0.1 Overplotting

Important Point (X): Lorem Ipsum

Plotting all the points in this df (there are around 50,000!) takes a little bit of time and causes the plot to become saturated so that we can't see individual observations. A 2D histogram is a type of heatmap, where count is mapped to color, you could also have used a mark that maps size to color, which might even be more effective but that is not as commonly seen. Here we can clearer see that a small area is much more dense than the others, although they looked similar in the saturated plot. How can we zoom into this area?

```
alt.Chart(diamonds).mark_rect().encode(
    alt.X('carat').bin(maxbins=40),
    alt.Y('price').bin(maxbins=40),
    alt.Color('count()')
)
```

Instead of squares,**hexagonal bins** can be used. These have theoretically superior qualities over squares, such as a more natural notation of neighbors.
2 dimensional **KDEs** in ggplot. This works just like 1D KDEs, except that the kernel on each data point extends in 2 dimensions
Use **ridges contours**, similar to a topographic map. These contour plots are often less intuitive than the density plot above, so the recommendation is to use the density plot instead.

```
ggplot(diamonds) +
    aes(x = carat,
        y = price) +
    geom_bin2d() or geom_hex() or geom_density_2d_filled() or
         geom_density_2d()
```

0.2 Axis label formatting

Scientific Notation & Grid Tick Modifications
Scientific notation can be useful for internal analysis or when displaying very large or very small numbers. However, it may be confusing for a general audience. Consider formatting axis labels using plain numbers or appropriate unit prefixes (e.g., thousands, millions, micro, milli) to improve readability.

The number of grid ticks can be modified using `tickCount`. Note that `tickCount` cannot be applied to binned data, so the plot has been adjusted here to demonstrate its effect.

```
.axis(format='e') # For e like notation
.axis(format='s') # Standard international (SI) units are often
     easier to digest.
.axis(format='~s') # A prefaced ~ removes trailing zeros.
.axis(format='\$~s') # Formaters can also be combined.
.alt.X('carat').axis(None),
.alt.Y('price').axis(tickCount=2)
alt.themes.enable('dark')
```

```
alt.Color('count()').legend(None) # remove a legend
alt.Y('price').bin(maxbins=400).scale(domain=(0, 2000),
     reverse=True), #Reversing an axis
```

```
ggplot(diamonds) +
    aes(x = carat,
        y = price) +
    geom_hex() +
    scale_y_continuous(labels = scales::label_number(scale_cut =
         scales::cut_si(''))) +
    scale_y_continuous(labels = scales::label_dollar()) +
    scale_y_continuous(labels = scales::label_dollar(scale = .001,
         suffix = "K")) +
    scale_fill_continuous(labels = scales::label_number(scale_cut =
         scales::cut_si(''))) +
    guides(fill = "none") # remove a legend
    scale_x_continuous(
        limits = c(10, 31), oob = scales::oob_keep, #scale limit
        labels = scales::label_dollar(), #label formater
        breaks = scales::pretty_breaks(n = 10) #tick count
    ) +
    scale_y_continuous(limits = c(2000, 0), trans = 'reverse')
         #reversing an axis
    scale_fill_continuous(trans = 'reverse')
```

0.3 Labels and Subtitles

```
alt.Chart(
    diamonds,
    title=alt.Title(
        text='Higher carat diamonds are more expensive',
        subtitle='But most diamonds are of low carat'
    )
).mark_rect().encode(
    alt.X('carat').bin(maxbins=40).title('Carat'),
    alt.Y('price').bin(maxbins=40).title('Price'),
    alt.Color('count()').title('Number of Records')
)
```

```
ggplot(diamonds) +
    aes(x = carat,
        y = price) +
    geom_hex() +
    labs(x = 'Carat', y = 'Price', fill = 'Number', title =
         'Diamonds') +
    scale_y_continuous(labels = scales::label_dollar())
```

Trendlines (also sometimes called "lines of best fit", or "fitted lines") are good to highlight general trends in the data that can be hard to elucidate by looking at the raw data points. This can happen if there are many data points or many groups inside the data. A moving average becomes a bit complicated because there are so many y-values for the exact same x-value, so we would need to calculate the average for each year first, and then move/roll over that, which can be done using the window_transform method in Altair If it is important that the line has values that are easy to interpret, choose a rolling mean (or maybe a mean if it is not too noisy). These are also the most straightforward trendlines when communicating data to a general audience. If you think a simple line equation (e.g. linear) describes your data well, this can be advantageous since you would know that your data follows a set pattern, and it is easy to predict how the data behaves outside the values you have collected (of course still with more uncertainty the further away from your data you predict). If you are mainly interested in highlighting a trend in the current data, and the two situations described above are not of great importance for your figure, then a loess line could be suitable. It has the advantage that it describes trends in data very "naturally", meaning that it highlights patterns we would tend to highlight ourselves in qualitative assessment. It also less strict in its statistical assumption compared to e.g. a linear regression, so you don't have to worry about finding the correct equation for the line, and assessing whether your data truly follows that equation globally. A downside is that it is hard to extrapolate this trend outside the available data.

0.4 When to choose which trendline? Reduce content

Trendlines (also sometimes called "lines of best fit", or "fitted lines") are good to highlight general trends in the data that can be hard to elucidate by looking at the raw data points. This can happen if there are many data points or many groups inside the data. A moving average becomes a bit complicated because there are so many y-values for the exact same x-value, so we would need to calculate the average for each year first, and then move/roll over that, which can be done using the window_transform method in Altair If it is important that the line has values that are easy to interpret, choose a rolling mean (or maybe a mean if it is not too noisy). These are also the most straightforward trendlines when communicating data to a general audience. If you think a simple line equation (e.g. linear) describes your data well, this can be advantageous since you would know that your data follows a set pattern, and it is easy to predict how the data behaves outside the values you have collected (of course still with more uncertainty the further away from your data you predict). If you are mainly interested in highlighting a trend in the current data, and the two situations described above are not of great importance for your figure, then a loess line could be suitable. It has the advantage that it describes trends in data very "naturally", meaning that it highlights patterns we would tend to highlight ourselves in qualitative assessment. It also less strict in its statistical assumption compared to e.g. a linear regression, so you don't have to worry about finding the correct equation for the line, and assessing whether your data truly follows that equation globally. A downside is that it is hard to extrapolate this trend outside the available data.

```
points + points.mark_line().encode(y='mean(Horsepower)')
```