Storytelling with Data! in Altair

by Maisa de Oliveira Fraiz

Introduction

This project aims to replicate the examples from Cole Nussbaumer's book, "Storytelling with Data - Let's Practice!", using `Python Altair`. Our primary objective is to document the reasoning behind the modifications proposed by the author, while also highlighting the challenges that arise when transitioning from the book's Excel-based approach to programming in a different software environment.

`Altair` was selected for this project due to its declarative syntax, interactivity, grammar of graphics, and compatibility with `Streamlit` and other web formatting tools, while within the user-friendly Python environment. Anticipated challenges include the comparatively smaller documentation and development community of Altair compared to more established libraries like `Matplotlib`, `Seaborn`, or `Plotly`. Furthermore, tasks that might appear straightforward in Excel may require multiple iterations to translate effectively into the language.

Imports

In [4]:
```
import pandas as pd
import numpy as np
import altair as alt
```

Chapter 5 - Think like a designer

"*Where do you want your audience to look?*" - Cole Nussbaumer

Exercise 4 - design in style

This exercise revolves around integrating brand design into graphs. While the author in the book focuses on creating a graph with a Coke and Light Coke theme, our approach takes it a step further. We will delve into various color inspirations using the most colorful graph thus far, which is the one from exercise 4.2.

We will separate the exercise into three categories: Accessibility, Branding, Paintings, and Nature.

In [5]:
```
table = pd.read_excel(r".\Data\4.2 EXERCISE.xlsx", usecols = [1, 2, 3], header =
table['Brands'] = table['Unnamed: 1']
table['Change'] = table['$ Vol % change']
```

```
table.drop(columns = ['Unnamed: 1', '$ Vol % change'], inplace = True)
table
```

Out[5]:

	spacing for dot plot	Brands	Change
0	0	Fran's Recipe	-0.14
1	1	Wholesome Goodness	-0.13
2	2	Lifestyle	-0.10
3	3	Coat protection	-0.09
4	4	Diet Lifestyle	-0.08
5	5	Feline Basics	-0.05
6	6	Lifestyle Plus	-0.04
7	7	Feline Freedom	-0.02
8	8	Feline Gold	0.01
9	9	Feline Platinum	0.01
10	10	Feline Instinct	0.02
11	11	Feline Pro	0.03
12	12	Farm Fresh Tasties	0.04
13	13	Feline Royal	0.05
14	14	Feline Focus	0.09
15	15	Feline Grain Free	0.09
16	16	Feline Silver	0.12
17	17	Nutri Balance	0.16
18	18	Farm Fresh Basics	0.17

The Original graph:

In [6]:

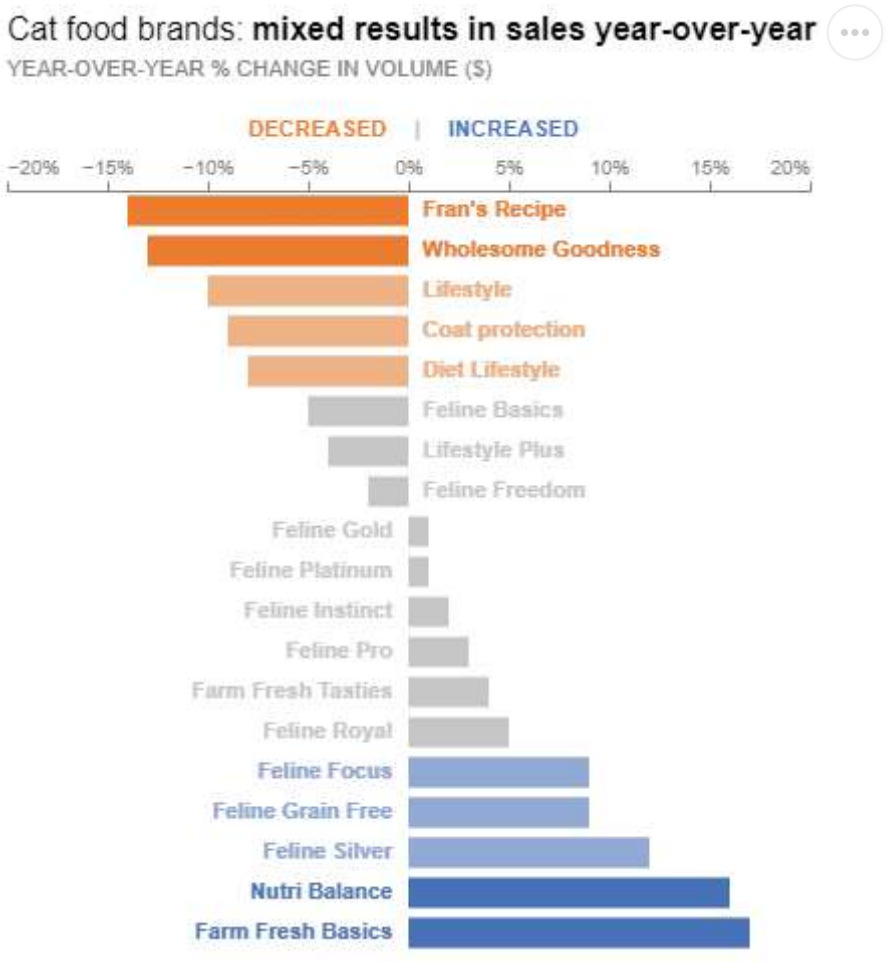
```
decreased_most = table.nsmallest(2, 'Change')
increased_most = table.nlargest(2, 'Change')

brands_decreased = decreased_most['Brands'].tolist()
brands_increased = increased_most['Brands'].tolist()

conditions_decreased = [f'datum.Brands == "{brand}"' for brand in brands_decreas
condition_decreased = f"({'|'.join(conditions_decreased)})"

conditions_increased = [f'datum.Brands == "{brand}"' for brand in brands_increas
condition_increased = f"({'|'.join(conditions_increased)})"

chart_gray = alt.Chart(
    table
    ).mark_bar(color = "#c6c6c6", size = 15).encode(
    x = alt.X(
        "Change",
```

```
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = alt.Axis(grid = False, orient = "top",
                        labelColor = "#888888", titleColor = '#888888',
                        titleFontWeight = 'normal', format = "%"),
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None)
    )

chart_oranges_mix = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = alt.Axis(grid = False, orient = "top",
                        labelColor = "#888888", titleColor = '#888888',
                        titleFontWeight = 'normal', format = "%"),
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_decreased, alt.value('#ec7c30'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_blue_mix = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = alt.Axis(grid = False, orient = "top",
                        labelColor = "#888888", titleColor = '#888888',
                        titleFontWeight = 'normal', format = "%"),
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('#4772b8'), alt.value('#91a9d5'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

label1_gray = alt.Chart(table.loc[table['Change'] < 0]).mark_text(align = 'left'
    x = alt.value(207),
    y = alt.Y('Brands', sort = None),
    text = alt.Text('Brands')
    )

label2_gray = alt.Chart(table.loc[table['Change'] > 0]).mark_text(align = 'right
    x = alt.value(192),
    y = alt.Y('Brands', sort = None),
    text = alt.Text('Brands')
    )

label_oranges = alt.Chart(table.loc[table['Change'] < 0]).mark_text(align = 'lef
    x = alt.value(207),
```

```
    y = alt.Y('Brands', sort = None),
    text = alt.Text('Brands'),
    color = alt.condition(condition_decreased,
                          alt.value('#ec7c30'),
                          alt.value('#efb284'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )


label_blue = alt.Chart(table.loc[table['Change'] > 0]).mark_text(align = 'right'
    x = alt.value(192),
    y = alt.Y('Brands', sort = None),
    text = alt.Text('Brands'),
    color = alt.condition(condition_increased,
                          alt.value('#4772b8'),
                          alt.value('#91a9d5'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

title_bw = alt.Chart(
    {"values": [{"text":  ["Cat food brands:"]}]}
    ).mark_text(
        size = 16, align = "left", dx = -200, dy = -270, fontWeight = 'normal',
        ).encode(
            text = "text:N"
            )

title_bw_bold = alt.Chart(
    {"values": [{"text":  [
        'Lifestyle line brands decline'
         ]}]}
).mark_text(size = 16, align = "left", dx = -78, dy = -270, fontWeight = 700, cc
    text = "text:N"
)

title_bw_bold_2 = alt.Chart(
    {"values": [{"text":  [
        'mixed results in sales year-over-year'
         ]}]}
).mark_text(size = 16, align = "left", dx = -78, dy = -270, fontWeight = 700, cc
    text = "text:N"
)

subtitle_bw = alt.Chart(
    {"values": [{"text":  [
        "YEAR-OVER-YEAR % CHANGE IN VOLUME ($)"
         ]}]}
).mark_text(size = 11, align = "left", dx = -200, dy = -250, fontWeight = 'norma
    text = "text:N"
)

decreased_orange = alt.Chart(
    {"values": [{"text":  [
        'DECREASED'
         ]}]}
).mark_text(size = 11, align = "left", dx = -80, dy = -220, fontWeight = 700, cc
```

```
    text = "text:N"
)

increased_blue = alt.Chart(
    {"values": [{"text":  [
            'INCREASED'
        ]}]}
).mark_text(size = 11, align = "left", dx = 20, dy = -220, fontWeight = 700, col
    text = "text:N"
)

separation =  alt.Chart(
    {"values": [{"text":  [
            '|'
        ]}]}
).mark_text(size = 11, align = "left", dx = 3, dy = -220, fontWeight = 700, colo
    text = "text:N"
)

original = (chart_gray + chart_oranges_mix +
          chart_blue_mix + label1_gray +
          label2_gray + label_oranges + label_blue +
          title_bw + title_bw_bold_2 + subtitle_bw +
          decreased_orange + increased_blue + separation)

original.properties(width = 400).configure_view(stroke = None)
```

Out[6]:



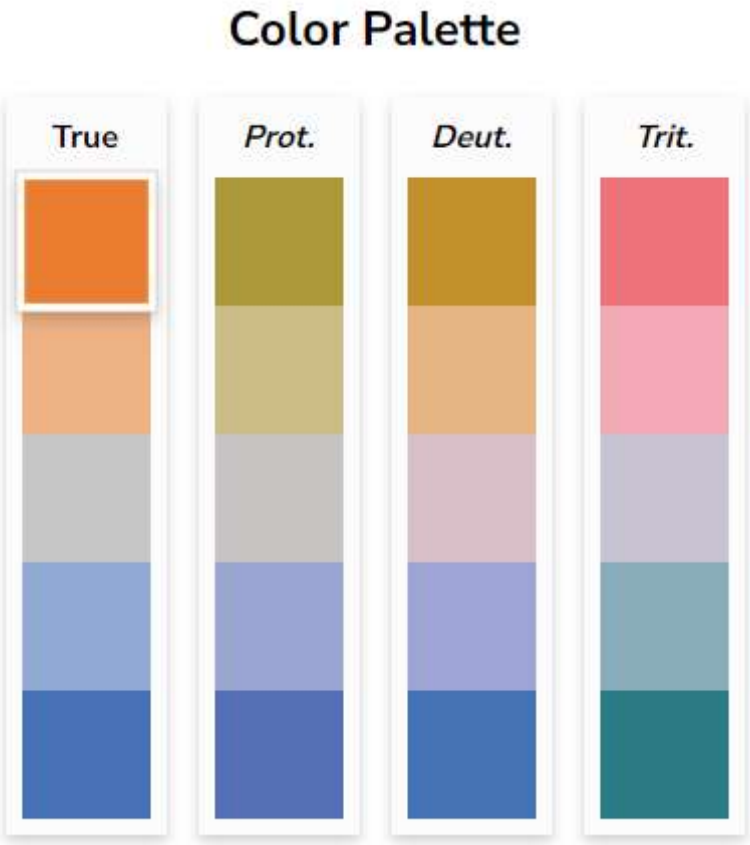
Accessibility

How do we assess the accessibility of the graph palette?

Colorblind

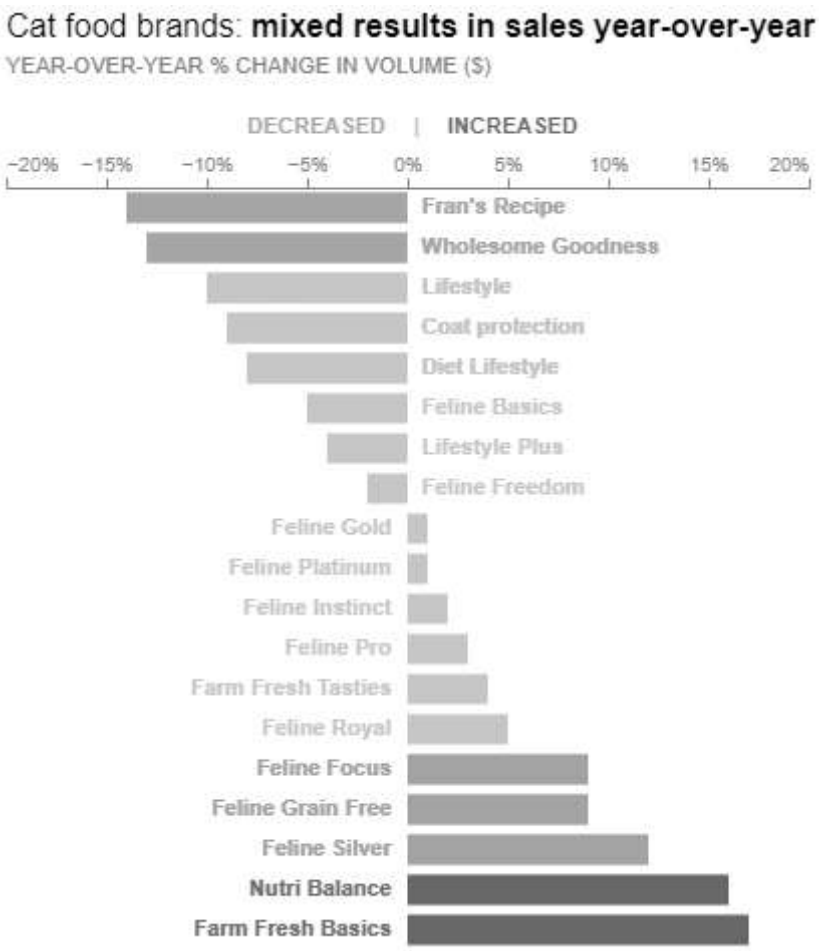
First, we can start by checking if the original colors are accessible to people with color blindness. The online tool Coloring for Colorblindness helps simulate how your selected color palette appears to viewers with protanopia, deuteranopia (both being the inability to tell the difference between red and green), and tritanopia (inability to tell the difference between blue and green, purple and red, and yellow and pink), respectively.

The image below shows that our initial palette can be distinguished by those with these visual deficiency. The website also offers some famous colorblind friendly palette, such as the Wong palette.



Black and White

Although the complete inability to distinguish colors is very rare, having a black and white version of your graph can be beneficial in certain situations, such as when intending to print it in a newspaper or an article. By using the Microsoft Photos App, we can preview how our visualization would appear without colors.



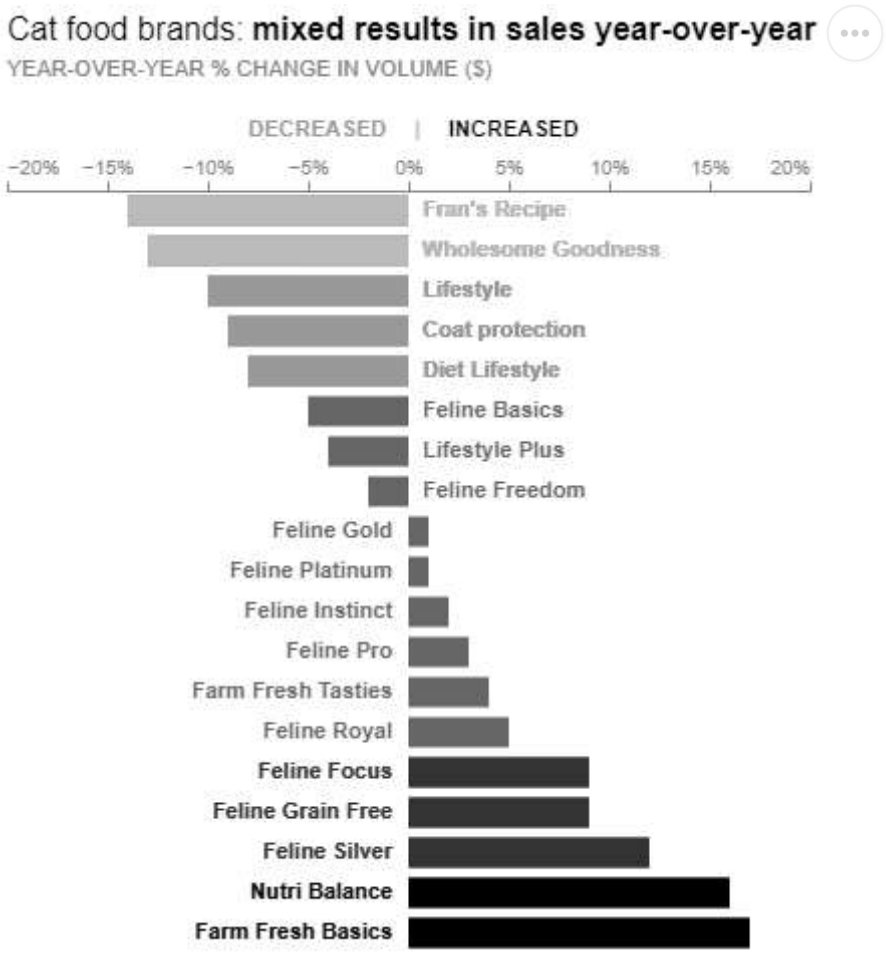
Since the top and bottom colors can not be distinguished easily, we can create a new black and white palette.

In [7]:
```
chart_gray = alt.Chart(
    table
    ).mark_bar(color = "#666666", size = 15).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = alt.Axis(grid = False, orient = "top",
                        labelColor = "#888888", titleColor = '#888888',
                        titleFontWeight = 'normal', format = "%"),
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None)
    )

chart_light_gray_mix = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = alt.Axis(grid = False, orient = "top",
                        labelColor = "#888888", titleColor = '#888888',
                        titleFontWeight = 'normal', format = "%"),
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
```

```
    color = alt.condition(condition_decreased, alt.value('#bbbbbb'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_black_mix = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = alt.Axis(grid = False, orient = "top",
                        labelColor = "#888888", titleColor = '#888888',
                        titleFontWeight = 'normal', format = "%"),
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('black'), alt.value('#333333'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

label1_gray = alt.Chart(table.loc[table['Change'] < 0]).mark_text(align = 'left'
    x = alt.value(207),
    y = alt.Y('Brands', sort = None),
    text = alt.Text('Brands')
    )

label2_gray = alt.Chart(table.loc[table['Change'] > 0]).mark_text(align = 'right
    x = alt.value(192),
    y = alt.Y('Brands', sort = None),
    text = alt.Text('Brands')
    )

label_light_gray = alt.Chart(table.loc[table['Change'] < 0]).mark_text(align = '
    x = alt.value(207),
    y = alt.Y('Brands', sort = None),
    text = alt.Text('Brands'),
    color = alt.condition(condition_decreased,
                          alt.value('#bbbbbb'),
                          alt.value('#999999'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )


label_black = alt.Chart(table.loc[table['Change'] > 0]).mark_text(align = 'right
    x = alt.value(192),
    y = alt.Y('Brands', sort = None),
    text = alt.Text('Brands'),
    color = alt.condition(condition_increased,
                          alt.value('black'),
                          alt.value('#333333'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
```

```
    )

title_bw = alt.Chart(
    {"values": [{"text":  ["Cat food brands:"]}]}
    ).mark_text(
        size = 16, align = "left", dx = -200, dy = -270, fontWeight = 'normal',
        ).encode(
            text = "text:N"
            )

decreased_light_gray = alt.Chart(
    {"values": [{"text":  [
        'DECREASED'
         ]}]}
).mark_text(size = 11, align = "left", dx = -80, dy = -220, fontWeight = 700, cc
    text = "text:N"
)

increased_black = alt.Chart(
    {"values": [{"text":  [
            'INCREASED'
        ]}]}
).mark_text(size = 11, align = "left", dx = 20, dy = -220, fontWeight = 700, col
    text = "text:N"
)

black_and_white = (chart_gray + chart_light_gray_mix +
          chart_black_mix + label1_gray +
          label2_gray + label_light_gray + label_black +
          title_bw + title_bw_bold_2 + subtitle_bw +
          decreased_light_gray + increased_black + separation)

black_and_white.properties(width = 400).configure_view(stroke = None)
```

Out[7]:



Branding

Branding is a really important aspect of marketing. Most companies have set color palettes, logos and design. Tools such as Image Color Picker and Adobe Color can help you extract and implement those official palettes into themed data visualizations.

Cookie Clicker

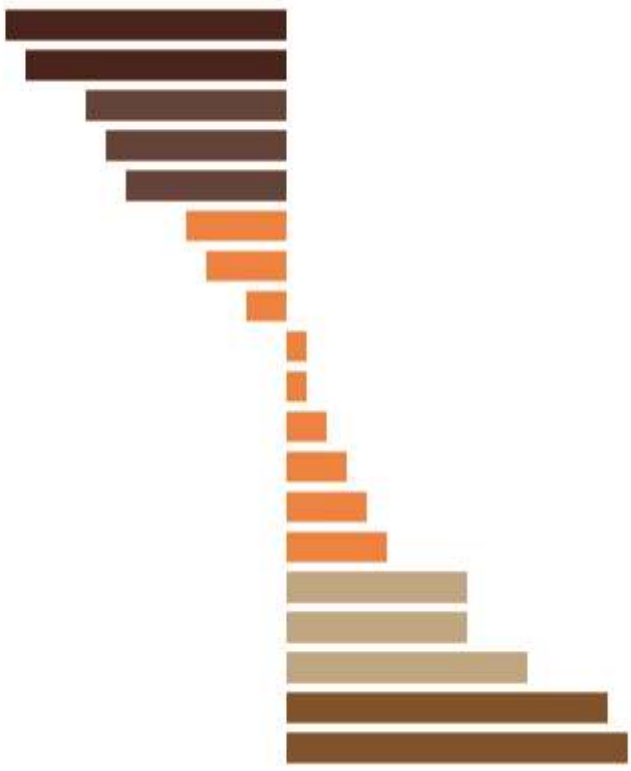
Cookie Clicker is an idle game created by the French programmer Orteil, and it has been open in a separate tab baking and accumulating cookies throughout the entirety of this project. Hence, it only seems fitting to pay homage with a dedicated color palette!

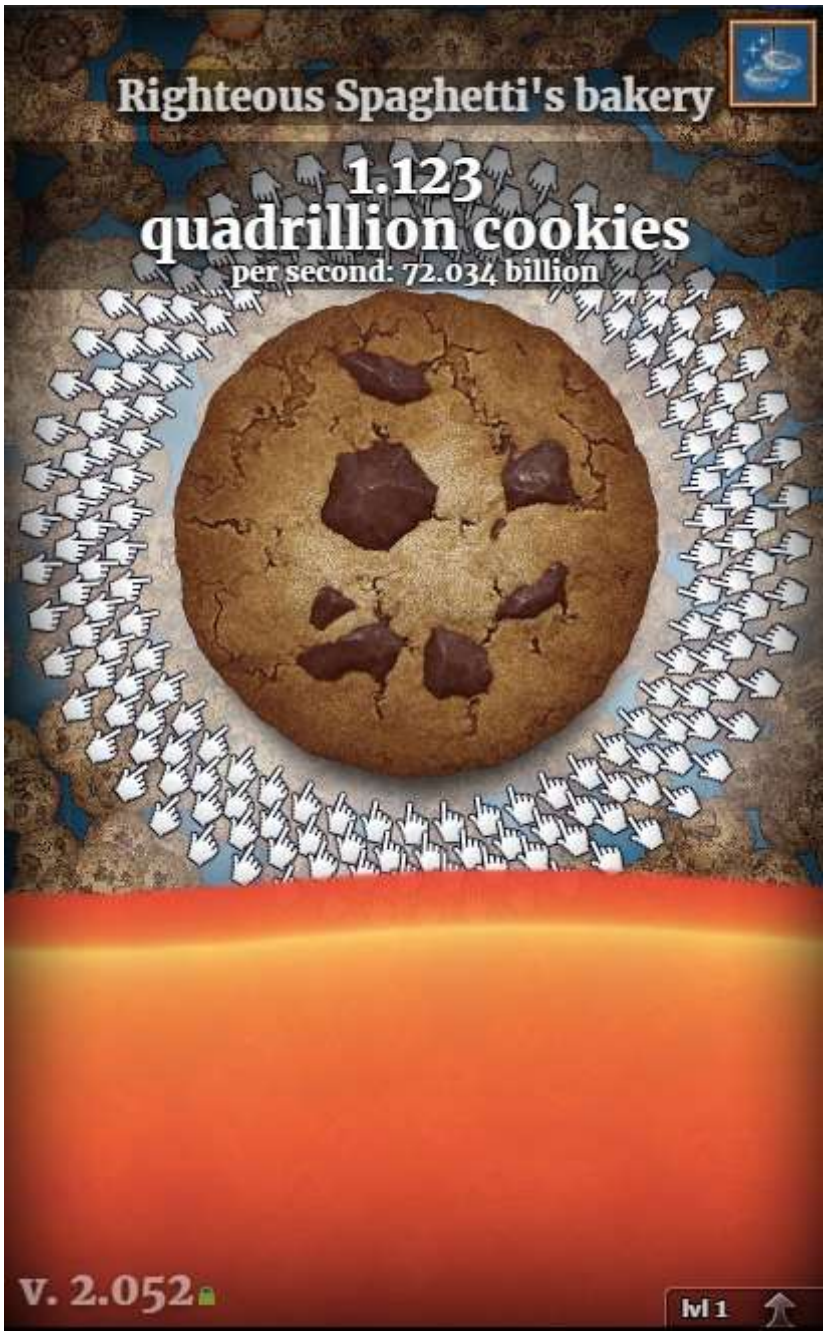
In [8]:

```
chart_middle = alt.Chart(
    table
    ).mark_bar(color = "#ee8241", size = 15).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None)
    )

chart_up = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
```

```
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_decreased, alt.value('#4a251d'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_down = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('#81532d'), alt.value('#c0a681'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

cookie_clicker = (chart_middle + chart_down + chart_up)

cookie_clicker.properties(width = 400).configure_view(stroke = None)
```

Out[8]:





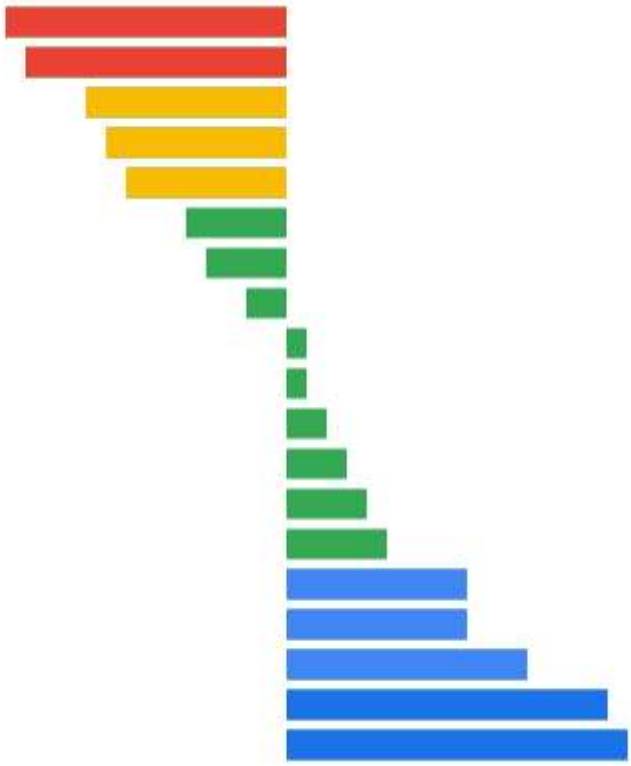
Google Maps

One of the most distinctive brand designs is the vibrant Google palette, widely employed in various apps associated with the company. Inspired by the Google Maps logo, which features precisely five colors, we will demonstrate how our graph looks when incorporating this design.

In [9]:
```
chart_middle = alt.Chart(
    table
    ).mark_bar(color = "#34A852", size = 15).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None)
```

```
    )

chart_up = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_decreased, alt.value('#EA4335'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_down = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('#1A73E8'), alt.value('#4285F4'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

google = (chart_middle + chart_down + chart_up)

google.properties(width = 400).configure_view(stroke = None)
```

Out[9]:



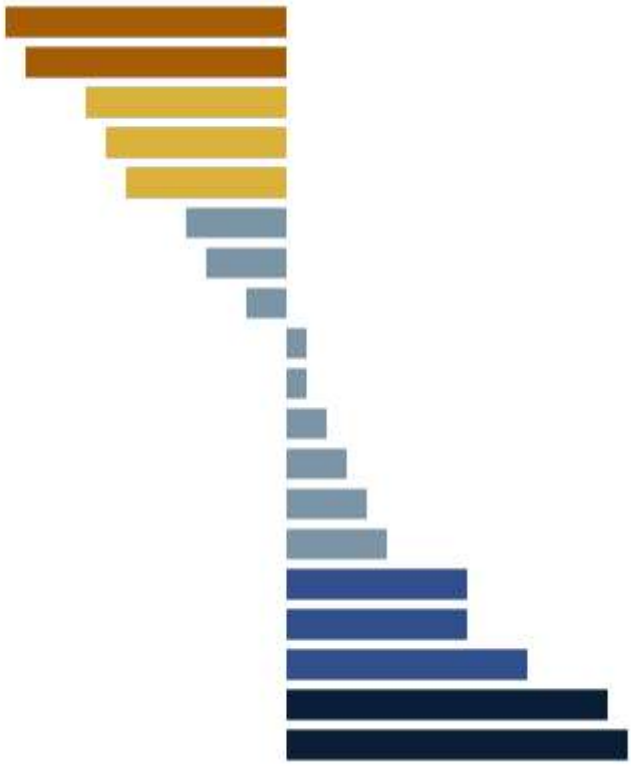
Paintings

Artist spend their lives dedicated to the study of colors and how we as humans perceive them. We can draw inspiration from their work when searching to evoke a specific emotion through our visualizations.

Starry Night by Vincent Van Gogh

In [10]:
```
chart_middle = alt.Chart(
    table
    ).mark_bar(color = "#7B95A6", size = 15).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None)
    )

chart_up = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_decreased, alt.value('#A65D05'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_down = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('#0B1E38'), alt.value('#304F8C'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

starry_night = (chart_middle + chart_down + chart_up)

starry_night.properties(width = 400).configure_view(stroke = None)
```

Out[10]:

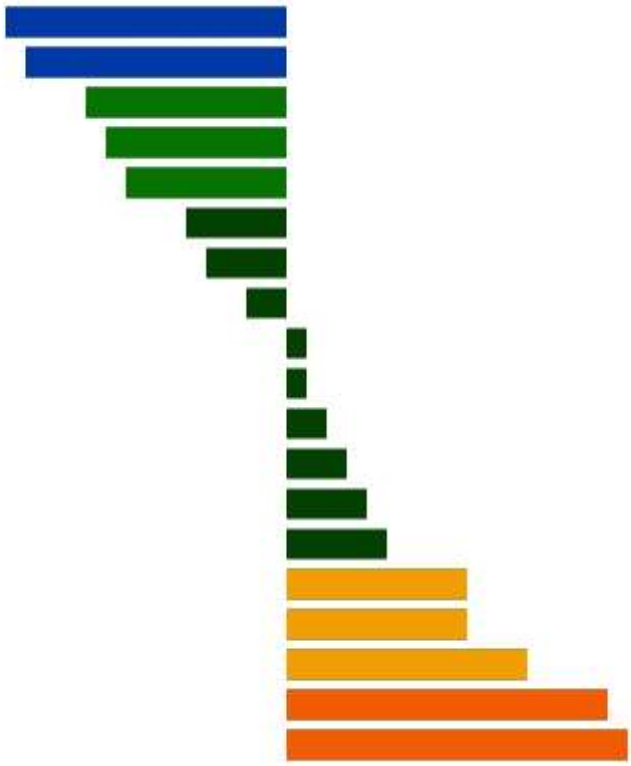


A Cuca by Tarsila do Amaral

In [11]:
```
chart_middle = alt.Chart(
    table
    ).mark_bar(color = "#034001", size = 15).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
```

```
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None)
    )

chart_up = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_decreased, alt.value('#0339A6'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_down = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('#F25C05'), alt.value('#F29F05'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

cuca = (chart_middle + chart_down + chart_up)

cuca.properties(width = 400).configure_view(stroke = None)
```

Out[11]:

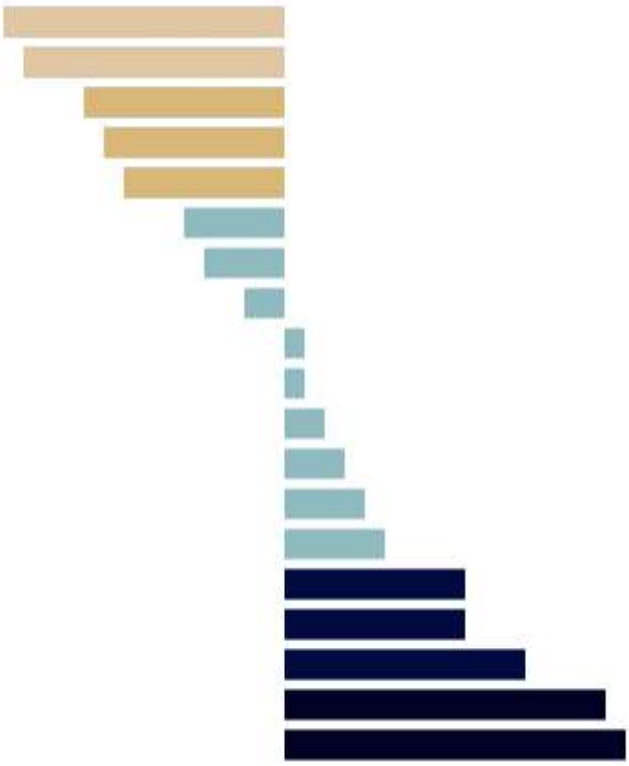


The Great Wave Off Kanagawa by Hokusai

In [12]:
```
chart_middle = alt.Chart(
    table
    ).mark_bar(color = "#8FBABF", size = 15).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
```

```
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None)
    )

chart_up = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_decreased, alt.value('#E0C6A3'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_down = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('#010326'), alt.value('#010B40'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

wave = (chart_middle + chart_down + chart_up)

wave.properties(width = 400).configure_view(stroke = None)
```

Out[12]:



Nature

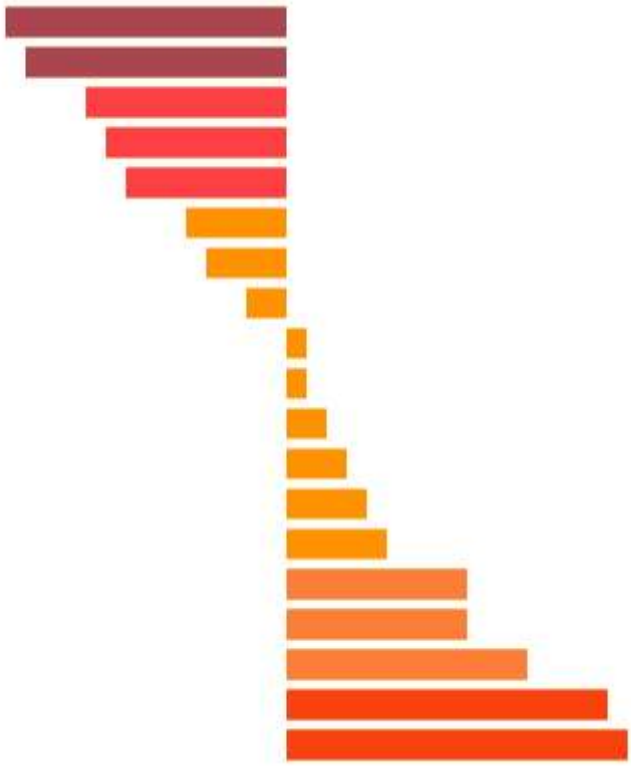
Sunset

This picture was taken by Sergio Mena Ferraira and can be found here.

In [13]:
```
chart_middle = alt.Chart(
    table
    ).mark_bar(color = "#FF9200", size = 15).encode(
    x = alt.X(
```

```
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None)
    )

chart_up = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_decreased, alt.value('#AA4650'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_down = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('#FA4210'), alt.value('#FD7E37'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

sunset = (chart_middle + chart_down + chart_up)

sunset.properties(width = 400).configure_view(stroke = None)
```

Out[13]:



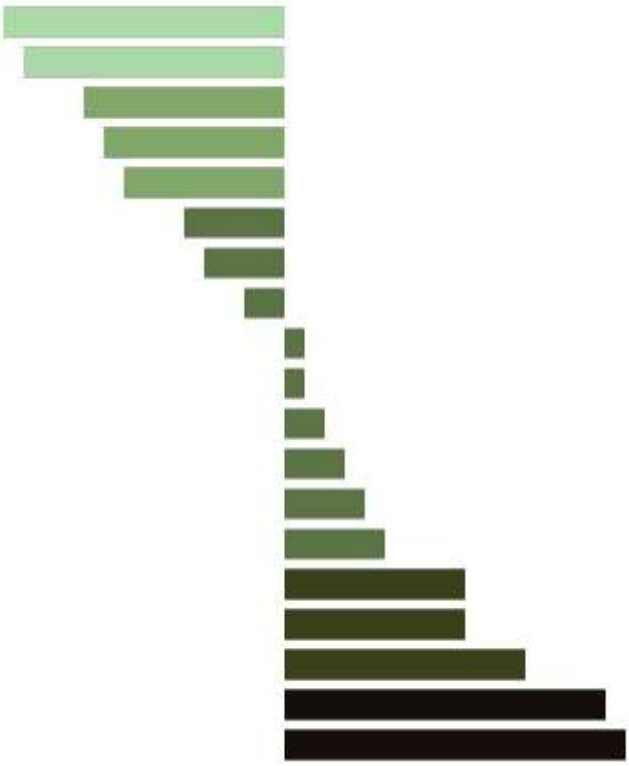
Forest

This picture was taken by Sam Abell and can be found here.

In [14]:
```
chart_middle = alt.Chart(
    table
    ).mark_bar(color = "#5C7346", size = 15).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
```

```
        ),
    y = alt.Y("Brands", sort = None, axis = None)
    )

chart_up = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_decreased, alt.value('#ABD9A9'), alt.value('
).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ["Fran's Recipe", 'Wholesome
                                                     'Lifestyle', 'Coat protecti
    )

chart_down = alt.Chart(table).mark_bar(
        size = 15
        ).encode(
    x = alt.X(
        "Change",
        scale = alt.Scale(domain = [-0.20, 0.20]),
        axis = None,
        title = None
        ),
    y = alt.Y("Brands", sort = None, axis = None),
    color = alt.condition(condition_increased,
                          alt.value('#140F09'), alt.value('#3B401B'))
    ).transform_filter(
    alt.FieldOneOfPredicate(field='Brands', oneOf = ['Feline Focus', 'Feline Gra
                                                     'Nutri Balance', 'Farm Fresh
    )

forest = (chart_middle + chart_down + chart_up)

forest.properties(width = 400).configure_view(stroke = None)
```

Out[14]:



Cell to create the .html

In [2]:
```
import nbconvert
import nbformat

with open('index.ipynb') as nb_file:
    nb_contents = nb_file.read()

# Convert using the ordinary exporter
notebook = nbformat.reads(nb_contents, as_version=4)

# HTML Export
```

```
html_exporter = nbconvert.HTMLExporter()
body, res = html_exporter.from_notebook_node(notebook)

# PDF Export
pdf_exporter = nbconvert.PDFExporter()
pdf_body, pdf_res = pdf_exporter.from_notebook_node(notebook)

# Create a dict mapping all image attachments to their base64 representations
images = {}
for cell in notebook['cells']:
    if 'attachments' in cell:
        attachments = cell['attachments']
        for filename, attachment in attachments.items():
            for mime, base64 in attachment.items():
                images[f'attachment:{filename}'] = f'data:{mime};base64,{base64}

# Fix up the HTML and write it to disk
for src, base64 in images.items():
    body = body.replace(f'src="{src}"', f'src="{base64}"')

# Write HTML to file
with open('index.html', 'w') as html_output_file:
    html_output_file.write(body)

# Write PDF to file
with open('index.pdf', 'wb') as pdf_output_file:
    pdf_output_file.write(pdf_body)
```

```
---------------------------------------------------------------------------
LatexFailed                               Traceback (most recent call last)
c:\Users\maisa\OneDrive\Documentos\GitHub\tcc\index.ipynb Cell 38 line 1
     <a href='vscode-notebook-cell:/c%3A/Users/maisa/OneDrive/Documentos/GitHub/tcc/index.ipynb#X52sZmlsZQ%3D%3D?line=13'>14</a> # PDF Export
     <a href='vscode-notebook-cell:/c%3A/Users/maisa/OneDrive/Documentos/GitHub/tcc/index.ipynb#X52sZmlsZQ%3D%3D?line=14'>15</a> pdf_exporter = nbconvert.PDFExporter()
---> <a href='vscode-notebook-cell:/c%3A/Users/maisa/OneDrive/Documentos/GitHub/tcc/index.ipynb#X52sZmlsZQ%3D%3D?line=15'>16</a> pdf_body, pdf_res = pdf_exporter.from_notebook_node(notebook)
     <a href='vscode-notebook-cell:/c%3A/Users/maisa/OneDrive/Documentos/GitHub/tcc/index.ipynb#X52sZmlsZQ%3D%3D?line=17'>18</a> # Create a dict mapping all image attachments to their base64 representations
     <a href='vscode-notebook-cell:/c%3A/Users/maisa/OneDrive/Documentos/GitHub/tcc/index.ipynb#X52sZmlsZQ%3D%3D?line=18'>19</a> images = {}

File c:\Users\maisa\anaconda3\envs\tcc\Lib\site-packages\nbconvert\exporters\pdf.py:200, in PDFExporter.from_notebook_node(self, nb, resources, **kw)
    198 tex_file = self.writer.write(latex, resources, notebook_name=notebook_name)
    199 self.log.info("Building PDF")
--> 200 self.run_latex(tex_file)
    201 if self.run_bib(tex_file):
    202     self.run_latex(tex_file)

File c:\Users\maisa\anaconda3\envs\tcc\Lib\site-packages\nbconvert\exporters\pdf.py:169, in PDFExporter.run_latex(self, filename, raise_on_failure)
    166 def log_error(command, out):
    167     self.log.critical("%s failed: %s\n%s", command[0], command, out)
--> 169 return self.run_command(
    170     self.latex_command, filename, self.latex_count, log_error, raise_on_failure
    171 )

File c:\Users\maisa\anaconda3\envs\tcc\Lib\site-packages\nbconvert\exporters\pdf.py:159, in PDFExporter.run_command(self, command_list, filename, count, log_function, raise_on_failure)
    157             if raise_on_failure:
    158                 msg = f'Failed to run "{command}" command:\n{out_str}'
--> 159                 raise raise_on_failure(msg)
    160             return False  # failure
    161 return True

LatexFailed: PDF creating failed, captured latex output:
Failed to run "xelatex notebook.tex -quiet" command:
Error: /undefined in RIFF
Operand stack:

Execution stack:
   %interp_exit   .runexec2   --nostringval--   --nostringval--   --nostringval--   2   %stopped_push   --nostringval--   --nostringval--   --nostringval--   false   1   %stopped_push   2015   1   3   %oparray_pop   2014   1   3   %oparray_pop   1998   1   3   %oparray_pop   1884   1   3   %oparray_pop   --nostringval--   %errorexec_pop   .runexec2   --nostringval--   --nostringval--   --nostringval--   2   %stopped_push   --nostringval--
Dictionary stack:
   --dict:975/1684(ro)(G)--   --dict:0/20(G)--   --dict:78/200(L)--
Current allocation mode is local
Current file position is 5
```

```
MiKTeX GPL Ghostscript 9.25: Unrecoverable error, exit code 1

dvipdfmx:fatal: pdf_link_obj(): passed invalid object.


No output PDF file written.


Sorry, but xelatex did not succeed.


The log file hopefully contains the information to get MiKTeX going again:


  C:\Users\maisa\AppData\Local\MiKTeX\miktex\log\xelatex.log
```