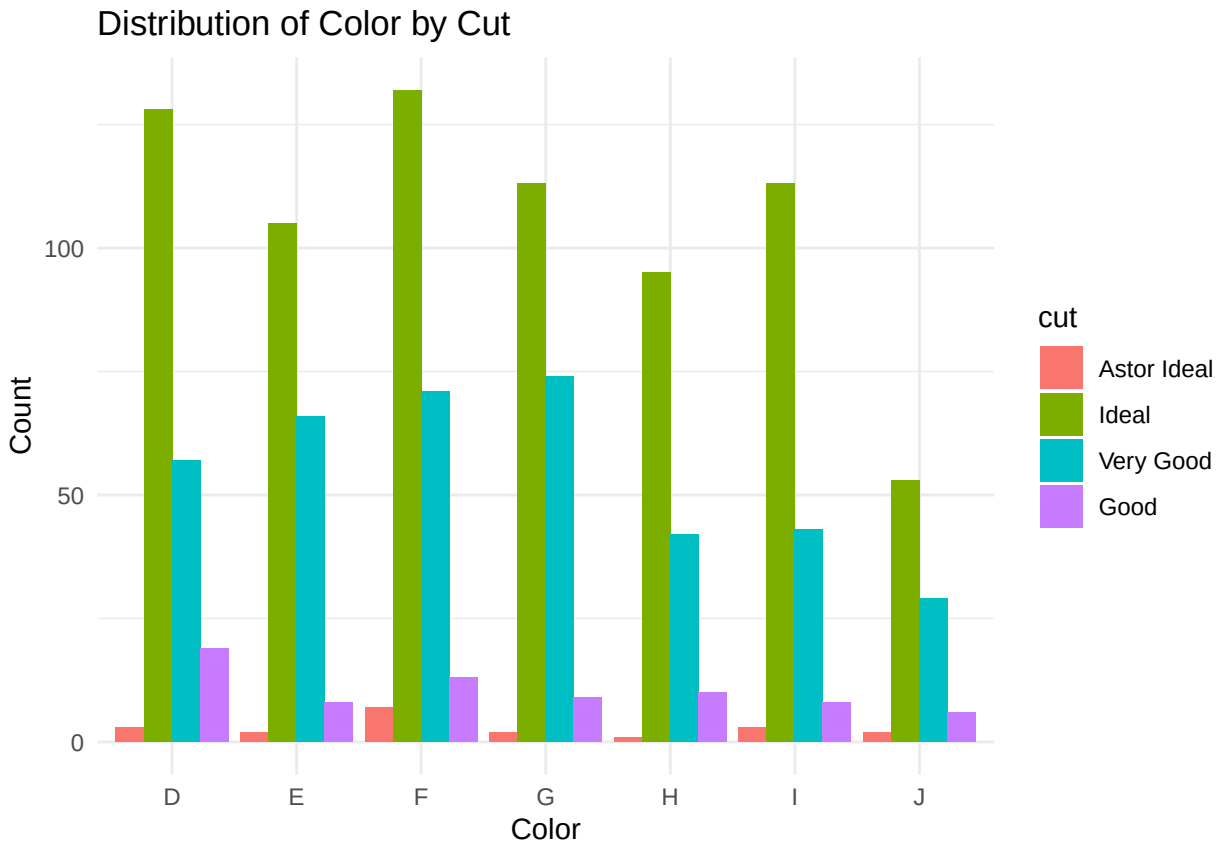


Distribution of clarity and color by cut The following plot shows the distribution of clarity and color by cut.

```
# Plotting the distribution of clarity and color by cut
ggplot(diamonds, aes(x = clarity, fill = cut)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Clarity by Cut", x = "Clarity", y = "Count") +
  theme_minimal()
```



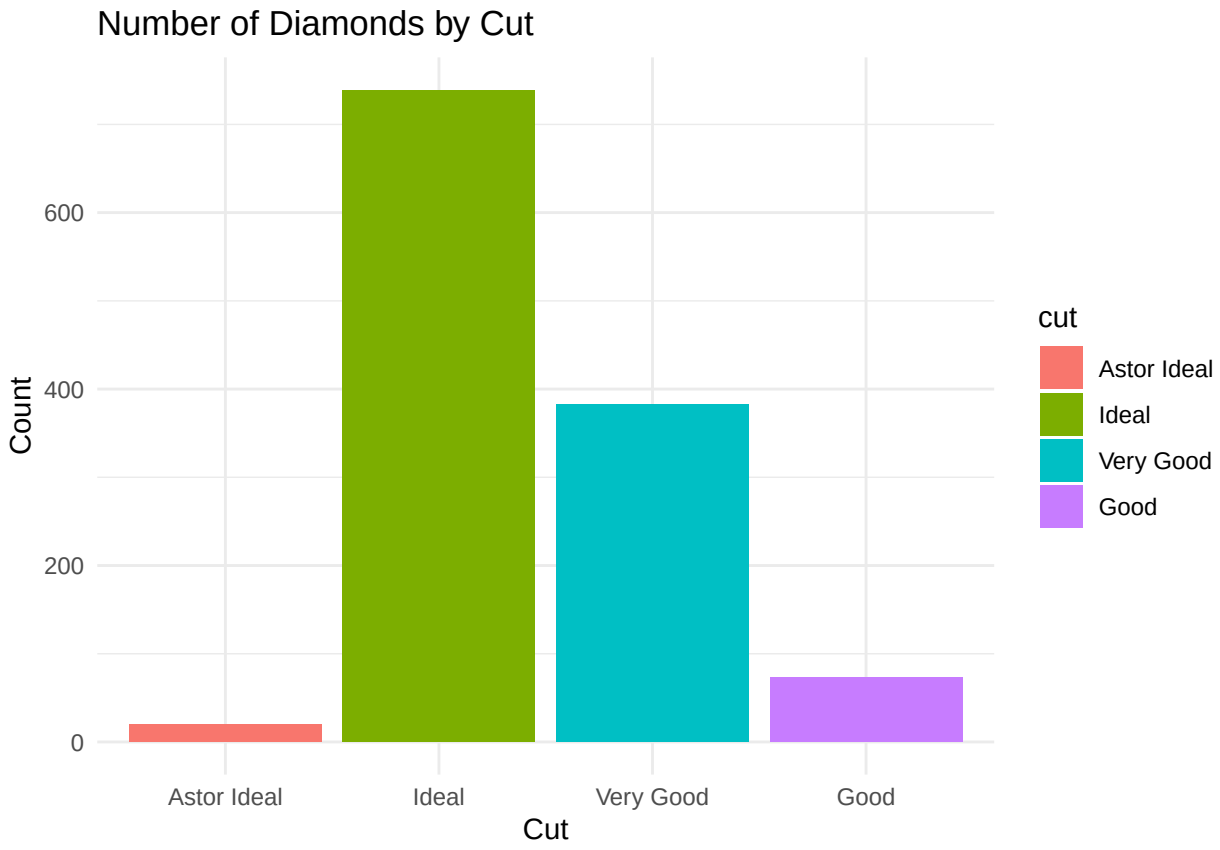

```r
# Plotting the distribution of clarity and color by cut
ggplot(diamonds, aes(x = color, fill = cut)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Color by Cut", x = "Color", y = "Count") +
  theme_minimal()
```



Number of diamonds by cut

The following plot shows the number of diamonds by cut.

```
# Plotting the number of diamonds by cut
diamonds %>%
  count(cut) %>%
  ggplot(aes(x = cut, y = n, fill = cut)) +
  geom_bar(stat = "identity") +
  labs(title = "Number of Diamonds by Cut", x = "Cut", y = "Count") +
  theme_minimal()
```

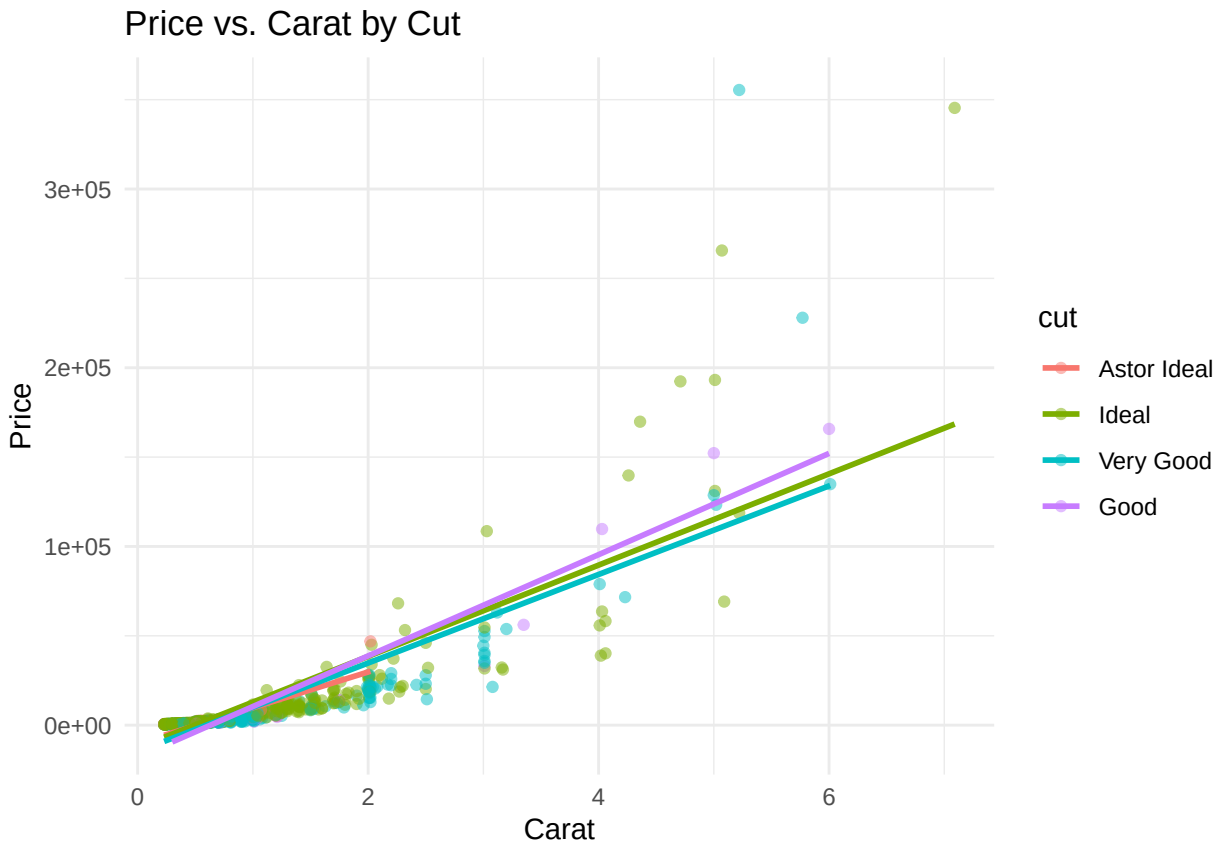


Relationship between price and carat by cut

The following plot shows the relationship between price and carat by cut.

```
# Plotting the relationship between price and carat by cut
ggplot(diamonds, aes(x = carat, y = price, color = cut)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = lm, se = FALSE) +
  labs(title = "Price vs. Carat by Cut", x = "Carat", y = "Price") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Correlation test between carat and price for each cut category

The following table shows the correlation test between carat and price for each cut category.

```
# Correlation test between carat and price for each cut category
diamonds %>%
  group_by(cut) %>%
  summarise(correlation = cor(price, carat))
```

```
## # A tibble: 4 x 2
##   cut         correlation
##   <fct>             <dbl>
## 1 Astor Ideal       0.834
## 2 Ideal             0.828
## 3 Very Good         0.797
## 4 Good              0.958
```

```
# Fit separate linear regression models for each cut category
models_by_cut <- diamonds %>%
  group_by(cut) %>%
  do(model = lm(price ~ carat, data = .))
```

```
# Fit separate linear regression models for each cut category
models_by_cut <- diamonds %>%
  group_by(cut) %>%
  do(tidy(lm(price ~ carat, data = .)))

# View the results
```

```r
print(models_by_cut)
```

```
## # A tibble: 8 x 6
## # Groups:   cut [4]
##   cut         term        estimate std.error statistic   p.value
##   <fct>       <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 Astor Ideal (Intercept)  -10258.     2816.     -3.64 1.86e-  3
## 2 Astor Ideal carat         20000.     3120.      6.41 4.93e-  6
## 3 Ideal       (Intercept)  -12634.      672.    -18.8  8.86e- 65
## 4 Ideal       carat         25545.      637.     40.1  1.40e-187
## 5 Very Good   (Intercept)  -14795.     1184.    -12.5  3.08e- 30
## 6 Very Good   carat         24779.      962.     25.8  2.37e- 85
## 7 Good        (Intercept)  -17902.     1389.    -12.9  2.85e- 20
## 8 Good        carat         28328.     1010.     28.0  3.78e- 40
```

Fit a model with interaction between carat and cut

The following table shows the results of fitting a model with interaction between carat and cut.

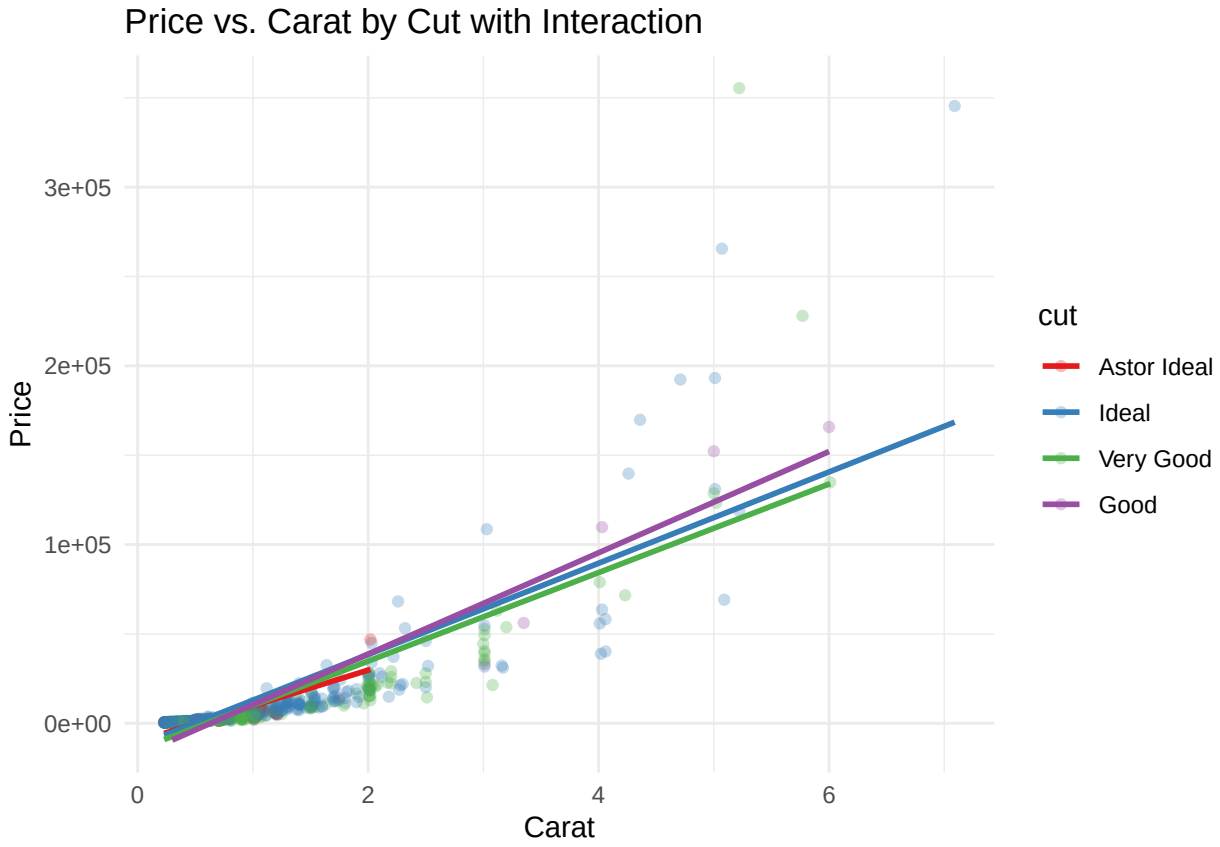
```r
# Fit a model with interaction between carat and cut
interaction_model <- lm(price ~ carat * cut, data = diamonds)

# Summary of the interaction model
summary(interaction_model)
```

```
##
## Call:
## lm(formula = price ~ carat * cut, data = diamonds)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -51142  -4499   1207   4775 240851
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          -10258       6696  -1.532  0.12581
## carat                 20000       7420   2.695  0.00713 **
## cutIdeal              -2375       6734  -0.353  0.72432
## cutVery Good          -4537       6775  -0.670  0.50315
## cutGood               -7644       7055  -1.083  0.27885
## carat:cutIdeal         5545       7450   0.744  0.45683
## carat:cutVery Good     4780       7467   0.640  0.52223
## carat:cutGood          8328       7594   1.097  0.27302
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13500 on 1206 degrees of freedom
## Multiple R-squared:  0.6884, Adjusted R-squared:  0.6866
## F-statistic: 380.7 on 7 and 1206 DF,  p-value: < 2.2e-16
```

```r
# Visual comparison of regression lines
ggplot(diamonds, aes(x = carat, y = price, color = cut)) +
  geom_point(alpha = 0.3) +
  geom_smooth(method = "lm", aes(group = cut), se = FALSE) +
  labs(title = "Price vs. Carat by Cut with Interaction", x = "Carat", y = "Price") +
```

```
  theme_minimal() +
  scale_color_brewer(palette = "Set1")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Calculate predictive performance metrics

The following table shows the predictive performance metrics for the interaction model.

```
# Calculate predictive performance metrics
# install.packages("Metrics")
library(Metrics)
predicted <- predict(interaction_model, diamonds)
actual <- diamonds$price

rmse_value <- rmse(actual, predicted)
mae_value <- mae(actual, predicted)

print(paste("RMSE:", rmse_value))
```

```
## [1] "RMSE: 13459.2848948903"
```

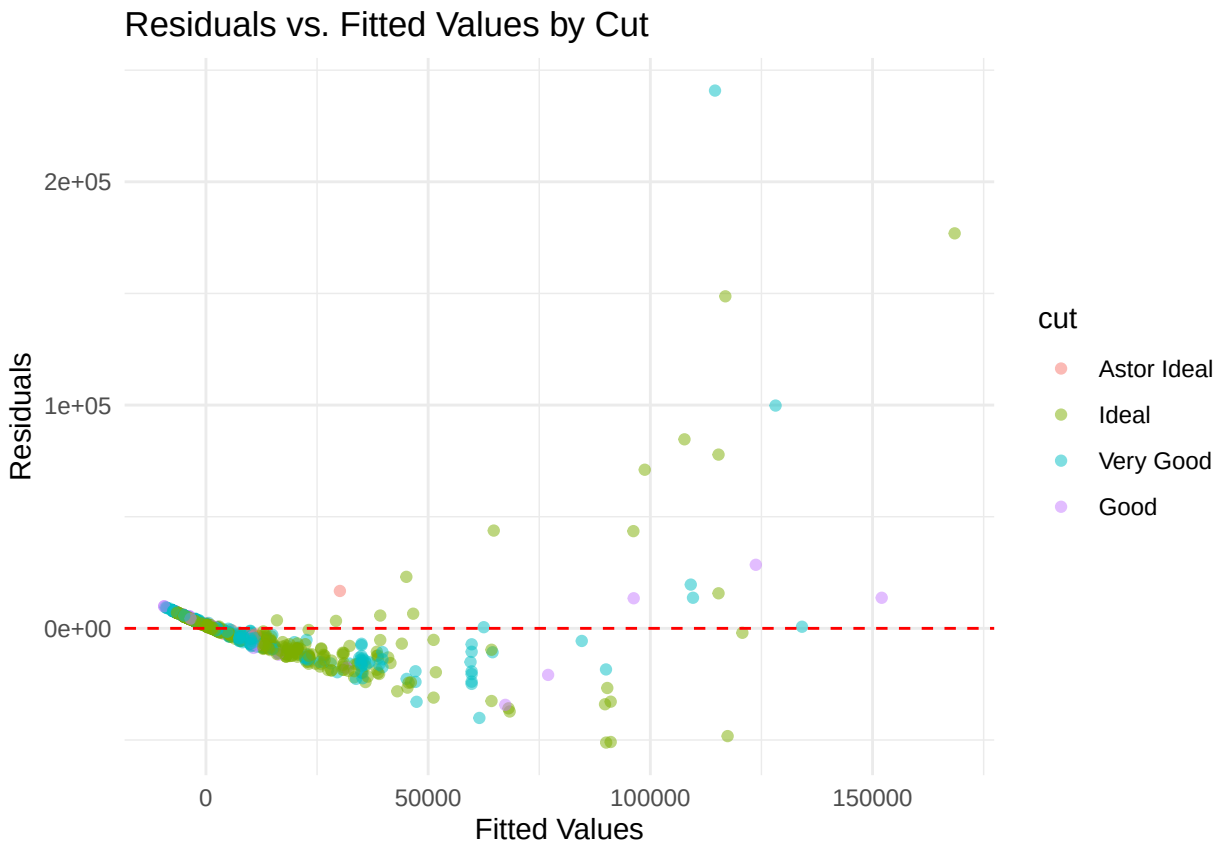
```
print(paste("MAE:", mae_value))
```

```
## [1] "MAE: 6404.98165326948"
```

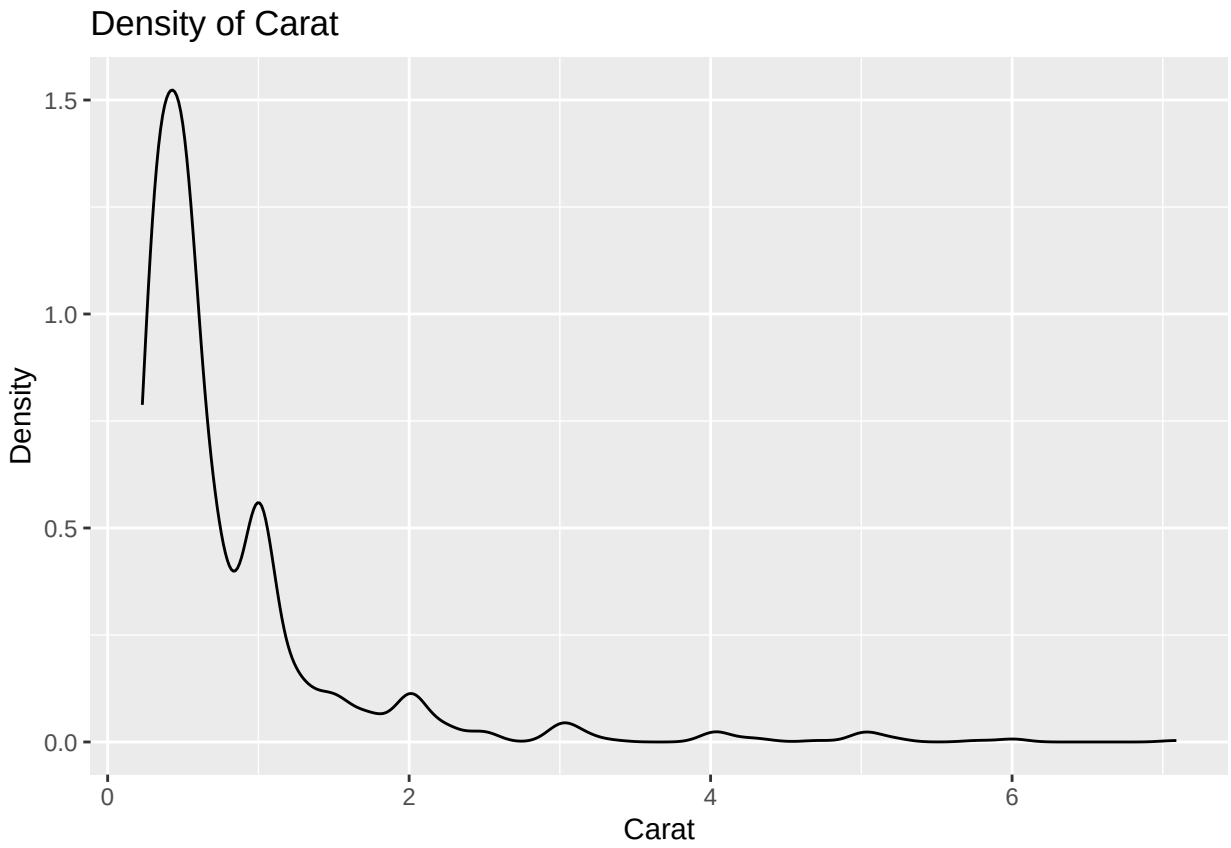
Visualizing residuals for the interaction model

The following plot shows the residuals for the interaction model.

```r
# Visualizing residuals for the interaction model
residuals_df <- data.frame(residuals = resid(interaction_model), fitted = fitted(interaction_model), cu
ggplot(residuals_df, aes(x = fitted, y = residuals, color = cut)) +
  geom_point(alpha = 0.5) +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Residuals vs. Fitted Values by Cut", x = "Fitted Values", y = "Residuals") +
  theme_minimal()
```

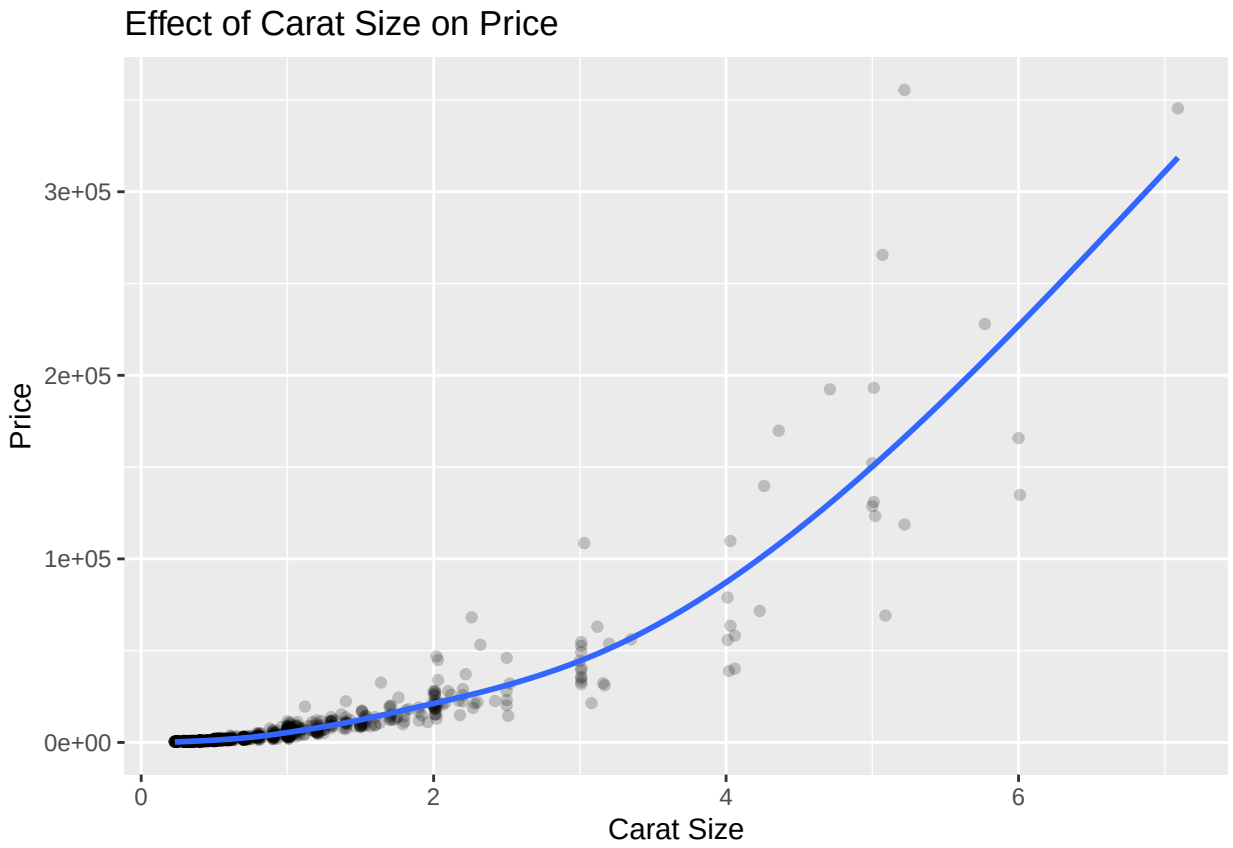



```r
ggplot(diamonds,aes(x=carat))+ geom_density()+ labs(x="Carat",y="Density",title="Density of Carat")
```

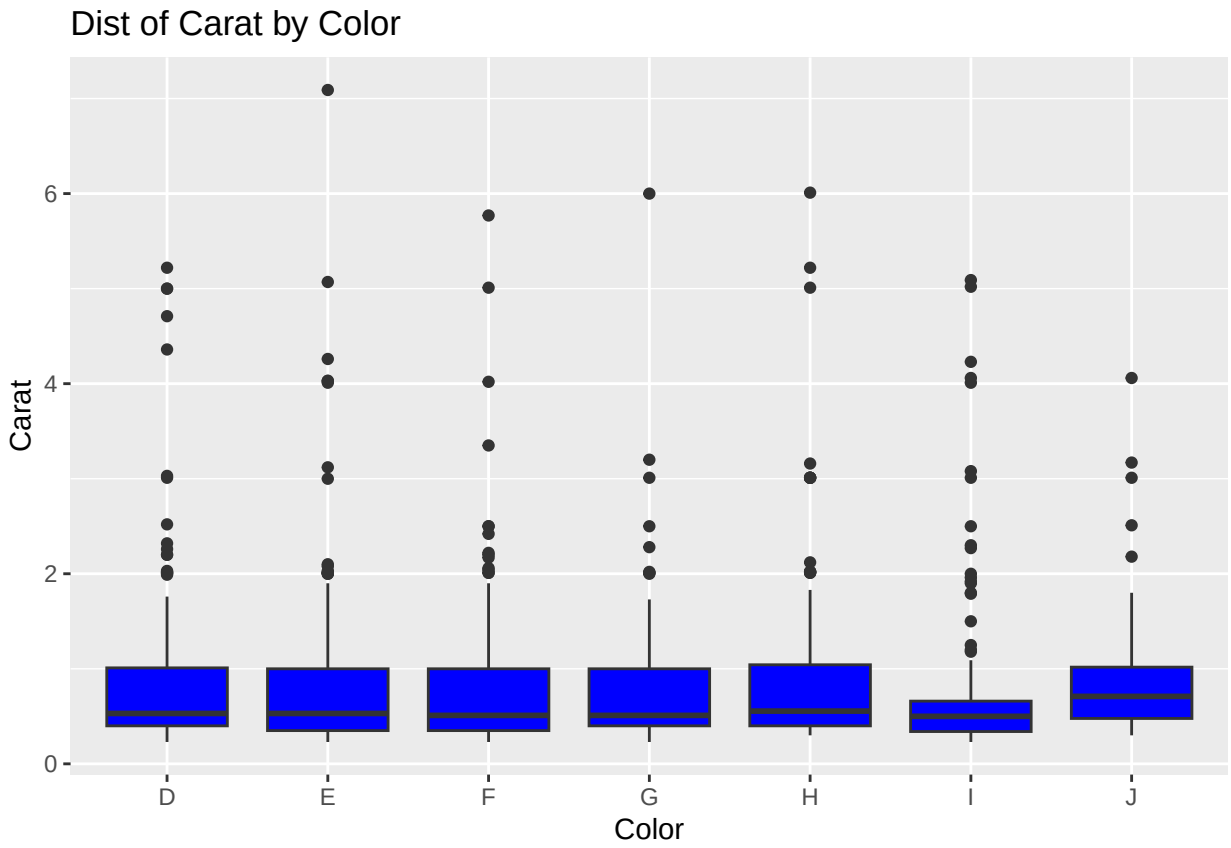



```
ggplot2::ggplot(diamonds, aes(x=carat,y=price))+
  geom_point(alpha=0.2)+
  geom_smooth(se=FALSE)+
  labs(x="Carat Size", y="Price", title="Effect of Carat Size on Price")
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```

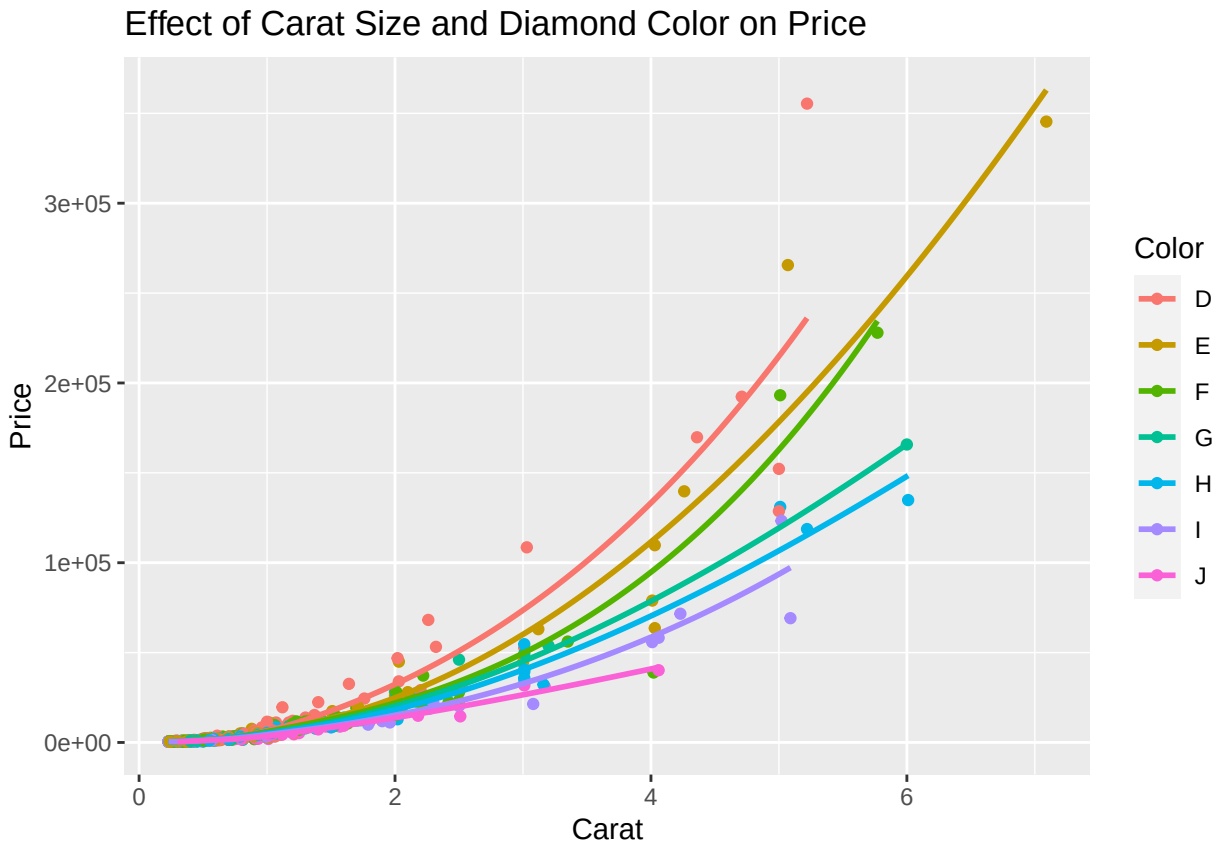



```
ggplot(diamonds, aes(x=color, y=carat))+
  geom_boxplot(fill="Blue")+
  labs(x="Color", y="Carat", title="Dist of Carat by Color")
```

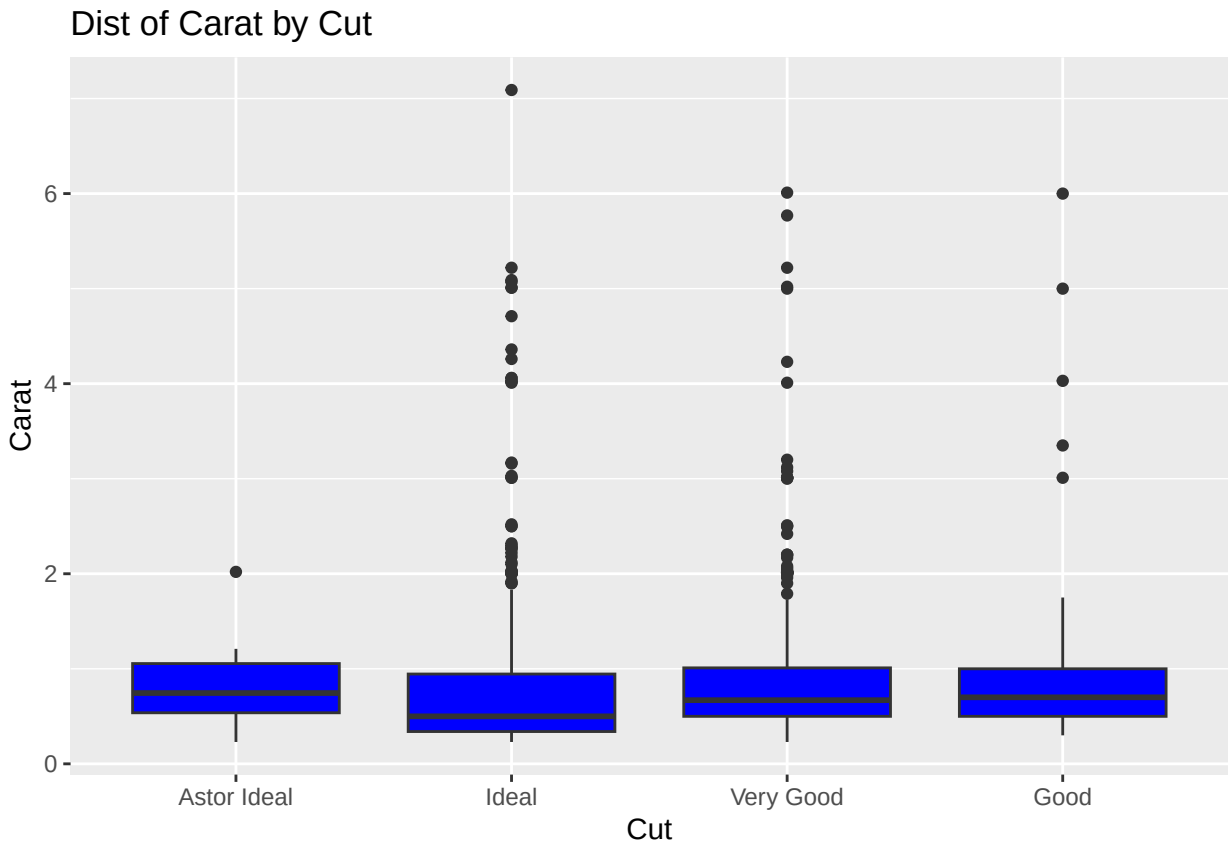



```
ggplot(diamonds, aes(x=carat, y=price, color=color)) +
geom_point() +
geom_smooth(se=FALSE)+
labs(x="Carat", y="Price", title="Effect of Carat Size and Diamond Color on Price", color = "Color")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

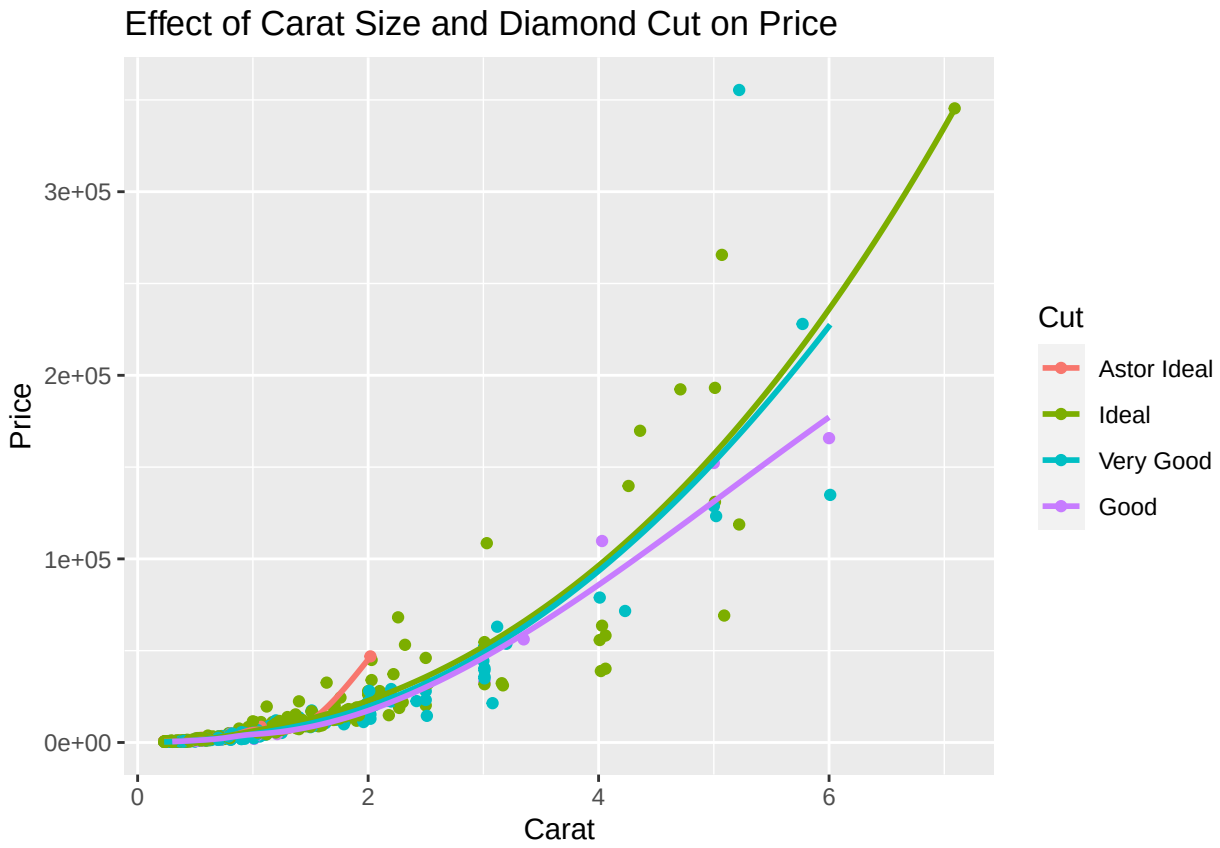



```
ggplot(diamonds, aes(x=cut, y=carat))+
  geom_boxplot(fill="Blue")+
  labs(x="Cut", y="Carat", title="Dist of Carat by Cut")
```

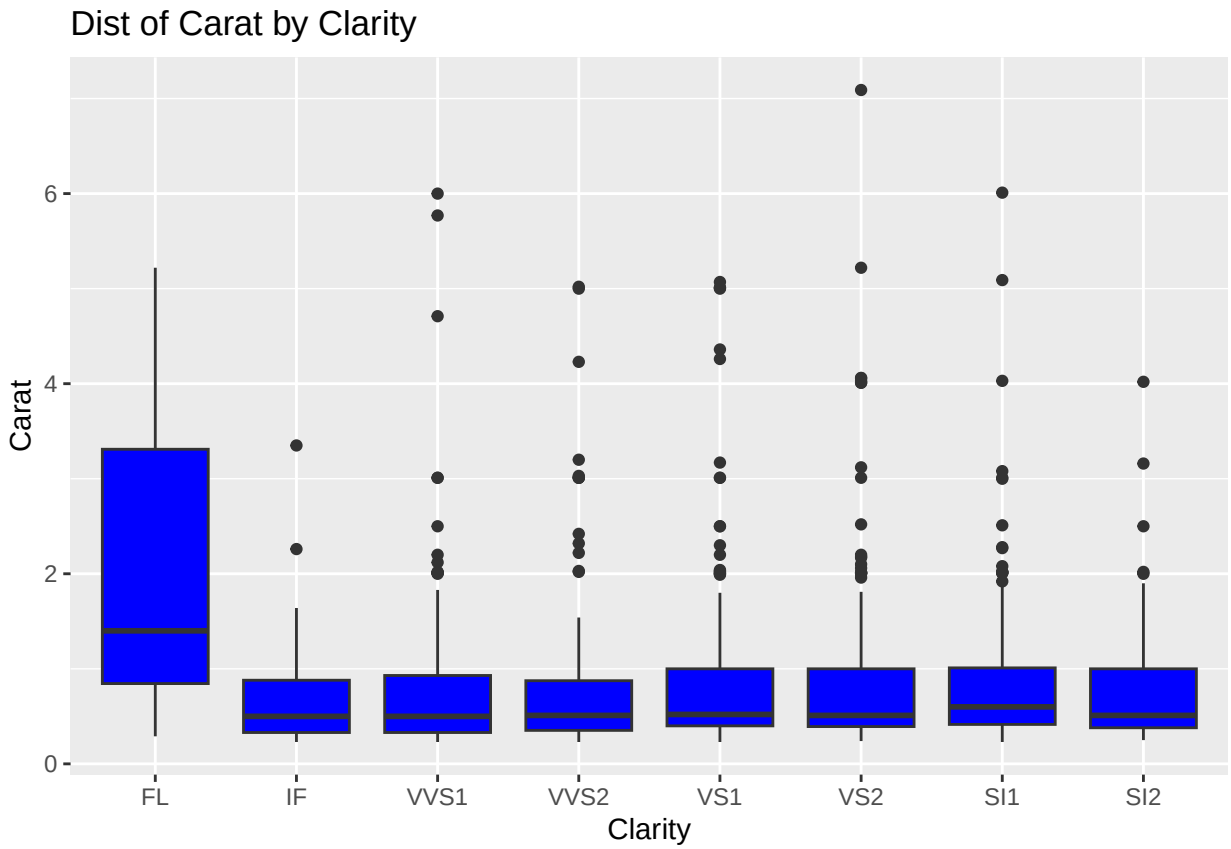



```
ggplot(diamonds, aes(x=carat, y=price, color=cut)) +
geom_point() +
geom_smooth(se=FALSE)+
labs(x="Carat", y="Price", title="Effect of Carat Size and Diamond Cut on Price", color = "Cut")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

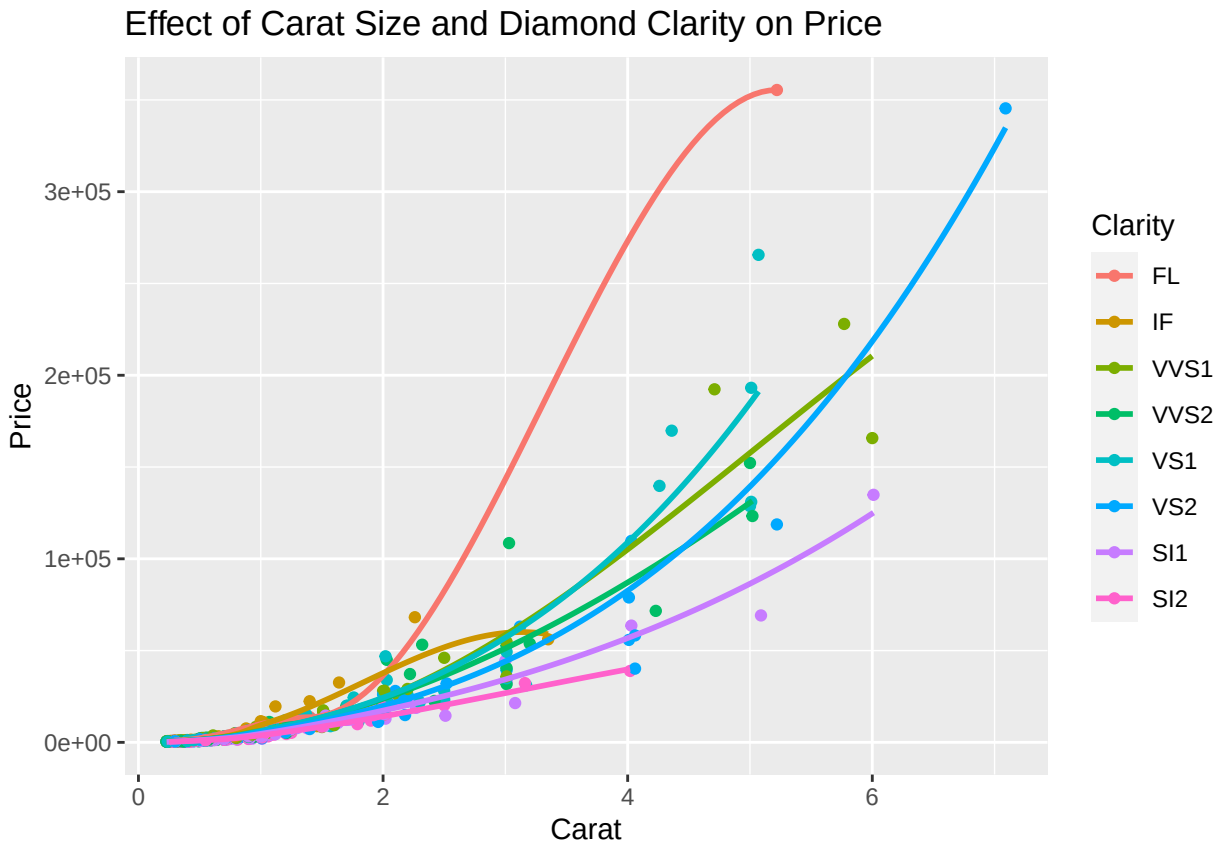



```
ggplot(diamonds, aes(x=clarity, y=carat))+
  geom_boxplot(fill="Blue")+
  labs(x="Clarity", y="Carat", title="Dist of Carat by Clarity")
```




```
ggplot(diamonds, aes(x=carat, y=price, color=clarity)) +
geom_point() +
geom_smooth(se=FALSE)+
labs(x="Carat", y="Price", title="Effect of Carat Size and Diamond Clarity on Price", color = "Clarity")
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : span too small.  fewer data values than degrees of freedom.

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : pseudoinverse used at 0.26535

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : neighborhood radius 1.1346

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : reciprocal condition number 0

## Warning in simpleLoess(y, x, w, span, degree = degree, parametric = parametric,
## : There are other near singularities as well. 14.781
```


Effect of Carat Size and Diamond Clarity on Price
Price
3e+05
2e+05
1e+05
0e+00
0
2
4
6
Carat
Clarity
FL
IF
VVS1
VVS2
VS1
VS2
SI1
SI2